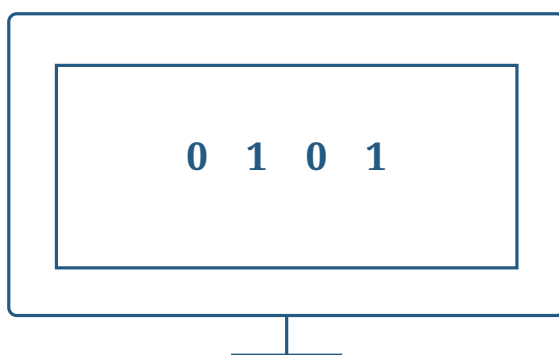


Tecnologie informatiche

Hardware, linguaggio digitale, reti e programmazione



Manuale per il primo anno della scuola secondaria di secondo grado

Pietro D'Ettòle

Versione 1.1

© 2026 Pietro D'Ettale. Tutti i diritti riservati.

Versione di pubblicazione: 1.1.

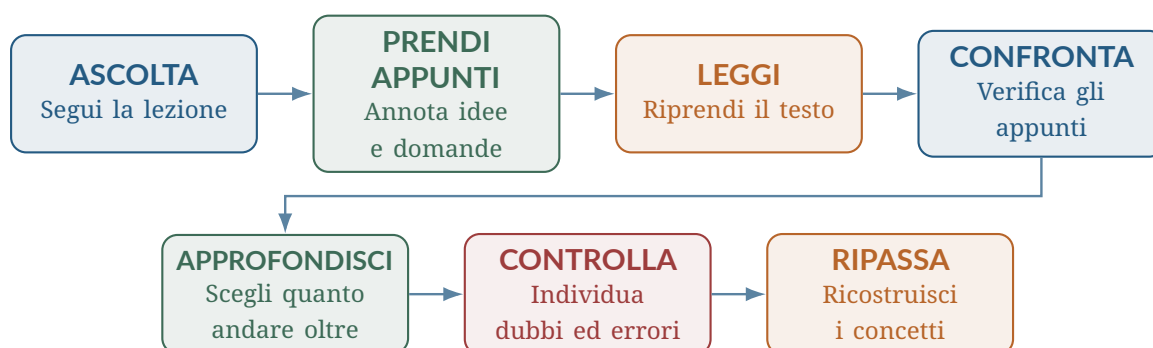
Quest'opera è resa disponibile gratuitamente per finalità didattiche. Non è consentita la vendita, la modifica o la redistribuzione commerciale senza autorizzazione scritta dell'autore.

Guida alla lettura

Un manuale da usare insieme alle lezioni

Questo manuale nasce dalla rielaborazione e dall'organizzazione degli appunti preparati per le lezioni di Tecnologie informatiche. È pensato come **strumento di appoggio**: aiuta a ritrovare spiegazioni, esempi, figure e collegamenti, ma non sostituisce l'ascolto, il confronto in classe, gli appunti personali e le indicazioni del docente.

Un possibile percorso di studio



Seguire il percorso del libro

I capitoli sono organizzati in modo progressivo: i concetti introdotti nelle prime parti vengono ripresi e applicati nelle successive. La lettura nell'ordine proposto facilita quindi la costruzione graduale delle conoscenze.

Consultare un singolo argomento

Il manuale può essere usato anche per ritrovare una spiegazione specifica. L'indice permette di raggiungere capitoli e sezioni; il glossario raccoglie le definizioni essenziali dei termini incontrati.

Approfondimento: scegliere il livello di approfondimento

Non tutte le parti devono essere affrontate nello stesso modo. Il testo principale costruisce il percorso di base; esempi, approfondimenti, curiosità e appendici permettono di ampliare lo studio secondo l'argomento, l'interesse personale e il livello di dettaglio che si desidera raggiungere.

Orientarsi tra le parti del manuale

Testo principale

Presenta e sviluppa i concetti fondamentali con un linguaggio progressivo.

Appendici

Offrono contenuti integrativi e percorsi pratici o laboratoriali.

Figure e tabelle

Rendono visibili strutture, confronti, percorsi e relazioni tra i concetti.

Glossario

Permette di ritrovare rapidamente il significato dei termini principali.

I riquadri ricorrenti

Obiettivi di apprendimento

Indicano conoscenze, abilità e competenze sviluppate nel capitolo o nell'appendice.

Parole chiave

Raccolgono i termini centrali da riconoscere e collegare durante lo studio.

Definizione

Fissa con precisione il significato di un termine o di un concetto importante.

Esempio

Mostra come un concetto viene applicato in una situazione concreta.

Approfondimento

Aggiunge dettagli, collegamenti o spiegazioni oltre il percorso essenziale.

Curiosità

Propone informazioni utili a collegare l'informatica alla realtà e alla sua storia.

Errore comune

Richiama confusioni frequenti e chiarisce come riconoscerle o evitarle.

Esercizi

Raccoglie domande e attività collegate ai contenuti della sezione.

Attività guidata

Accompagna passo dopo passo nella realizzazione di un compito o di un prodotto.

Preparazione alla scheda di laboratorio

Organizza problema, materiali, procedimento, osservazioni e risultati di un'attività pratica.

La lettura non è un percorso identico per tutti. Alcuni argomenti richiederanno una rilettura, altri il confronto con gli appunti o con una figura, altri ancora un approfondimento. Il manuale diventa utile quando viene collegato a ciò che è stato ascoltato, osservato e discusso durante la lezione.

Indice

Guida alla lettura

iii

I	Il computer: hardware e software	1
1	Che cos'è un computer?	3
1.1	I computer intorno a noi	4
1.1.1	Computer per uso generale e computer specializzati	4
1.2	La definizione di computer	6
1.2.1	Le operazioni fondamentali	6
1.2.2	Perché il computer è programmabile	8
1.3	Dati e informazioni	9
1.3.1	I principali tipi di dato	10
1.3.2	Il ruolo del contesto e dei metadati	11
1.3.3	La qualità dei dati	11
1.4	L'hardware del computer	13
1.4.1	Dispositivi di input, output e input/output	13
1.4.2	I componenti interni	14
1.4.3	Porte e connessioni del computer	21
1.4.4	Il modello di von Neumann	24
1.4.5	Dai computer tradizionali ai sistemi moderni	26
1.5	Il software	27
1.5.1	Software di sistema e software applicativo	27
1.5.2	Software open source e software proprietario	30
1.5.3	Modelli di distribuzione del software proprietario	31
1.5.4	Dalla licenza perpetua all'abbonamento	32
1.6	Esercizi riepilogativi	36
II	Il linguaggio digitale	39
2	Il linguaggio digitale	41
2.1	Misurare l'informazione	42
2.1.1	Il bit: l'unità elementare	42
2.1.2	Il byte	43
2.1.3	I multipli del byte	44
2.1.4	La regola delle conversioni	44
2.1.5	Capacità, dimensione e spazio disponibile	46
2.2	Il sistema binario	48
2.2.1	Sistemi di numerazione posizionali	48
2.2.2	Il sistema binario	49
2.2.3	Perché i computer usano il sistema binario	50
2.2.4	Conversione da binario a decimale	51
2.2.5	Conversione da decimale a binario	51
2.2.6	Zeri iniziali e numero di bit	53
2.3	I numeri binari con la virgola	54
2.3.1	Il valore delle cifre dopo la virgola	54
2.3.2	Le moltiplicazioni successive per 2	55

2.3.3	Quando la conversione termina?	56
2.3.4	Arrotondare a 10 cifre binarie	58
2.3.5	Casi particolari da riconoscere	59
2.3.6	Esercizi proposti	60
2.3.7	Soluzioni commentate	60
2.4	Le operazioni con i numeri binari	61
2.4.1	L'addizione binaria	61
2.4.2	La sottrazione binaria diretta	63
2.4.3	La moltiplicazione binaria	64
2.4.4	Controllare il risultato delle operazioni	65
2.5	Numeri negativi e complemento a 2	67
2.5.1	Perché occorre stabilire il numero di bit	67
2.5.2	L'intervallo dei valori rappresentabili	68
2.5.3	Ottenere l'opposto di un numero	69
2.5.4	Interpretare un numero negativo	71
2.5.5	Somma di numeri con segno	71
2.5.6	Eseguire una sottrazione usando soltanto la somma	72
2.5.7	L'overflow	74
2.6	Il sistema esadecimale	76
2.6.1	Un sistema di numerazione in base 16	76
2.6.2	Perché si usa il sistema esadecimale	77
2.6.3	Il valore posizionale delle cifre	77
2.6.4	Conversione da decimale a esadecimale	78
2.6.5	Conversione da binario a esadecimale	79
2.6.6	Conversione da esadecimale a binario	80
2.6.7	Conversioni tra le tre basi	81
2.7	I colori in formato esadecimale	82
2.7.1	Il modello RGB	83
2.7.2	La notazione #RRGGBB	84
2.7.3	Interpretare un codice colore	84
2.7.4	Da valori RGB decimali a codice esadecimale	85
2.7.5	Da codice esadecimale a valori RGB	86
2.7.6	Prevedere e modificare un colore	86
2.7.7	Applicazione facoltativa: costruire una piccola tavolozza	87
2.8	Quanto pesa una fotografia digitale?	88
2.8.1	Il pixel: il più piccolo elemento dell'immagine	88
2.8.2	La risoluzione e il numero totale di pixel	89
2.8.3	Quanti bit servono per descrivere un pixel?	90
2.8.4	Calcolare la dimensione teorica dell'immagine	91
2.8.5	Perché il file reale può essere più piccolo?	93
2.8.6	Ridimensionare una fotografia	94
2.9	Esercizi riepilogativi	97
III	Reti e Internet	101
3	Reti e Internet	103
3.1	Il viaggio di una fotografia	104
3.1.1	Prima tappa: dallo smartphone di Bob al router	104
3.1.2	Seconda tappa: dal router alla rete dell'operatore	105
3.1.3	Terza tappa: attraverso la rete di Internet	106
3.1.4	Quarta tappa: il server del servizio	107

3.1.5	Quinta tappa: dall'operatore mobile all'antenna 5G	107
3.1.6	Sesta tappa: la fotografia arriva ad Alice	108
3.1.7	Un viaggio composto da molti piccoli trasferimenti	108
3.1.8	Quando qualcosa non funziona	109
3.2	I mezzi trasmissivi	110
3.2.1	Trasmettere attraverso il rame	111
3.2.2	Trasmettere attraverso la luce: la fibra ottica	114
3.2.3	Trasmettere senza fili	115
3.2.4	Confrontare i mezzi trasmissivi	121
3.2.5	Come scegliere il mezzo adatto	122
3.3	I dispositivi di indirizzamento	124
3.3.1	Dalla trasmissione all'inoltro	124
3.3.2	L'hub: ripetere verso tutti	125
3.3.3	Lo switch: scegliere la porta corretta	126
3.3.4	Il router: collegare reti differenti	127
3.3.5	Confrontare i dispositivi di indirizzamento	129
3.4	Le topologie di rete	130
3.4.1	Topologia a bus	130
3.4.2	Topologia ad anello	131
3.4.3	Topologia a stella	132
3.4.4	Topologia ad albero	133
3.4.5	Topologia a maglia	134
3.4.6	Topologie ibride	135
3.4.7	Confrontare le topologie	136
3.4.8	Scegliere una topologia	137
3.5	Tipologie di rete per estensione	138
3.5.1	PAN: la rete personale	139
3.5.2	LAN: la rete locale	140
3.5.3	MAN: la rete metropolitana	141
3.5.4	WAN: la rete geografica	141
3.5.5	GAN: una rete di estensione globale	142
3.5.6	Confrontare le tipologie di rete	143
3.6	Il protocollo IP	144
3.6.1	La similitudine del servizio postale	144
3.6.2	Che cosa fa il protocollo IP	145
3.6.3	L'indirizzo IPv4	145
3.6.4	Indirizzi privati e indirizzi pubblici	146
3.6.5	Il NAT e la condivisione dell'indirizzo pubblico	147
3.6.6	Come viene assegnato un indirizzo	148
3.6.7	IPv6	148
3.6.8	Seguire il pacchetto di Bob	149
3.7	I protocolli di rete	151
3.7.1	Regole diverse per compiti diversi	151
3.7.2	Una panoramica dei protocolli più comuni	152
3.7.3	TCP: consegna controllata e ordinata	153
3.7.4	UDP: pochi controlli e minore attesa	154
3.7.5	I protocolli della posta elettronica	155
3.7.6	Un accenno alla posta elettronica certificata	156
3.8	Navigare sul Web	158
3.8.1	Che cos'è un URL	158
3.8.2	HTTP: richieste e risposte	159
3.8.3	HTTPS: proteggere la comunicazione	161

3.8.4	DNS: dai nomi agli indirizzi IP	162
3.8.5	Che cosa accade quando si preme Invio	163
3.8.6	Interpretare alcuni problemi di navigazione	164
3.9	Client-server e reti distribuite	165
3.9.1	Il modello client-server	166
3.9.2	Un server può servire molti client	167
3.9.3	Vantaggi e limiti del modello client-server	169
3.9.4	Il modello peer-to-peer	170
3.9.5	BitTorrent: distribuire un file tra molti peer	171
3.10	Esercizi riepilogativi	175
IV	Algoritmi e programmazione	177
4	Algoritmi e programmazione	179
4.1	Dal problema all'algoritmo	180
4.1.1	Che cos'è un algoritmo	180
4.1.2	Le caratteristiche di un buon algoritmo	181
4.1.3	Algoritmo e programma	181
4.1.4	Input, elaborazione e output	182
4.2	Variabili, assegnamento, input e output	184
4.2.1	La variabile come contenitore	184
4.2.2	Scegliere il nome delle variabili	184
4.2.3	L'assegnamento	185
4.2.4	Acquisire un dato: l'input	186
4.2.5	Comunicare un risultato: l'output	186
4.2.6	Seguire i valori passo dopo passo	187
4.3	I simboli dei diagrammi di flusso	188
4.3.1	Dalla descrizione al diagramma di flusso	188
4.3.2	Il cerchio: inizio e fine	189
4.3.3	Il rettangolo: operazioni e assegnamenti	189
4.3.4	Il parallelogramma IN: acquisire un dato	190
4.3.5	Il parallelogramma OUT: comunicare un risultato	190
4.3.6	Il rombo: condizioni e decisioni	191
4.3.7	Le frecce: la direzione dell'esecuzione	192
4.3.8	Un primo diagramma completo	192
4.3.9	Regole grafiche per diagrammi leggibili	193
4.4	Gli algoritmi sequenziali	195
4.4.1	Gli operatori aritmetici	195
4.4.2	Esempio guidato: sommare due numeri	196
4.4.3	Esempio guidato: area del rettangolo	197
4.4.4	Produrre più risultati	198
4.4.5	Esempio guidato: media di tre numeri	198
4.4.6	Applicazioni semplici	199
4.5	Le decisioni	200
4.5.1	Dalla sequenza alla scelta	201
4.5.2	Gli operatori di confronto	201
4.5.3	Decisione semplice: il maggiore tra due numeri	202
4.5.4	Decisioni annidate: positivo, negativo o zero	203
4.5.5	Calcolare uno sconto	204
4.5.6	Sconti su più fasce	205
4.5.7	Verificare entrambi i rami	205

4.6	Le condizioni composte	207
4.6.1	L'operatore AND: entrambe le condizioni devono essere vere . . .	207
4.6.2	L'operatore OR: almeno una condizione deve essere vera	208
4.6.3	Intervalli interni ed esterni	209
4.6.4	Applicare uno sconto con due requisiti	210
4.6.5	Scegliere tra AND e OR	211
4.7	Il resto della divisione: l'operatore modulo	212
4.7.1	Verificare se un numero è pari o dispari	212
4.7.2	Verificare la divisibilità e i multipli	213
4.7.3	Estrarre l'ultima cifra	213
4.7.4	Costruire condizioni con il modulo	214
4.8	Le iterazioni nei diagrammi di flusso	215
4.8.1	La freccia che torna indietro	215
4.8.2	Gli elementi fondamentali	215
4.8.3	Contatori, accumulatori e sentinelle	216
4.9	Esempio 1: sommare valori fino a una sentinella	217
4.9.1	Perché il primo input precede il rombo	218
4.9.2	Seguire il diagramma passo dopo passo	218
4.10	Esempio 2: chiedere nuovamente un dato non valido	219
4.10.1	Il controllo avviene dopo l'azione	220
4.10.2	Condizione di errore e condizione di validità	220
4.11	Esempio 3: elaborare un numero stabilito di misure	221
4.11.1	Seguire il valore del contatore	222
4.11.2	Controllare gli estremi	222
4.11.3	Calcolare anche la media	223
4.12	Progettare e verificare una ripetizione	224
4.12.1	Progettare prima di disegnare	224
4.12.2	Verificare la terminazione	225
4.12.3	Usare casi di prova	225
4.13	I numeri pseudocasuali	226
4.13.1	Il seed: il punto di partenza della sequenza	226
4.13.2	La funzione rand ()	227
4.13.3	Generare un numero da 1 a 10	228
4.13.4	La formula generale per un intervallo	228
4.13.5	Verificare l'intervallo prima di disegnare	229
4.14	Numeri casuali e decisioni	230
4.14.1	Simulare il lancio di un dado	230
4.14.2	Simulare il lancio di una moneta	231
4.14.3	Generare un numero e verificarne la parità	232
4.14.4	Simulare una temperatura	232
4.14.5	Simulare l'umidità del terreno	233
4.14.6	Simulare un valore di pH	234
4.14.7	Scegliere correttamente l'intervallo e la condizione	234
4.15	Esercizi riepilogativi	236
V	Appendici operative e cittadinanza digitale	241
A	Arduino e il controllo del mondo fisico	243
A.1	Dal diagramma di flusso al mondo fisico	244
A.1.1	Osservare, elaborare e agire	244
A.1.2	Dal diagramma al sistema	245

A.2	Che cos'è Arduino	246
A.2.1	Computer e microcontrollore	247
A.2.2	La scheda Arduino Uno	247
A.3	Componenti e sicurezza	250
A.3.1	La breadboard	250
A.3.2	LED e resistenza	250
A.3.3	Il pulsante	251
A.3.4	Sensori e attuatori	252
A.3.5	Regole di sicurezza	252
A.4	L'ambiente di programmazione	253
A.4.1	Dal codice all'esecuzione	254
A.4.2	Verificare non significa collaudare	254
A.4.3	Commenti e leggibilità	254
A.4.4	Maiuscole, parentesi e punto e virgola	255
A.5	La struttura di un programma Arduino	256
A.5.1	La funzione <code>setup()</code>	256
A.5.2	La funzione <code>loop()</code>	256
A.5.3	Dove collocare le istruzioni	257
A.5.4	Parentesi graffe e blocchi	257
A.6	Variabili e tipi	258
A.6.1	Dichiarare una variabile	259
A.6.2	Dichiarazione e inizializzazione	259
A.6.3	Il tipo <code>int</code>	260
A.6.4	Il tipo <code>float</code>	260
A.6.5	Il tipo <code>bool</code>	260
A.6.6	Scegliere nomi significativi	260
A.6.7	Assegnamento e confronto	261
A.7	Comunicare con il monitor seriale	262
A.7.1	Avviare la comunicazione	263
A.7.2	Stampare testo e valori	263
A.7.3	Testo e variabili nello stesso output	264
A.8	Le uscite digitali	265
A.8.1	Configurare il pin come uscita	265
A.8.2	Impostare lo stato del pin	266
A.8.3	La funzione <code>delay()</code>	266
A.9	Gli ingressi digitali e le decisioni	267
A.9.1	Leggere il valore di un pin	268
A.9.2	Dalla decisione del diagramma all'istruzione <code>if</code>	268
A.9.3	Un programma completo: pulsante e LED	269
A.10	Gli ingressi analogici	270
A.10.1	Leggere un ingresso analogico	270
A.10.2	Sensore reale e grandezza fisica	271
A.10.3	Osservare i valori sul monitor seriale	271
A.11	Applicazioni agricole e chimiche	272
A.11.1	Segnalare un terreno troppo asciutto	272
A.11.2	Controllare una soglia di temperatura	273
A.11.3	Verificare un intervallo di pH	274
A.11.4	Selezionare un campione in modo pseudocasuale	274
A.12	Esercitazione conclusiva	275

B	Creare e usare una mappa concettuale	277
B.1	Che cos'è una mappa concettuale	278
B.1.1	Gli elementi della mappa	278
B.1.2	Gerarchia e collegamenti trasversali	279
B.1.3	Utilità e confronto con altri strumenti	279
B.2	Come si costruisce una mappa concettuale	280
B.3	Esempio guidato: la mappa del Capitolo 1	284
B.3.1	Domanda guida e concetti essenziali	284
B.3.2	Dalla lista alla struttura	285
B.3.3	Dalla prima bozza alla mappa completa	286
B.3.4	Leggere e utilizzare la mappa	288
B.4	Errori comuni e controllo di qualità	290
B.5	Laboratorio: costruire la propria mappa	291
B.5.1	Consegna e vincoli	291
B.5.2	Realizzazione e revisione	292
B.5.3	Valutazione e autovalutazione	293
C	Uso consapevole delle tecnologie e benessere digitale	295
C.1	Tecnologia: opportunità e responsabilità	296
C.1.1	Usare uno strumento per raggiungere uno scopo	296
C.1.2	La quantità di tempo non racconta tutto	297
C.1.3	Responsabilità verso se stessi e verso gli altri	297
C.2	Attenzione, notifiche e tempo digitale	298
C.2.1	Interruzione e ripresa del compito	299
C.2.2	Il falso mito del multitasking	299
C.2.3	Perché è facile continuare	299
C.2.4	Proteggere il tempo di studio	300
C.3	Sonno, postura e benessere fisico	301
C.3.1	Dispositivi e sonno	301
C.3.2	Affaticamento visivo	302
C.3.3	Postura e organizzazione della postazione	302
C.3.4	Sedentarietà e movimento	303
C.3.5	Cuffie e ascolto sicuro	303
C.4	Social network, identità e confronto sociale	304
C.4.1	Il profilo è una rappresentazione parziale	304
C.4.2	Reazioni, visualizzazioni e ricerca di approvazione	305
C.4.3	Perché due persone vedono feed diversi	305
C.5	Uso problematico, relazioni e comportamenti dannosi	306
C.5.1	Riconoscere un uso problematico	306
C.5.2	Le azioni online hanno effetti reali	307
C.5.3	Cyberbullismo ed esclusione digitale	307
C.6	Costruire abitudini digitali equilibrate	308
C.6.1	Modificare l'ambiente, non soltanto affidarsi alla volontà	309
C.6.2	Scegliere una modifica osservabile	309
C.6.3	Un piano personale di benessere digitale	309
	Glossario	311
	Indice analitico	321

Parte I

Il computer: hardware e software

1. Che cos'è un computer?

Computer fissi, notebook, smartphone e tablet sono entrati stabilmente nella vita quotidiana. Altri computer sono meno visibili, perché si trovano all'interno di automobili, televisori, elettrodomestici, strumenti di misura e macchinari automatici. Nonostante le differenze nell'aspetto e nelle dimensioni, tutti questi sistemi condividono alcuni principi fondamentali.

Questo capitolo introduce il concetto di computer e ne descrive le caratteristiche essenziali. Verranno inoltre chiariti i concetti di *dato* e *informazione* e verranno presentati l'hardware, le periferiche, i principali componenti interni e le porte utilizzate per collegare dispositivi e reti. Dopo il modello di von Neumann, verrà affrontato il software: dalle funzioni del sistema operativo alle applicazioni, fino alle licenze open source e proprietarie e ai principali modelli di distribuzione commerciale. Il capitolo si concluderà con una scheda di laboratorio per l'osservazione diretta di un computer desktop.

Conoscenze	Abilità	Competenze
• significato di computer ed elaboratore; • concetti di dato e informazione; • dispositivi di input, output e input/output; • principali componenti interni del computer; • porte USB, video, audio, di rete e di alimentazione; • modello di von Neumann; • software di sistema, sistema operativo e software applicativo; • licenze open source e proprietarie; • modelli freeware, payware, trial, freemium e abbonamento.	• riconoscere computer di forme differenti; • classificare periferiche di input, output e input/output; • riconoscere CPU, RAM, memorie di massa, GPU e alimentatore; • collegare i componenti alle rispettive funzioni; • distinguere software di sistema e software applicativo; • riconoscere licenza e modello di distribuzione di un programma.	• interpretare il computer come un sistema che riceve, elabora, memorizza e comunica dati; • descrivere in modo corretto il funzionamento generale di un dispositivo digitale; • interpretare i componenti reali attraverso il modello di von Neumann; • preparare in sicurezza l'osservazione e lo smontaggio di un computer; • valutare in modo consapevole caratteristiche, licenza e costi di un software.

1.1 I computer intorno a noi

Nel linguaggio comune, il termine **computer** richiama spesso l'immagine di un personal computer: un dispositivo dotato di schermo, tastiera e mouse, utilizzato per scrivere documenti, navigare nel Web, comunicare o eseguire programmi. Questa immagine, tuttavia, descrive soltanto una delle molte forme che un computer può assumere.

Anche uno smartphone e un tablet sono computer. Possiedono infatti un processore, una memoria, dispositivi per ricevere e comunicare dati e un insieme di programmi che ne controllano il funzionamento. Lo stesso vale per una console per videogiochi o per un televisore intelligente.

Esistono inoltre computer meno evidenti, incorporati all'interno di altri apparecchi. Una moderna automobile contiene numerose unità elettroniche che controllano, per esempio, il motore, i sistemi di sicurezza, la climatizzazione e l'intrattenimento. Anche una lavatrice può contenere un piccolo computer che legge i comandi impostati, controlla alcuni sensori e gestisce le diverse fasi del lavaggio.

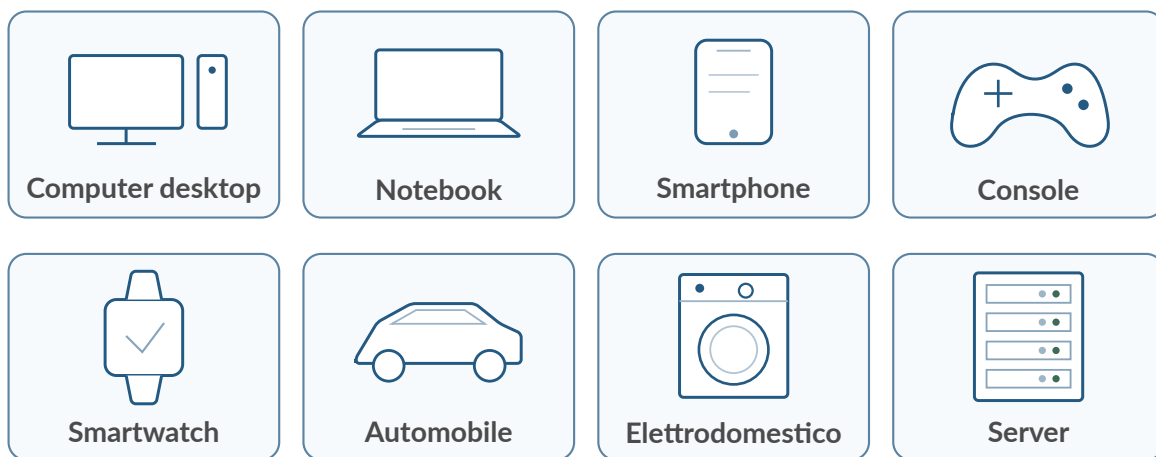


Figura 1.1: Esempi di computer riconoscibili e di sistemi di elaborazione incorporati in altri dispositivi.

Definizione: sistema incorporato

Un **sistema incorporato**, o *embedded system*, è un computer integrato all'interno di un dispositivo più grande e progettato per svolgere una funzione specifica o un insieme limitato di funzioni.

Un sistema incorporato è normalmente meno versatile di un personal computer. Un notebook può eseguire programmi molto diversi tra loro; la centralina di un elettrodomestico, invece, è progettata soprattutto per controllare quel particolare apparecchio. Entrambi, tuttavia, ricevono dati, eseguono istruzioni e producono risultati.

1.1.1 Computer per uso generale e computer specializzati

I computer possono essere distinti, in modo semplificato, in due grandi categorie.

- I **computer per uso generale** possono eseguire numerosi programmi e svolgere attività molto diverse. Ne sono esempi i computer desktop, i notebook, i tablet e gli smartphone.

- I **computer specializzati** sono progettati principalmente per un compito preciso. Ne sono esempi le centraline elettroniche, i sistemi di controllo industriale e molti strumenti automatici.

La distinzione non è sempre assoluta. Uno smartphone è un dispositivo per uso generale, ma può anche controllare apparecchi specializzati attraverso applicazioni dedicate. Allo stesso modo, alcuni macchinari includono computer abbastanza potenti da svolgere più funzioni.

Curiosità: un computer può non avere uno schermo

Uno schermo non è indispensabile perché un dispositivo possa essere considerato un computer. Molti sistemi incorporati funzionano senza mostrare direttamente informazioni all'utente. Possono ricevere dati da sensori, elaborarli e comandare altri dispositivi in modo automatico.

Errore comune: il monitor non è "il computer"

Nel linguaggio quotidiano il monitor viene talvolta chiamato impropriamente computer. Il monitor è invece un dispositivo che visualizza i risultati prodotti dal sistema. In un computer fisso, le principali componenti di elaborazione e memorizzazione sono generalmente contenute nel case.

Esercizi: sezione 1.1

1. Indicare quali dei seguenti dispositivi possono contenere un computer: frigorifero tradizionale, smartphone, automobile moderna, libro cartaceo, console per videogiochi, lavatrice elettronica.
2. Spiegare perché uno smartphone può essere considerato un computer.
3. Descrivere la differenza generale tra un computer per uso generale e un computer specializzato.
4. Individuare tre dispositivi di uso quotidiano che potrebbero contenere un sistema incorporato. Per ciascuno, indicare almeno una funzione controllata dal sistema.
5. Stabilire se le seguenti affermazioni sono vere o false. Correggere quelle false.
 - a) Tutti i computer possiedono necessariamente una tastiera.
 - b) Una console per videogiochi contiene componenti di elaborazione e memoria.
 - c) Un sistema incorporato è sempre più versatile di un notebook.
 - d) Un monitor è un dispositivo utilizzato per visualizzare informazioni.
6. Ricercare un esempio di computer specializzato non citato nel testo e descriverne brevemente la funzione.

1.2 La definizione di computer

Il computer può essere definito come un **elaboratore elettronico digitale programmabile**. Questa espressione riassume le principali caratteristiche comuni ai diversi tipi di computer, ma richiede di essere analizzata con attenzione.

Definizione: computer

Un **computer** è una macchina elettronica digitale e programmabile in grado di ricevere dati, elaborarli seguendo le istruzioni contenute in un programma, memorizzarli e comunicare i risultati dell'elaborazione.

La definizione contiene diversi concetti fondamentali:

- è una **macchina**, perché è un sistema costruito per svolgere operazioni;
- è **elettronica**, perché utilizza circuiti e componenti elettronici;
- è **digitale**, perché rappresenta ed elabora i dati attraverso valori discreti, che all'interno dei computer sono ricondotti a sequenze di 0 e 1;
- è **programmabile**, perché il suo comportamento dipende dalle istruzioni dei programmi eseguiti;
- è un **elaboratore**, perché trasforma dati iniziali in risultati utili.

I termini *elettronico*, *digitale* e *programmabile* saranno approfonditi gradualmente. Per il momento è importante comprendere che il computer non svolge una sola funzione prestabilita: può eseguire attività differenti caricando programmi differenti.

1.2.1 Le operazioni fondamentali

Il funzionamento generale di un computer può essere rappresentato attraverso quattro operazioni fondamentali:

1. **acquisizione dei dati** (input);
2. **elaborazione**;
3. **memorizzazione**;
4. **comunicazione dei risultati** (output).

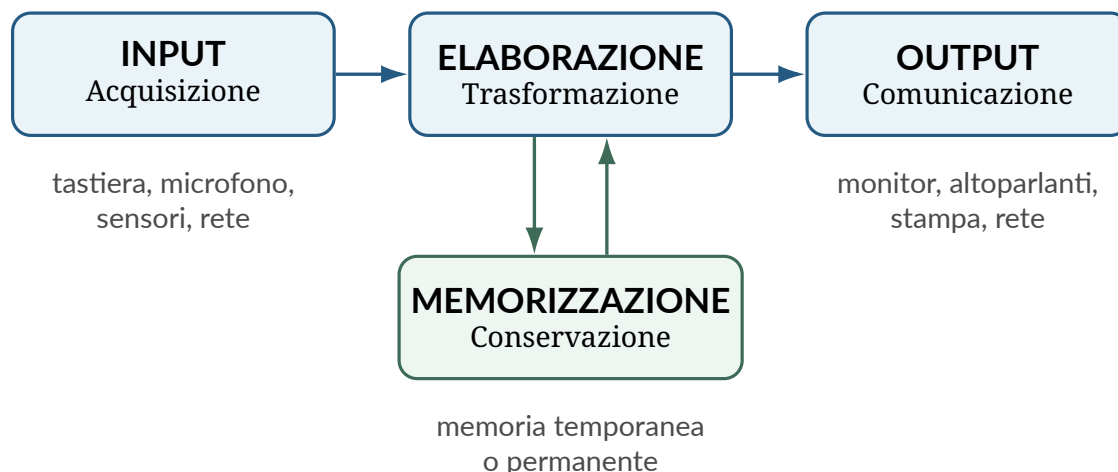


Figura 1.2: Modello generale del funzionamento di un computer. La memoria scambia dati con l'elaborazione durante l'esecuzione delle attività.

Lo schema non deve essere interpretato come una sequenza rigida che avviene una sola volta. Durante l'esecuzione di un programma, il computer può ricevere continuamente nuovi dati, accedere più volte alla memoria ed effettuare numerose elaborazioni prima di produrre il risultato finale.

Acquisizione dei dati. I dati e i comandi possono essere forniti dall'utente oppure provenire da altri dispositivi. Una tastiera acquisisce i caratteri digitati, un microfono acquisisce suoni, una fotocamera acquisisce immagini e un sensore acquisisce una grandezza fisica.

Elaborazione. I dati vengono trasformati seguendo le istruzioni di un programma. L'elaborazione può consistere in un calcolo, in un confronto, nell'ordinamento di informazioni, nella modifica di un'immagine o in molte altre operazioni.

Memorizzazione. I dati, i programmi e i risultati possono essere conservati temporaneamente o in modo permanente. La distinzione tra memoria temporanea e memoria di massa sarà approfondita nella [sezione 1.4](#).

Comunicazione dei risultati. Il risultato può essere mostrato su uno schermo, riprodotto attraverso altoparlanti, stampato, trasmesso attraverso una rete oppure utilizzato per comandare un altro dispositivo.

Esempio: scrittura e salvataggio di un documento

Quando viene digitata una frase, la tastiera invia al computer i dati corrispondenti ai tasti premuti. Il programma di videoscrittura elabora tali dati e li rappresenta sullo schermo. Durante il lavoro, il contenuto viene mantenuto nella memoria temporanea; con il comando di salvataggio, viene registrato in una memoria permanente sotto forma di file.

1.2.2 Perché il computer è programmabile

Una calcolatrice elementare e un personal computer sono entrambi dispositivi elettronici capaci di elaborare dati, ma il personal computer può svolgere una varietà molto maggiore di compiti. La differenza principale non dipende soltanto dalla potenza dei componenti: dipende soprattutto dalla possibilità di eseguire programmi differenti.

Un **programma** è un insieme ordinato di istruzioni che specifica quali operazioni devono essere eseguite. Un programma di videoscrittura contiene istruzioni per inserire, modificare, formattare e salvare testi; un browser contiene istruzioni per richiedere, interpretare e visualizzare contenuti provenienti dalla rete.

La stessa macchina può quindi assumere funzioni differenti in base al software utilizzato. Questo principio rende il computer uno strumento generale e adattabile.

Approfondimento: origine del termine

La parola inglese *computer* deriva dal verbo *to compute*, che significa “calcolare”. In origine il termine indicava anche una persona incaricata di svolgere calcoli. In italiano si usa spesso la parola **elaboratore**, più adatta a ricordare che i computer moderni non trattano soltanto numeri, ma anche testi, immagini, suoni e video.

Curiosità: il computer esegue istruzioni elementari

Le attività apparentemente complesse vengono scomposte in un grandissimo numero di istruzioni molto semplici. Anche aprire una fotografia, riprodurre un video o mostrare una pagina Web richiede l'esecuzione coordinata di numerose operazioni elementari.

Errore comune: il computer non comprende come una persona

Il computer elabora simboli e dati secondo regole definite dai programmi. Anche quando un sistema produce risposte simili a quelle umane, il suo funzionamento dipende da software, modelli matematici, dati e componenti hardware. Il termine “comprendere” deve quindi essere usato con cautela quando si descrive una macchina.

Esercizi: sezione 1.2

1. Completare la frase: un computer è una macchina _____, _____ e _____ in grado di ricevere, elaborare, memorizzare e comunicare dati.
2. Spiegare con parole proprie il significato del termine *elaboratore*.
3. Associare ciascuna attività alla fase prevalente: input, elaborazione, memorizzazione oppure output.
 - a) Visualizzare un'immagine sul monitor.
 - b) Registrare un file su un SSD.
 - c) Premere un tasto sulla tastiera.
 - d) Calcolare la media di una serie di valori.
 - e) Riprodurre un suono attraverso gli altoparlanti.

Esercizi: sezione 1.2 (continua)

4. Descrivere le quattro operazioni fondamentali compiute durante l'acquisizione e la modifica di una fotografia digitale.
5. Spiegare perché la memorizzazione non deve essere considerata necessariamente come una fase che avviene soltanto dopo l'elaborazione.
6. Indicare quale caratteristica consente allo stesso computer di essere utilizzato per scrivere un testo, riprodurre un video ed eseguire un calcolo.
7. Stabilire se le seguenti affermazioni sono vere o false. Correggere quelle false.
 - a) Un computer può elaborare soltanto dati numerici.
 - b) Un programma è un insieme ordinato di istruzioni.
 - c) L'output è sempre rappresentato da un'immagine sul monitor.
 - d) Un computer può ricevere dati anche da un sensore.
8. Scegliere un'attività svolta abitualmente con uno smartphone e descrivere quali dati vengono acquisiti, quale elaborazione viene compiuta e quale risultato viene prodotto.

1.3 Dati e informazioni

Ogni attività svolta da un computer riguarda, in modo diretto o indiretto, la raccolta, l'elaborazione, la memorizzazione oppure la comunicazione di dati. I termini *dato* e *informazione* vengono spesso usati come sinonimi, ma indicano concetti differenti.

Definizione: dato

Un **dato** è la rappresentazione elementare di un fatto, di una misura, di un simbolo o di un evento. Preso isolatamente, può non possedere un significato completo.

Definizione: informazione

Un'**informazione** è un dato, o un insieme di dati, interpretato all'interno di un contesto e organizzato in modo da assumere un significato utile.

Il numero 24, considerato da solo, è un dato. Non permette di stabilire che cosa rappresenti: potrebbe indicare un'età, una quantità, un'ora, una temperatura oppure il numero di una stanza. L'espressione "temperatura dell'ambiente: 24 °C" comunica invece un'informazione, perché associa al valore un significato, un'unità di misura e un contesto.

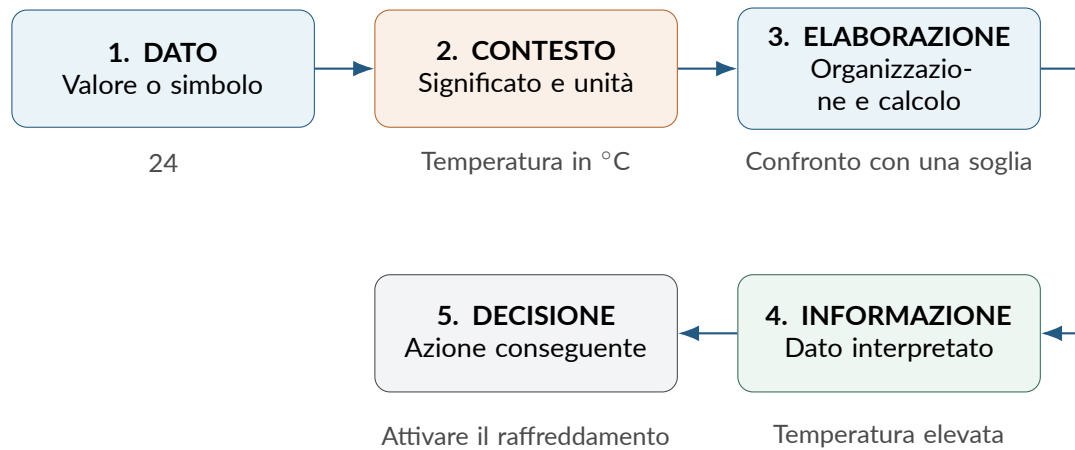


Figura 1.3: Un dato acquista significato quando viene inserito in un contesto e interpretato. L'informazione ottenuta può essere utilizzata per prendere una decisione.

Il passaggio dal dato all'informazione non richiede necessariamente un calcolo complesso. Può essere sufficiente aggiungere un'etichetta, ordinare alcuni valori, confrontarli con una soglia oppure collegarli al momento e al luogo in cui sono stati raccolti.

Esempio: dai dati a un'informazione

I valori 18, 21, 24 e 27 sono dati numerici. Sapere che rappresentano temperature misurate ogni ora permette di interpretarli. Ordinandoli nel tempo si ricava inoltre l'informazione che la temperatura è aumentata progressivamente.

1.3.1 I principali tipi di dato

Un computer può elaborare contenuti molto diversi. Tutti devono però essere rappresentati in forma digitale prima di poter essere memorizzati e trattati. In modo semplificato, è possibile distinguere le categorie riportate nella [tabella 1.1](#).

Tabella 1.1: Alcuni tipi di dato elaborati dai computer.

Tipo di dato	Descrizione	Esempi
Numerico	Rappresenta quantità, misure o risultati di calcoli.	Età, prezzo, temperatura, distanza, percentuale.
Testuale o alfanumerico	È formato da lettere, cifre, spazi e altri caratteri.	Nome, indirizzo, codice fiscale, messaggio.
Logico	Può assumere due valori contrapposti.	Vero/falso, acceso/spento, sì/no.
Immagine	Rappresenta elementi visivi mediante punti e valori di colore.	Fotografia, disegno, grafico, icona.
Audio	Rappresenta suoni, voce o musica.	Registrazione, brano musicale, messaggio vocale.
Video	Combina una sequenza di immagini, generalmente accompagnata da audio.	Filmato, videolezione, ripresa di una telecamera.

La categoria dipende anche dall'uso che viene fatto del dato. Una sequenza come 2026 può essere trattata come numero quando viene utilizzata in un calcolo, oppure come testo quando costituisce una parte di un codice identificativo.

Errore comune: le cifre non indicano sempre un numero

Un numero di telefono, un codice postale e il numero di una carta non vengono normalmente utilizzati per effettuare calcoli. Anche se sono formati da cifre, vengono quindi trattati spesso come dati testuali. Lo zero iniziale di un numero di telefono, per esempio, deve essere conservato e non ha il significato matematico di una cifra non significativa.

1.3.2 Il ruolo del contesto e dei metadati

Per interpretare correttamente un dato è necessario conoscere il contesto in cui è stato prodotto. Una misura è poco utile se non sono note l'unità utilizzata, la data, il luogo o lo strumento con cui è stata ottenuta. Informazioni di questo tipo vengono spesso registrate insieme al contenuto principale.

Definizione: metadato

Un **metadato** è un dato che descrive un altro dato e ne facilita l'identificazione, l'organizzazione o l'interpretazione.

Una fotografia digitale contiene i valori necessari per rappresentare l'immagine, ma può includere anche metadati come data e ora dello scatto, dimensioni, modello della fotocamera e, se attivata, posizione geografica. Un documento può essere accompagnato dal nome dell'autore, dalla data dell'ultima modifica e dal tipo di file.

Approfondimento: i metadati possono contenere informazioni personali

Prima di condividere una fotografia o un documento è utile ricordare che i metadati possono rivelare informazioni non immediatamente visibili, come il luogo in cui è stata scattata un'immagine o il nome dell'autore del file. Per questo motivo alcuni servizi eliminano automaticamente parte dei metadati, mentre altri li conservano.

1.3.3 La qualità dei dati

Un'elaborazione può produrre risultati affidabili soltanto se i dati utilizzati sono adeguati. La qualità di un insieme di dati dipende da diverse caratteristiche:

- **correttezza:** i valori devono rappresentare in modo fedele ciò che è stato osservato o misurato;
- **completezza:** non devono mancare elementi necessari all'analisi;
- **coerenza:** dati riferiti allo stesso fenomeno non devono contraddirsi senza una spiegazione;
- **aggiornamento:** le informazioni devono essere sufficientemente recenti rispetto allo scopo;

- **pertinenza:** devono essere raccolti dati utili al problema da risolvere.

Un errore di digitazione, un sensore non calibrato oppure una domanda formulata male possono produrre dati inesatti. Il computer può elaborarli con grande velocità, ma non è in grado di trasformare automaticamente dati sbagliati in risultati corretti.

Curiosità: Garbage in, garbage out

In informatica viene spesso utilizzata l'espressione inglese *Garbage in, garbage out*, abbreviata in **GIGO**. Significa che dati di scarsa qualità forniti in ingresso tendono a produrre risultati di scarsa qualità in uscita, anche quando il programma funziona correttamente.

Esercizi: sezione 1.3

1. Completare le definizioni.
 - a) Un dato è la rappresentazione elementare di un _____, di una misura, di un simbolo o di un evento.
 - b) Un'informazione è un dato interpretato all'interno di un _____.
 - c) Un metadato è un dato che _____ un altro dato.
2. Per ciascuno dei seguenti dati, indicare due possibili significati: 18, 100, 07:30, PV.
3. Classificare i seguenti elementi come dati numerici, testuali, logici, immagini, audio oppure video: fotografia, cognome, temperatura, registrazione vocale, stato acceso/spento, filmato, numero di telefono.
4. Spiegare perché il numero di telefono 0383123456 viene normalmente trattato come testo e non come quantità numerica.
5. Trasformare ciascun dato in un'informazione completa aggiungendo il contesto necessario.
 - a) 12,5
 - b) 45
 - c) 15/03/2026
 - d) vero
6. Indicare almeno quattro metadati che possono accompagnare una fotografia digitale.
7. Un archivio contiene cento misurazioni, ma per venti di esse non è indicata l'unità di misura. Quale caratteristica della qualità dei dati risulta compromessa? Spiegare la risposta.
8. Stabilire se le seguenti affermazioni sono vere o false. Correggere quelle false.
 - a) Un dato possiede sempre un significato completo anche senza contesto.
 - b) Uno stesso contenuto può essere interpretato in modi differenti.
 - c) I metadati descrivono altri dati.

Esercizi: sezione 1.3 (continua)

- d) Un programma corretto produce necessariamente risultati corretti anche quando riceve dati errati.
 - e) Una sequenza di cifre viene sempre trattata come numero.
9. Descrivere un esempio in cui dati incompleti o non aggiornati potrebbero condurre a una decisione sbagliata.

1.4 L'hardware del computer

Il termine **hardware** indica l'insieme delle parti fisiche di un sistema informatico: componenti elettronici, circuiti, cavi, connettori, dispositivi meccanici e periferiche. Tutto ciò che può essere osservato o toccato appartiene quindi all'hardware.

Definizione: hardware

L'**hardware** è l'insieme dei componenti materiali che permettono a un computer di acquisire dati, elaborarli, memorizzarli e comunicare i risultati.

L'hardware comprende sia i componenti collocati all'interno del case, come processore, memoria RAM e unità di archiviazione, sia i dispositivi esterni collegati al computer, come tastiera, mouse, monitor e stampante. La distinzione tra componenti interni ed esterni riguarda la loro posizione; la distinzione tra dispositivi di input e output riguarda invece la funzione svolta nello scambio dei dati.

1.4.1 Dispositivi di input, output e input/output

Un computer deve poter comunicare con l'esterno. I dispositivi utilizzati per acquisire dati o restituire risultati vengono chiamati **periferiche**. In base alla direzione prevalente del flusso dei dati, possono essere classificati come dispositivi di input, di output oppure di input/output.

Definizione: dispositivo di input

Un **dispositivo di input** trasferisce dati o comandi dall'esterno verso il computer.

Sono dispositivi di input, per esempio:

- la **tastiera**, che acquisisce caratteri e comandi;
- il **mouse** e il touchpad, che rilevano movimenti e selezioni;
- il **microfono**, che trasforma i suoni in dati digitali;
- la **fotocamera** e lo scanner, che acquisiscono immagini;
- i **sensori**, che misurano grandezze fisiche come temperatura, pressione o luminosità.

Definizione: dispositivo di output

Un **dispositivo di output** riceve dati dal computer e li trasforma in una forma comprensibile all'utente oppure in un'azione sull'ambiente.

Sono dispositivi di output, per esempio:

- il **monitor**, che visualizza testi, immagini e interfacce;
- la **stampante**, che produce una copia su carta;
- gli **altoparlanti** e le cuffie, che riproducono suoni;
- un **proiettore**, che visualizza immagini su una superficie;
- un attuatore o un motore comandato dal computer, che produce un'azione fisica.

Alcuni dispositivi scambiano dati in entrambe le direzioni e vengono quindi considerati dispositivi di **input/output**. Un touchscreen mostra immagini ma rileva anche il tocco; una scheda di rete invia e riceve dati; un'unità di memoria può essere letta e scritta dal computer.

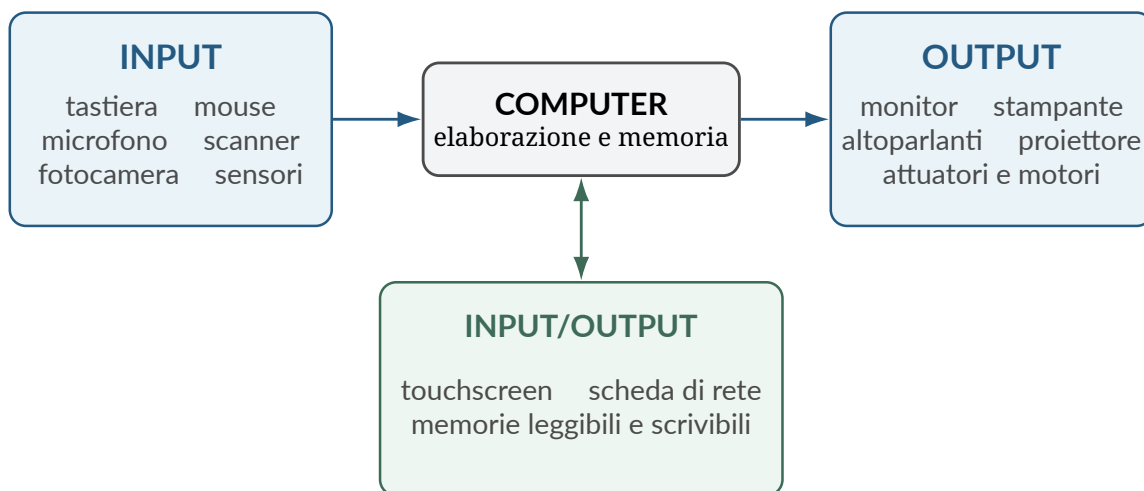


Figura 1.4: La classificazione delle periferiche dipende dalla direzione con cui i dati vengono scambiati con il computer.

Errore comune: interno ed esterno non significano input e output

Un componente interno non è necessariamente dedicato all'elaborazione, così come un dispositivo esterno non è necessariamente di input. Un disco interno svolge una funzione di memorizzazione e scambia dati in entrambe le direzioni; un monitor esterno è invece un dispositivo di output.

1.4.2 I componenti interni

Nei computer desktop, i principali componenti sono montati all'interno del **case**, una struttura che li protegge, li mantiene ordinati e favorisce il passaggio dell'aria. Nei notebook e negli altri dispositivi compatti gli stessi elementi sono presenti, ma possono avere forme diverse ed essere integrati maggiormente tra loro.



Figura 1.5: Il case contiene, protegge e organizza i componenti interni di un computer desktop.

La scheda madre

La **scheda madre**, o *motherboard*, è la scheda elettronica principale del computer. Ospita o collega il processore, la memoria RAM, le unità di archiviazione, le schede di espansione e numerosi connettori. Attraverso piste elettriche e canali di comunicazione chiamati **bus**, permette ai componenti di scambiarsi dati e segnali di controllo.

La scheda madre non esegue da sola tutti i calcoli, ma costituisce la struttura di collegamento su cui il sistema viene organizzato. La disposizione e il tipo dei connettori determinano quali componenti possono essere installati e aggiornati.



Figura 1.6: Una scheda madre vista dall'alto, con socket della CPU, slot della RAM e connettori per le unità di archiviazione.

La CPU

La **CPU**, acronimo di *Central Processing Unit*, è il processore principale del computer. Legge ed esegue le istruzioni dei programmi e coordina le attività necessarie per elaborare i dati. Per comprenderne il funzionamento è utile distinguere due ruoli fondamentali:

- la **Control Unit** o **unità di controllo** preleva le istruzioni dalla memoria, le interpreta e coordina il lavoro delle altre parti del computer, inviando i segnali necessari;

- l'**ALU**, acronimo di *Arithmetic Logic Unit*, esegue operazioni aritmetiche, confronti e operazioni logiche sui dati.

Il lavoro della CPU può essere descritto, in modo semplificato, attraverso un ciclo ripetuto continuamente: l'unità di controllo **preleva** un'istruzione dalla memoria, la **decodifica** e ne guida l'**esecuzione**; quando sono richiesti calcoli o confronti interviene l'ALU.

I processori moderni contengono normalmente più **core**, cioè più unità capaci di eseguire istruzioni, e piccole memorie molto veloci chiamate **cache**, utilizzate per conservare temporaneamente dati e istruzioni di uso frequente.

In molti computer il processore comprende anche una **GPU integrata**. In questo caso lo stesso chip che ospita la CPU contiene un'unità specializzata nella produzione delle immagini da inviare al monitor. Questa soluzione è adatta alle attività quotidiane e riduce ingombro e consumo energetico; per elaborazioni grafiche molto impegnative può invece essere utilizzata una scheda video dedicata.

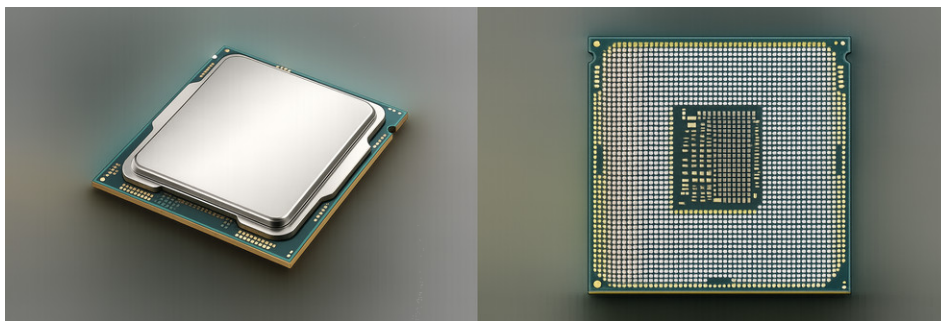


Figura 1.7: Una CPU per computer desktop osservata dal lato superiore e dal lato inferiore. I contatti elettrici collegano il processore al socket della scheda madre.

Approfondimento: il raffreddamento del processore

Durante il funzionamento la CPU produce calore. Un **dissipatore** assorbe e disperde questo calore attraverso una superficie metallica; spesso è accompagnato da una ventola. Tra processore e dissipatore viene applicato un sottile strato di pasta termica, che migliora il trasferimento del calore. Un raffreddamento insufficiente può causare una riduzione automatica delle prestazioni, instabilità o spegnimenti di sicurezza.



Figura 1.8: Un sistema di raffreddamento ad aria per la CPU, formato da dissipatore metallico e ventola.

La memoria RAM

La **RAM**, acronimo di *Random Access Memory*, è la memoria di lavoro del computer. Contiene temporaneamente le istruzioni e i dati che la CPU deve utilizzare durante l'esecuzione dei programmi.

La RAM è molto più veloce delle comuni memorie di massa, ma è **volatile**: il suo contenuto viene perso quando il computer viene spento o non riceve più alimentazione. Una maggiore quantità di RAM consente di mantenere attivi più programmi e dati contemporaneamente, ma non sostituisce la memoria permanente.

Esempio: RAM e piano di lavoro

La RAM può essere paragonata al piano di un banco da lavoro. Gli strumenti e i materiali necessari nell'immediato vengono disposti sul banco per essere raggiunti rapidamente; ciò che non serve subito viene conservato in un armadio. Un piano più ampio permette di lavorare con più elementi contemporaneamente, ma non è un archivio permanente.

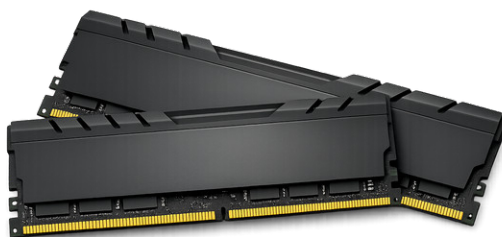


Figura 1.9: Un modulo RAM installabile negli appositi slot della scheda madre.

Le memorie di massa

Le **memorie di massa** conservano in modo permanente il sistema operativo, i programmi e i file. A differenza della RAM, mantengono i dati anche quando il computer è spento. Le tecnologie più comuni nei personal computer sono HDD e SSD.

HDD

Un **HDD**, o disco rigido, memorizza i dati magneticamente su uno o più dischi rotanti. Una testina si sposta sulla superficie per leggere e scrivere le informazioni. Gli HDD offrono capacità elevate a costi contenuti, ma contengono parti meccaniche, sono più lenti degli SSD e risultano più sensibili agli urti durante il funzionamento.



Figura 1.10: Un disco rigido da computer desktop, basato su piatti magnetici e componenti meccanici.

SSD

Un **SSD**, o unità a stato solido, utilizza una tecnologia elettronica basata su celle di memoria flash. Non contiene dischi, testine o altre parti meccaniche in movimento. È generalmente più veloce, silenzioso e resistente agli urti rispetto a un HDD, ma a parità di capacità può avere un costo maggiore.



Figura 1.11: Un'unità SSD nel formato da 2,5 pollici, basata su memoria elettronica flash.

NVMe

Un SSD NVMe utilizza una tecnologia elettronica basata su memoria flash, come gli altri SSD. **NVMe**, acronimo di *Non-Volatile Memory Express*, non indica quindi una memoria completamente diversa, ma un protocollo progettato per comunicare rapidamente con il computer attraverso il collegamento PCI Express.

Molti SSD NVMe utilizzano il formato **M.2**, una piccola scheda allungata che viene inserita direttamente in uno slot della scheda madre. Il nome M.2 non è un acronimo: indica soprattutto la forma e le dimensioni fisiche dell'unità e, da solo, non descrive la tecnologia di memoria o la velocità del dispositivo.

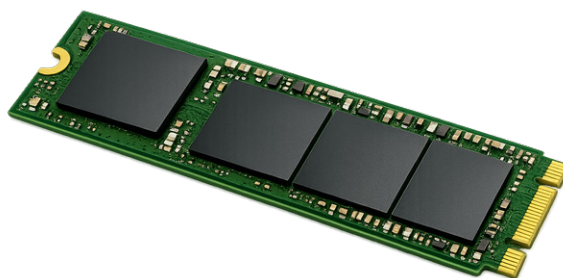


Figura 1.12: Un SSD elettronico nel formato M.2 che utilizza il protocollo NVMe e il collegamento PCI Express.

Tabella 1.2: Confronto generale tra HDD, SSD e SSD NVMe in formato M.2.

Caratteristica	HDD	SSD da 2,5 pollici	SSD NVMe M.2
Tecnologia	Dischi magnetici e parti meccaniche	Memoria elettronica flash	Memoria elettronica flash
Collegamento	Cavo dati e alimentazione	Cavo dati e alimentazione	Inserimento diretto nello slot M.2
Velocità	Inferiore	Elevata	Generalmente molto elevata
Rumore	Può produrre rumore e vibrazioni	Silenzioso	Silenzioso
Resistenza agli urti	Più ridotta durante il funzionamento	Maggiore	Maggiore
Impiego tipico	Archivi di grande capacità	Computer generici e aggiornamenti	Computer recenti e attività ad alte prestazioni

La GPU

La **GPU**, acronimo di *Graphics Processing Unit*, è un processore specializzato nell'elaborazione grafica e nell'esecuzione di numerose operazioni in parallelo. Genera le immagini da inviare al monitor e accelera attività come grafica tridimensionale, montaggio video, elaborazione scientifica e alcuni calcoli usati nell'intelligenza artificiale.

Una GPU può essere:

- **integrata**, quando è incorporata nel processore o nello stesso chip e utilizza generalmente una parte della memoria di sistema;
- **dedicata**, quando è montata su una scheda separata e dispone normalmente di memoria video propria.

Una GPU dedicata offre in genere prestazioni maggiori, ma aumenta costo, consumo energetico e produzione di calore. Per attività comuni, una GPU integrata moderna può essere sufficiente.



Figura 1.13: Una scheda video dedicata con GPU, memoria video, sistema di raffreddamento e connettore PCI Express.

L'alimentatore

L'**alimentatore**, indicato anche con la sigla PSU da *Power Supply Unit*, riceve la corrente alternata della rete elettrica e la converte nelle tensioni continue necessarie ai componenti del computer. Distribuisce energia alla scheda madre, al processore, alle memorie, alle unità di archiviazione e alle eventuali schede di espansione.

La potenza dell'alimentatore viene espressa in watt. Un modello adeguato deve fornire energia sufficiente e stabile, con un margine compatibile con i componenti installati. Una potenza nominale molto elevata non rende automaticamente più veloce il computer.

Errore comune: l'alimentatore non deve essere aperto

L'alimentatore può contenere componenti capaci di conservare cariche elettriche pericolose anche dopo aver scollegato il computer dalla presa. Per questo motivo non deve essere aperto né smontato internamente: eventuali controlli o riparazioni devono essere affidati a personale qualificato.



Figura 1.14: Un alimentatore per computer desktop, con ventola e cavi destinati ai diversi componenti.

1.4.3 Porte e connessioni del computer

Le **porte di comunicazione** permettono al computer di collegarsi a periferiche, monitor, dispositivi audio, reti e unità esterne. Alcuni collegamenti trasportano soltanto dati, altri possono trasferire anche segnali audio e video oppure fornire energia elettrica.

È utile distinguere tre concetti:

- il **connettore** è la forma fisica della presa e della spina;
- l'**interfaccia** stabilisce il modo in cui i dispositivi comunicano;
- il **protocollo** definisce le regole con cui le informazioni vengono scambiate.

La forma della porta, quindi, non è sempre sufficiente per conoscere tutte le funzioni disponibili. Due connettori apparentemente uguali possono supportare velocità, alimentazione o segnali video differenti.

Porte USB

La sigla **USB** deriva da *Universal Serial Bus*. Lo standard USB è stato progettato per collegare numerosi tipi di periferiche attraverso un'interfaccia comune. Tastiere, mouse, stampanti, fotocamere, memorie USB, smartphone e unità SSD esterne sono esempi di dispositivi che possono utilizzare un collegamento USB.

Le porte USB possono trasferire dati e fornire una quantità limitata di energia al dispositivo collegato. Nel corso del tempo sono state introdotte diverse versioni dello standard, con velocità e capacità di alimentazione differenti. Per riconoscere le prestazioni effettive non basta quindi osservare soltanto la forma della presa: occorre consultare le specifiche del computer, del dispositivo e del cavo.

Il connettore rettangolare più diffuso sui computer tradizionali viene comunemente chiamato **USB Type-A**. Esistono anche connettori più piccoli, come Micro-USB, ancora presenti su alcuni dispositivi meno recenti.

USB-C

USB-C, o USB Type-C, indica un connettore compatto e reversibile: può essere inserito in entrambi i versi. È utilizzato su notebook, smartphone, tablet, monitor, docking station e numerose periferiche moderne.

Una porta USB-C può essere impiegata per:

- trasferire dati;
- ricaricare un dispositivo o alimentare un notebook;
- collegare monitor e proiettori, quando è supportata l'uscita video;
- collegare adattatori e docking station che aggiungono altre porte.

Queste funzioni non sono però presenti automaticamente su tutte le porte USB-C. Il connettore descrive soprattutto la forma fisica; le capacità dipendono dagli standard e dalle modalità supportate dal dispositivo e dal cavo.

Approfondimento: USB-C come connettore comune nell'Unione europea

L'Unione europea ha scelto USB-C come porta comune per la ricarica via cavo di molte categorie di dispositivi elettronici portatili venduti nel mercato europeo. Le regole si applicano dal 28 dicembre 2024, tra gli altri, a smartphone, tablet, fotocamere digitali, cuffie, tastiere e mouse, e dal 28 aprile 2026 anche ai computer portatili.

L'obiettivo è permettere di riutilizzare lo stesso caricatore con dispositivi di marche differenti, ridurre la quantità di caricabatterie inutilizzati e di rifiuti elettronici, semplificare gli acquisti e rendere più trasparenti le caratteristiche di ricarica. La presenza della porta USB-C non significa comunque che tutti i dispositivi richiedano la stessa potenza: caricatore, cavo e apparecchio devono supportare le caratteristiche necessarie.

Errore comune: tutte le porte USB-C non sono equivalenti

Due porte USB-C possono avere lo stesso aspetto ma offrire funzioni diverse. Una può consentire soltanto ricarica e trasferimento dati, mentre un'altra può supportare anche un monitor esterno o una docking station. Anche il cavo deve essere compatibile con la funzione richiesta.

**USB Type-A****USB-C**

Figura 1.15: Esempi di connettori USB. USB-C è reversibile, ma le funzioni disponibili dipendono dalle caratteristiche della porta e del cavo.

HDMI e DisplayPort

Le interfacce **HDMI** e **DisplayPort** vengono utilizzate principalmente per collegare monitor, televisori e proiettori. Trasportano un segnale video digitale e possono trasmettere anche l'audio.

- **HDMI** è molto comune su televisori, console, notebook, proiettori e dispositivi multimediali.
- **DisplayPort** è frequente sui computer desktop, sulle schede video e sui monitor. È particolarmente diffuso nelle postazioni informatiche e può essere disponibile anche attraverso una porta USB-C compatibile.

La risoluzione, la frequenza di aggiornamento e le altre caratteristiche supportate dipendono dalla versione dell'interfaccia e dalle capacità di computer, cavo e schermo. Un adattatore può rendere compatibili connettori diversi, ma non può creare una funzione che il dispositivo di origine non supporta.

**HDMI****DisplayPort**

Figura 1.16: Porte digitali utilizzate per trasmettere immagini e, normalmente, anche audio verso monitor, televisori e proiettori.

Collegamenti audio

I computer possono utilizzare differenti collegamenti per il suono. Il connettore analogico più riconoscibile è il **jack da 3,5 mm**, impiegato per cuffie, altoparlanti e microfoni. Nei computer desktop possono essere presenti prese separate, spesso contraddistinte da colori o simboli; su notebook e smartphone è comune una presa combinata per cuffie e microfono.

L'audio può essere trasmesso anche in forma digitale attraverso USB, HDMI o DisplayPort. Una cuffia USB, per esempio, contiene componenti che convertono direttamente il segnale digitale in suono.

Ethernet

La porta **Ethernet** permette di collegare il computer a una rete locale mediante un cavo. Nei personal computer utilizza normalmente un connettore chiamato **RJ45**, più largo di quello telefonico e dotato di una linguetta di blocco.

Un cavo Ethernet può collegare il computer a uno switch, a un router oppure a una presa di rete dell'edificio. Il collegamento via cavo offre spesso elevata stabilità, ridotta sensibilità alle interferenze e prestazioni prevedibili. La velocità effettiva dipende comunque dalle porte, dal cavo e dagli apparati di rete utilizzati.

**Porte audio****Ethernet RJ45**

Figura 1.17: Esempi di collegamenti dedicati all'audio analogico e alla rete cablata.

Porte di alimentazione

Il collegamento di alimentazione fornisce l'energia necessaria al funzionamento o alla ricarica del dispositivo. I computer desktop ricevono normalmente corrente attraverso la presa dell'alimentatore; i notebook possono utilizzare un connettore dedicato, un connettore cilindrico oppure una porta USB-C compatibile con la ricarica.

Un connettore fisicamente compatibile non garantisce che tensione e potenza siano adeguate. È quindi necessario utilizzare alimentatori conformi alle specifiche del dispositivo. Un caricatore con potenza insufficiente può ricaricare molto lentamente o non riuscire ad alimentare il computer durante l'uso.



Figura 1.18: Esempio di collegamento utilizzato per alimentare o ricaricare un computer.

Approfondimento: una porta può svolgere più funzioni

Le porte moderne tendono a riunire più funzioni nello stesso collegamento. Una sola porta USB-C, quando adeguatamente supportata, può trasferire dati, inviare immagini a un monitor e alimentare il computer. Questa versatilità riduce il numero di connettori, ma rende più importante controllare simboli e specifiche tecniche prima di scegliere cavi e adattatori.

1.4.4 Il modello di von Neumann

Per comprendere come i componenti collaborano è utile fare riferimento al **modello di von Neumann**, una rappresentazione concettuale dell'architettura dei computer formulata nel periodo in cui venivano progettati i primi elaboratori elettronici programmabili.

Approfondimento: John von Neumann

John von Neumann (1903–1957) fu un matematico di origine ungherese naturalizzato statunitense. Contribuì a numerosi settori, tra cui matematica, fisica, teoria dei giochi e progettazione dei primi calcolatori elettronici. Nel 1945 partecipò alla descrizione dell'EDVAC e contribuì a rendere celebre l'idea del **programma memorizzato**, secondo cui dati e istruzioni possono essere conservati nella stessa memoria. L'architettura porta il suo nome, anche se nacque dal lavoro di più ricercatori e ingegneri.

Il modello distingue quattro elementi fondamentali:

- l'**unità di elaborazione**, cioè la CPU;
- la **memoria centrale**, che contiene dati e istruzioni;
- i **dispositivi di input**;
- i **dispositivi di output**.

Questi elementi comunicano attraverso i **bus**, cioè insiemi di collegamenti utilizzati per trasferire dati, indirizzi e segnali di controllo.

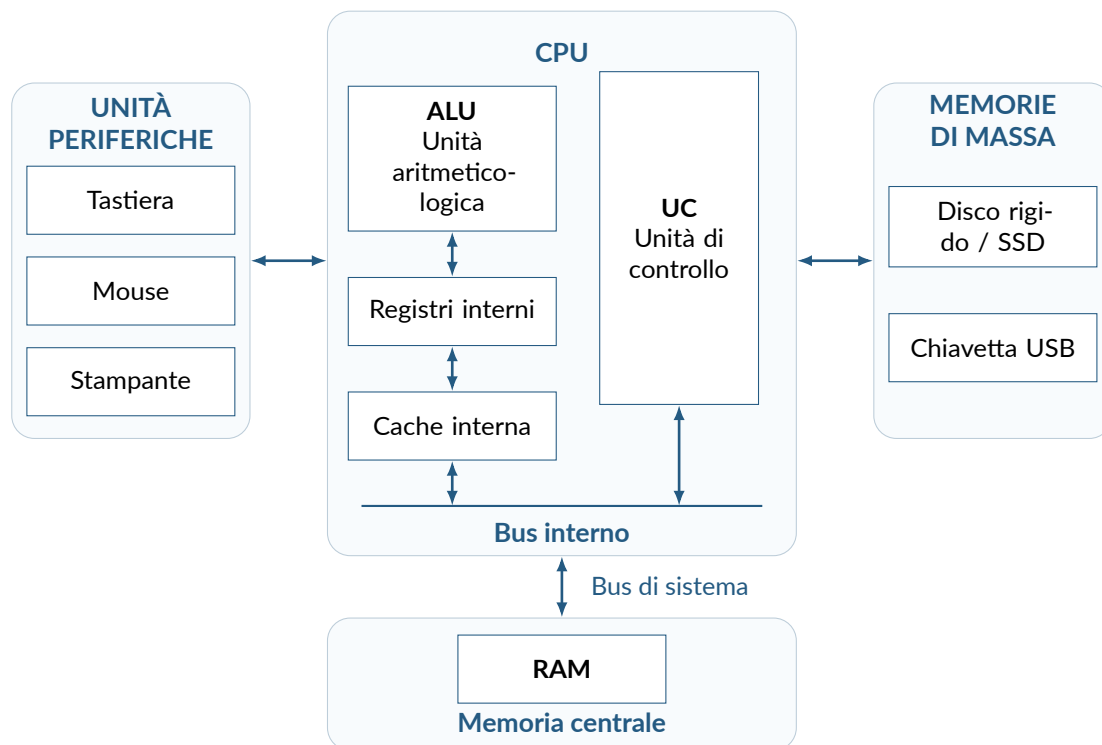


Figura 1.19: Schema semplificato del modello di von Neumann. Le frecce evidenziano gli scambi tra periferiche, CPU, memoria centrale e memorie di massa; all'interno della CPU sono rappresentati i principali blocchi funzionali.

L'idea più importante del modello è il principio del **programma memorizzato**: le istruzioni non sono fissate soltanto nei collegamenti fisici della macchina, ma vengono conservate in memoria insieme ai dati. Lo stesso hardware può quindi svolgere compiti differenti caricando istruzioni differenti.

Il modello di von Neumann è importante perché permette di descrivere in modo semplice il funzionamento di gran parte dei computer per uso generale. Le tecnologie moderne sono molto più complesse, ma continuano a utilizzare CPU, memoria e dispositivi di input/output che collaborano secondo principi riconducibili a questa architettura.

Approfondimento: il collo di bottiglia di von Neumann

La CPU può elaborare dati molto rapidamente, ma deve riceverli dalla memoria attraverso collegamenti con capacità limitata. Quando il trasferimento non è abbastanza veloce, il processore rimane in attesa. Questo limite viene chiamato **collo di bottiglia di von Neumann**. Memorie cache, bus più rapidi e tecniche di esecuzione avanzate cercano di ridurne gli effetti.

1.4.5 Dai computer tradizionali ai sistemi moderni

Nei computer desktop tradizionali i componenti principali sono facilmente riconoscibili e spesso sostituibili. Nei dispositivi moderni, soprattutto notebook sottili, tablet e smartphone, numerose funzioni possono essere integrate in un unico circuito chiamato **System on a Chip** o SoC.

Un SoC può riunire CPU, GPU, controller della memoria, funzioni di comunicazione e altri elementi. Questa integrazione riduce ingombro e consumo energetico, ma può rendere più difficile la riparazione o l'aggiornamento. Anche la RAM e la memoria di massa possono essere saldate direttamente sulla scheda e non essere sostituibili.

I computer recenti utilizzano frequentemente processori multicore, SSD NVMe, collegamenti ad alta velocità e sistemi di gestione energetica. Queste innovazioni non annullano il modello generale studiato: rendono più stretta e complessa la collaborazione tra elaborazione, memoria, archiviazione e dispositivi di comunicazione.

Curiosità: un computer moderno può essere molto piccolo

Un microcontrollore grande pochi centimetri può contenere processore, memoria e interfacce di input/output. Può controllare un sensore, un piccolo robot, un sistema di illuminazione o uno strumento di misura consumando pochissima energia.

Esercizi: sezione 1.4

1. Classificare i seguenti dispositivi come input, output oppure input/output: tastiera, monitor, touchscreen, stampante, scheda di rete, microfono, memoria USB, cuffie.
2. Spiegare perché la distinzione tra componente interno ed esterno non coincide con quella tra input e output.
3. Associare ogni componente alla funzione principale: CPU, RAM, HDD, SSD, GPU, alimentatore, scheda madre.
4. Spiegare le funzioni dell'unità di controllo e dell'ALU e descrivere il ciclo di prelievo, decodifica ed esecuzione.
5. Spiegare la differenza tra RAM e memoria di massa, indicando che cosa accade ai dati quando viene interrotta l'alimentazione.
6. Completare una tabella di confronto tra HDD e SSD considerando velocità, presenza di parti meccaniche, rumore e resistenza agli urti.
7. Spiegare la differenza tra una memoria SSD, il formato M.2 e il protocollo NVMe.
8. Spiegare la differenza tra GPU integrata e GPU dedicata e indicare un possibile vantaggio di ciascuna soluzione.
9. Distinguere USB da USB-C, spiegando perché i due termini non indicano esattamente la stessa cosa.
10. Indicare la porta più adatta nei seguenti casi: monitor esterno, cuffie analogiche, rete cablata, mouse USB, unità SSD esterna e alimentazione di un notebook.
11. Spiegare perché due porte USB-C identiche nell'aspetto possono offrire funzioni differenti.

Esercizi: sezione 1.4 (continua)

12. Stabilire se le seguenti affermazioni sono vere o false. Correggere quelle false.
 - a) La RAM conserva i dati anche quando il computer è spento.
 - b) NVMe è un protocollo usato da molte unità SSD.
 - c) L'ALU esegue operazioni aritmetiche e logiche, mentre l'unità di controllo coordina l'esecuzione delle istruzioni.
 - d) L'alimentatore trasforma e distribuisce l'energia necessaria ai componenti.
 - e) Una GPU dedicata dispone normalmente di memoria video propria.
 - f) Nel modello di von Neumann dati e istruzioni possono essere conservati nella memoria.
13. Descrivere il percorso seguito dai dati quando viene aperta una fotografia memorizzata su un SSD e visualizzata sul monitor, citando almeno memoria di massa, RAM, CPU o GPU e dispositivo di output.
14. Spiegare perché il principio del programma memorizzato ha reso possibile utilizzare lo stesso computer per attività differenti.

1.5 Il software

L'hardware, da solo, non è sufficiente per rendere utile un computer. I componenti fisici devono ricevere istruzioni che stabiliscano quali operazioni eseguire, come gestire i dati e in che modo comunicare con l'utente. L'insieme di programmi, dati e procedure utilizzati da un sistema informatico prende il nome di **software**.

Definizione: software

Il **software** è la parte immateriale di un sistema informatico. Comprende programmi, istruzioni, dati di configurazione e altri elementi necessari per controllare l'hardware e svolgere attività specifiche.

Hardware e software sono quindi complementari. Un programma di videoscrittura non può funzionare senza processore, memoria e dispositivi di input/output; allo stesso tempo, un computer privo di software non saprebbe quali operazioni compiere.

1.5.1 Software di sistema e software applicativo

Una prima classificazione del software riguarda la **funzione** svolta. Si distinguono principalmente il software di sistema e il software applicativo.

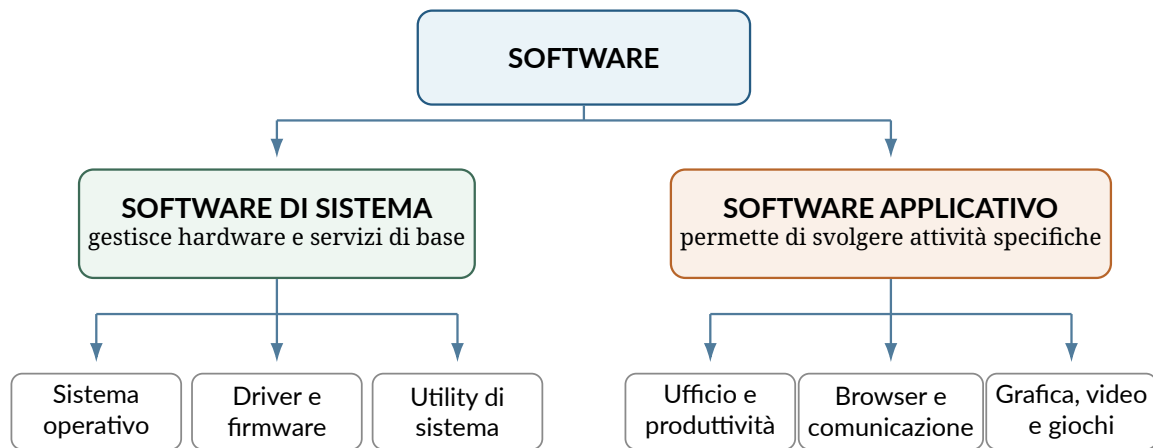


Figura 1.20: La classificazione del software in base alla funzione svolta.

Il software di sistema

Il **software di sistema** comprende i programmi che controllano e coordinano le risorse fondamentali del computer. Fa da collegamento tra l'hardware, l'utente e il software applicativo, permettendo alle diverse componenti di lavorare insieme.

Appartengono al software di sistema:

- il **sistema operativo**;
- i **driver**, che permettono al sistema operativo di comunicare con specifici dispositivi hardware;
- le **utility di sistema**, come gli strumenti di backup, gestione dei dischi, aggiornamento e diagnostica;
- il **firmware**, cioè software memorizzato stabilmente in un dispositivo e destinato a controllarne le funzioni di base.

Errore comune: software di sistema e sistema operativo non sono sinonimi

Il sistema operativo è il principale software di sistema, ma non è l'unico. Driver, firmware e numerosi strumenti di manutenzione appartengono alla stessa categoria pur svolgendo funzioni differenti.

Il sistema operativo

Il **sistema operativo**, spesso indicato con la sigla OS dall'inglese *Operating System*, è l'insieme coordinato di programmi che gestisce le risorse del computer e fornisce i servizi di base necessari alle applicazioni.

Dal punto di vista dell'utente, tre funzioni sono particolarmente importanti:

- **organizzare file e cartelle**, permettendo di salvare, trovare, copiare, spostare e cancellare i dati nelle unità di archiviazione;
- **offrire un'interfaccia**, grafica o testuale, attraverso cui impartire comandi e utilizzare il computer;

- **permettere di installare ed eseguire altri programmi**, mettendo a disposizione delle applicazioni le risorse e i servizi necessari.

Per rendere possibili queste attività, il sistema operativo svolge anche numerose funzioni tecniche di supporto: gestisce l'utilizzo della CPU e della RAM, controlla periferiche e dispositivi tramite i driver, amministra utenti e autorizzazioni, applica meccanismi di sicurezza e coordina la comunicazione in rete e gli altri servizi di sistema.

Windows, macOS e le distribuzioni GNU/Linux, come Ubuntu, sono esempi di sistemi operativi per computer. Android e iOS svolgono la stessa funzione su smartphone e tablet.

Esempio: apertura di un documento

Quando viene aperto un documento, il sistema operativo individua il file nell'unità di archiviazione, carica in memoria l'applicazione adatta, assegna le risorse necessarie e permette al programma di utilizzare monitor, tastiera e mouse. L'applicazione svolge il compito richiesto, ma si appoggia continuamente ai servizi del sistema operativo.

Il software applicativo

Il **software applicativo** comprende i programmi progettati per consentire all'utente di svolgere attività specifiche. Le applicazioni utilizzano i servizi messi a disposizione dal sistema operativo e non controllano normalmente l'hardware in modo diretto.

Sono esempi di software applicativo:

- programmi di videoscrittura, fogli elettronici e software per presentazioni;
- browser Web e programmi di posta elettronica;
- applicazioni per grafica, fotoritocco, montaggio video e riproduzione multimediale;
- videogiochi;
- programmi gestionali, scientifici e didattici;
- applicazioni per strumenti di laboratorio, cartografia, progettazione e analisi dei dati.

Microsoft Word e LibreOffice Writer svolgono entrambi funzioni di videoscrittura; Adobe Photoshop e GIMP permettono di elaborare immagini; Microsoft Edge, Google Chrome e Mozilla Firefox sono browser. Appartengono tutti al software applicativo, anche se possiedono licenze e modelli economici differenti.

Approfondimento: applicazioni installate e applicazioni Web

Un'applicazione può essere installata nel dispositivo oppure utilizzata attraverso un browser. Nel secondo caso una parte importante dell'elaborazione e della memorizzazione può avvenire su server remoti. Le applicazioni Web favoriscono collaborazione e accesso da dispositivi differenti, ma possono dipendere dalla connessione Internet e dalle condizioni del servizio offerto dal fornitore.

1.5.2 Software open source e software proprietario

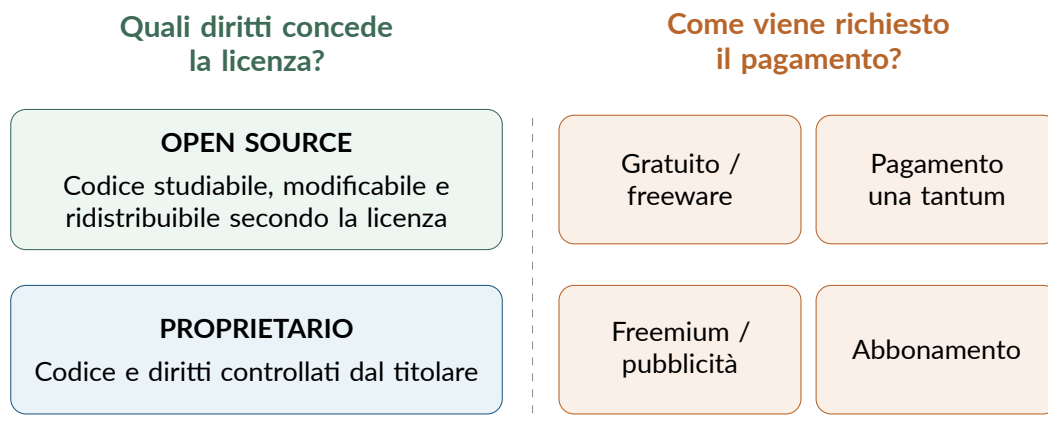
Una seconda classificazione riguarda la **licenza**, cioè l'insieme delle condizioni giuridiche che stabiliscono come il software può essere utilizzato, copiato, studiato, modificato e distribuito. Questa distinzione è indipendente dalla funzione: sia un sistema operativo sia un'applicazione possono essere open source oppure proprietari.

Definizione: EULA

La sigla **EULA** deriva da *End User License Agreement*, cioè **contratto di licenza con l'utente finale**. È il documento che stabilisce le condizioni con cui l'utente può installare e utilizzare un software. Può specificare, per esempio, il numero di dispositivi consentiti, i limiti alla copia o alla modifica, la durata della licenza, le modalità di aggiornamento e le responsabilità delle parti. L'accettazione viene spesso richiesta durante l'installazione o al primo avvio del programma.

L'EULA non descrive necessariamente il prezzo del programma e non coincide con l'informativa sulla privacy: la prima regola l'uso del software, mentre la seconda spiega come vengono trattati i dati personali.

DUE DOMANDE DIFFERENTI



Licenza e prezzo non coincidono: le combinazioni possibili sono numerose.

Figura 1.21: Licenza e prezzo sono due caratteristiche differenti. Un software open source può essere venduto e un software proprietario può essere distribuito gratuitamente.

Il software open source

Nel **software open source** il codice sorgente è reso disponibile secondo una licenza che consente di studiarlo e, rispettando determinate condizioni, di modificarlo e ridistribuirlo. La semplice possibilità di vedere una parte del codice non è sufficiente: sono i diritti concessi dalla licenza a determinare se il software è realmente open source.

Tra i possibili vantaggi vi sono:

- maggiore possibilità di controllo e verifica del funzionamento;
- possibilità di adattare il programma a esigenze specifiche;
- collaborazione tra sviluppatori, organizzazioni e comunità;

- minore dipendenza da un singolo produttore;
- possibilità di continuare lo sviluppo attraverso nuovi progetti derivati, quando la licenza lo consente.

LibreOffice, Mozilla Firefox, GIMP, Blender e numerose distribuzioni GNU/Linux sono esempi di software open source. Ciò non significa che ogni servizio collegato sia necessariamente gratuito o che chiunque possa utilizzare senza limiti nomi, marchi e loghi del progetto.

Il software proprietario

Nel **software proprietario** il titolare conserva il controllo esclusivo sul codice sorgente e concede agli utenti soltanto i diritti previsti dalla licenza. In genere non è possibile modificare il programma, distribuirne copie liberamente o accedere al codice sorgente completo.

Windows, macOS, Microsoft Word, Adobe Photoshop e AutoCAD sono esempi di software proprietario. Il produttore può stabilire su quanti dispositivi installare il programma, per quanto tempo utilizzarlo, quali funzioni includere e quali condizioni applicare agli aggiornamenti.

Il software proprietario può offrire assistenza centralizzata, integrazione con specifici servizi e uno sviluppo coordinato dal produttore. Può però creare maggiore dipendenza da formati, piattaforme e decisioni commerciali del fornitore.

Errore comune: open source non significa semplicemente gratuito

Le parole *open source*, *gratuito* e *commerciale* non sono sinonimi. Un programma open source può essere venduto oppure accompagnato da assistenza a pagamento. Un programma proprietario può invece essere offerto gratuitamente, pur senza concedere il diritto di studiarne o modificarne il codice.

1.5.3 Modelli di distribuzione del software proprietario

Le espressioni utilizzate per descrivere il modello economico non indicano necessariamente una diversa licenza. Freeware, payware, freemium e software in abbonamento possono essere tutti proprietari. Inoltre, le categorie possono sovrapporsi: un'applicazione può essere contemporaneamente proprietaria, freemium e finanziata anche dalla pubblicità.

Tabella 1.3: Principali modelli di distribuzione del software proprietario.

Modello	Caratteristiche	Esempio indicativo
Freeware	Utilizzo gratuito, ma codice e diritti di modifica rimangono normalmente sotto il controllo del produttore.	Adobe Acrobat Reader; molte applicazioni gratuite distribuite dai produttori.
Payware con licenza perpetua	Pagamento una tantum per utilizzare una determinata versione del programma. Gli aggiornamenti principali successivi possono richiedere un nuovo acquisto.	Office 2024; numerosi videogiochi acquistati singolarmente.
Shareware	Programma distribuibile e provabile, ma l'uso continuato o completo richiede il pagamento della licenza.	Modello diffuso storicamente per utility e programmi distribuiti tramite Internet.
Trial o demo	Versione utilizzabile per un periodo limitato oppure con alcune funzioni disattivate.	Prove gratuite di programmi professionali o videogiochi dimostrativi.
Freemium	Versione di base gratuita; funzioni avanzate, spazio aggiuntivo o rimozione di limiti richiedono un pagamento.	Spotify, Dropbox e numerose applicazioni per dispositivi mobili.
Adware	Utilizzo gratuito finanziato dalla visualizzazione di pubblicità; può essere associato ad altri modelli.	Molte applicazioni e giochi gratuiti per smartphone.
Abbonamento	Pagamento periodico mensile o annuale che mantiene attivi utilizzo, aggiornamenti e servizi collegati.	Microsoft 365 e Adobe Creative Cloud.

Approfondimento: freeware e software open source

Il termine *freeware* descrive soprattutto il fatto che il programma può essere usato senza pagare. Non implica normalmente accesso al codice sorgente o libertà di modifica. Firefox e LibreOffice non sono semplicemente freeware: sono software open source distribuiti con licenze che concedono diritti ulteriori.

1.5.4 Dalla licenza perpetua all'abbonamento

Per molti anni il modello più comune per i programmi professionali è stato il **payware con licenza perpetua**: si acquistava una determinata versione e la si poteva utilizzare senza una scadenza prestabilita. Quando veniva pubblicata una nuova versione importante, l'utente decideva se continuare con quella già posseduta oppure acquistare l'aggiornamento.

Oggi molti produttori propongono invece un **abbonamento**. Il software resta utilizzabile finché vengono effettuati i pagamenti periodici e, durante questo periodo, riceve nuove

funzioni, correzioni e aggiornamenti. Microsoft, per esempio, affianca ancora Office 2024 acquistabile una volta sola ai piani Microsoft 365 in abbonamento; Adobe distribuisce le applicazioni professionali della Creative Cloud attraverso piani periodici.

Il passaggio all'abbonamento è stato favorito da diversi fattori:

- i programmi vengono aggiornati continuamente invece di essere sostituiti soltanto da grandi versioni periodiche;
- sicurezza, compatibilità e assistenza richiedono un lavoro costante;
- molte applicazioni includono servizi cloud, archiviazione, sincronizzazione e collaborazione online;
- il produttore ottiene entrate più regolari e prevedibili;
- il costo iniziale per l'utente può essere inferiore rispetto all'acquisto di una licenza professionale completa;
- l'account e l'abbonamento possono permettere l'uso su più dispositivi e l'accesso a servizi aggiuntivi.

Il modello presenta però anche svantaggi. Nel lungo periodo la spesa complessiva può superare quella di una licenza perpetua; interrompendo il pagamento si possono perdere alcune funzioni o la possibilità di utilizzare il programma; inoltre aumenta la dipendenza dal produttore, dall'account e talvolta dalla connessione Internet.

Esempio: licenza perpetua e abbonamento

Con una licenza perpetua viene acquistata una specifica versione del programma: continuerà a funzionare, ma non riceverà necessariamente le nuove funzioni delle versioni future. Con un abbonamento si paga periodicamente e si utilizzano le versioni aggiornate finché il contratto rimane attivo. Nessuno dei due modelli è sempre migliore: la scelta dipende dalla durata prevista, dai servizi necessari, dal numero di dispositivi e dalla frequenza degli aggiornamenti.

Curiosità: Software as a Service

Quando il programma viene fornito principalmente attraverso Internet insieme a infrastruttura, aggiornamenti e archiviazione, si parla spesso di *Software as a Service* o SaaS. In questo modello l'utente acquista soprattutto l'accesso continuativo a un servizio, non il possesso di una copia definitiva del programma.

Esercizi: sezione 1.5

1. Spiegare la differenza tra hardware e software.
2. Distinguere software di sistema, sistema operativo e software applicativo.
3. Indicare a quale categoria appartengono: driver della stampante, Windows, LibreOffice Writer, programma di backup, browser, firmware di un router.
4. Indicare le tre funzioni principali del sistema operativo presentate nel testo e descrivere almeno due funzioni tecniche di supporto.
5. Descrivere che cosa accade, dal punto di vista del software, quando viene aperto

Esercizi: sezione 1.5 (continua)

un documento.

6. Classificare i seguenti programmi come software di sistema o applicativo: macOS, GIMP, driver della scheda video, Microsoft Excel, utility di gestione del disco, Firefox.
7. Spiegare che cos'è un EULA e indicare almeno tre condizioni che può stabilire.
8. Spiegare le principali differenze tra software open source e software proprietario.
9. Stabilire se le seguenti affermazioni sono vere o false. Correggere quelle false.
 - a) Ogni software gratuito è open source.
 - b) Un software open source non può essere venduto.
 - c) Il sistema operativo appartiene al software di sistema.
 - d) Un programma proprietario può essere freeware.
 - e) Il modello freemium offre sempre tutte le funzioni gratuitamente.
 - f) Un abbonamento e una licenza perpetua sono due modalità differenti di pagamento e utilizzo.
10. Associare i termini freeware, payware, trial, freemium, adware e abbonamento alla corretta descrizione.
11. Scegliere tre programmi utilizzati abitualmente e individuare, per ciascuno: funzione, tipo di licenza e modello di distribuzione.
12. Confrontare una licenza perpetua e un abbonamento indicando almeno due vantaggi e due svantaggi di ciascun modello.
13. Un'applicazione offre gratuitamente le funzioni di base, mostra pubblicità e richiede un pagamento mensile per rimuoverla e usare strumenti avanzati. Indicare tutti i modelli di distribuzione che possono descriverla.
14. Spiegare perché la classificazione in software di sistema/applicativo è indipendente dalla classificazione open source/proprietario.
15. Preparare una breve relazione sulla licenza di un software open source, indicando quali diritti concede agli utenti e quali condizioni devono essere rispettate.

Preparazione alla scheda di laboratorio

Durante l'esercitazione pratica saranno fornite dal docente le indicazioni operative, i materiali da utilizzare e le norme di sicurezza da seguire. Gli studenti dovranno raccogliere in modo ordinato le informazioni necessarie per realizzare successivamente una **scheda di laboratorio** oppure una **relazione tecnica**.

Prima dell'attività è opportuno predisporre un foglio o un documento digitale contenente i seguenti campi:

- titolo provvisorio dell'esercitazione;
- data, classe, gruppo e componenti del gruppo;
- spazio per annotare l'obiettivo comunicato dal docente;
- elenco dei materiali e degli strumenti utilizzati;
- spazio per descrivere in ordine le operazioni svolte;
- tabella per registrare i componenti osservati, la loro funzione e i collegamenti individuati;
- spazio per difficoltà, imprevisti e soluzioni adottate;
- elenco delle fotografie o degli schemi realizzati;
- spazio per le osservazioni conclusive.

Traccia minima per la raccolta delle osservazioni

Voce	Appunti raccolti durante l'esercitazione
Obiettivo	
Materiali e strumenti	
Operazioni svolte	
Componenti osservati	
Collegamenti individuati	
Problemi e soluzioni	
Fotografie o schemi	
Osservazioni finali	

Gli appunti non devono essere già scritti nella forma definitiva della relazione: devono essere completi, comprensibili e sufficientemente ordinati da permettere, dopo il laboratorio, di ricostruire ciò che è stato realmente osservato e svolto.

Esercizi riepilogativi

Che cos'è un computer?

Capitolo 1

1. Scegli la risposta corretta

1. Lo stesso computer può svolgere attività differenti soprattutto perché può: a) usare molte periferiche; b) eseguire programmi differenti; c) trattare solo numeri.
2. Quale elemento conserva temporaneamente dati e istruzioni durante il lavoro? a) RAM; b) SSD; c) alimentatore.
3. Un metadato è: a) un dato che descrive un altro dato; b) un programma; c) una periferica.
4. Quale tra i seguenti appartiene al software di sistema? a) sistema operativo; b) elaboratore di testi; c) foglio elettronico.

2. Vero o falso?

	V	F
1. La RAM è una memoria volatile.	☐	☐
2. Un touchscreen può svolgere sia funzioni di input sia di output.	☐	☐
3. USB-C indica sempre la velocità e tutte le funzioni disponibili della porta.	☐	☐
4. Un software di sistema può essere open source oppure proprietario.	☐	☐
5. Nel modello di von Neumann dati e istruzioni possono essere conservati in memoria.	☐	☐

3. Associa ogni componente alla funzione prevalente

Componenti: A. CPU; B. SSD; C. alimentatore; D. GPU; E. scheda madre; F. RAM.

- | | |
|-------------------------------------|---------------------------------------|
| 1. collega i componenti principali | 4. esegue le istruzioni |
| 2. conserva temporaneamente i dati | 5. conserva dati in modo non volatile |
| 3. elabora soprattutto dati grafici | 6. trasforma e distribuisce energia |

Scrivi le coppie corrette:

.....

4. Ricostruisci il percorso dei dati

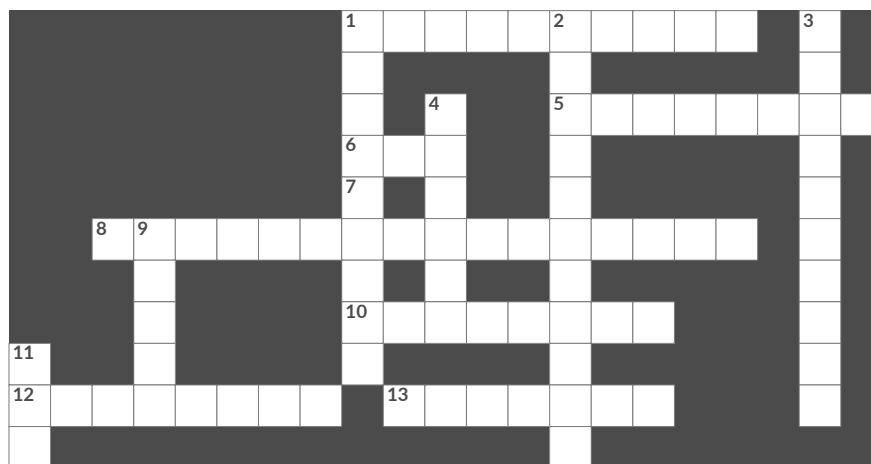
Una fotografia è memorizzata su un SSD e viene aperta sul computer. Descrivi, nell'ordine, il possibile ruolo di **SSD, RAM, CPU o GPU, sistema operativo e monitor**.

Esercizi riepilogativi

Cruciverba di hardware, software e dati

Capitolo 1**Completa lo schema**

Inserisci le parole senza spazi. Le definizioni riprendono i concetti principali del capitolo.

**Orizzontali**

1. Esegue le istruzioni dei programmi. (10)
5. Insieme dei componenti fisici. (8)
6. Componente specializzato nell'elaborazione grafica. (3)
8. Gestisce risorse hardware, programmi e file. (16)
10. Dato che descrive un altro dato. (8)
12. Insieme di programmi e dati. (8)
13. Parte in cui si conservano dati e istruzioni. (7)

Verticali

1. Insieme ordinato di istruzioni. (9)
2. Componente che collega le parti principali. (11)
3. Dispositivo collegato per funzioni di ingresso o uscita. (10)
4. Risultato comunicato dal computer. (6)
7. Memoria veloce e volatile. (3)
9. Dato introdotto nel sistema. (5)
11. Unità di memoria di massa a stato solido. (3)

Sintesi in tre righe

Spiega perché un computer non può funzionare utilmente considerando soltanto l'hardware oppure soltanto il software.

Parte II

Il linguaggio digitale

2. Il linguaggio digitale

Testi, immagini, suoni e programmi appaiono molto diversi tra loro, ma all'interno del computer vengono rappresentati mediante sequenze di simboli digitali. Per comprendere come un dispositivo memorizza ed elabora l'informazione è quindi necessario conoscere le unità con cui essa viene misurata e i sistemi di numerazione utilizzati dall'elaboratore.

Il capitolo introduce bit e byte, le conversioni tra le principali unità di misura informatiche, il sistema binario, le conversioni dei numeri con la virgola e le operazioni aritmetiche eseguite su sequenze di bit. Viene inoltre presentata la rappresentazione dei numeri negativi attraverso il complemento a 2. Le sezioni successive saranno dedicate al sistema esadecimale e ad alcune applicazioni concrete, come i codici dei colori e il calcolo della dimensione teorica di una fotografia digitale.

Conoscenze	Abilità	Competenze
• bit, byte e multipli del byte; • sistema di numerazione posizionale; • sistema binario, potenze di 2 e parte frazionaria; • operazioni con numeri binari; • complemento a 2; • sistema esadecimale e colori RGB; • pixel, risoluzione e profondità di colore.	• convertire b, B, KB, MB, GB e TB; • convertire numeri interi e frazionari tra decimale e binario, approssimando quando necessario; • eseguire operazioni binarie; • usare il complemento a 2 per effettuare sottrazioni; • convertire tra binario, decimale ed esadecimale; • interpretare codici colore e dimensioni di immagini.	• interpretare correttamente capacità di memoria e dimensione dei file; • comprendere come i dati vengono rappresentati in forma digitale; • verificare la correttezza di conversioni e operazioni; • collegare numerazione binaria, colori e immagini digitali; • risolvere problemi concreti relativi allo spazio di archiviazione.

2.1 Misurare l'informazione

La quantità di informazione contenuta in un file e la capacità di una memoria devono essere espresse attraverso unità di misura specifiche. Un documento di testo, una fotografia, un video e un programma possono occupare quantità di spazio molto differenti, ma la loro dimensione viene sempre ricondotta a bit, byte e multipli del byte.

2.1.1 Il bit: l'unità elementare

Il termine **bit** deriva dall'espressione inglese *binary digit*, cioè “cifra binaria”. Un bit può assumere soltanto due valori:

0 oppure 1.

Questi simboli possono rappresentare due stati alternativi, per esempio vero e falso, acceso e spento, presente e assente. All'interno dei circuiti elettronici i due valori vengono associati a condizioni fisiche distinguibili, come differenti intervalli di tensione.

Definizione: bit

Il **bit** è l'unità elementare dell'informazione digitale e può assumere uno dei due valori 0 oppure 1. Il suo simbolo è la lettera minuscola **b**.

Un solo bit permette di distinguere due possibilità. Aumentando il numero di bit aumenta rapidamente anche il numero delle combinazioni rappresentabili:

Numero di bit	Combinazioni possibili	Esempio
1	2	0, 1
2	4	00, 01, 10, 11
3	8	da 000 a 111
4	16	da 0000 a 1111

In generale, con n bit si possono rappresentare 2^n combinazioni differenti. Questo principio sarà importante nello studio dei numeri binari, dei colori e delle immagini digitali.

Se le combinazioni vengono interpretate come numeri binari interi non negativi, a ciascuna di esse corrisponde un valore decimale. Con 3 bit si ottengono tutte le combinazioni comprese tra $(000)_2$ e $(111)_2$, cioè i valori decimali da 0 a 7.

Binario	Decimale
(000) ₂	(0) ₁₀
(001) ₂	(1) ₁₀
(010) ₂	(2) ₁₀
(011) ₂	(3) ₁₀
(100) ₂	(4) ₁₀
(101) ₂	(5) ₁₀
(110) ₂	(6) ₁₀
(111) ₂	(7) ₁₀

Tabella 2.1: Tutte le combinazioni ottenibili con 3 bit e il corrispondente valore decimale.

La prima combinazione rappresenta il valore 0, mentre l'ultima rappresenta il valore 7. In generale, usando n bit per numeri interi non negativi, il valore minimo è 0 e il valore massimo è $2^n - 1$. Il procedimento per convertire un numero binario in decimale verrà spiegato nella sezione 2.2.

2.1.2 Il byte

Per rappresentare informazioni utili è necessario raggruppare più bit. Il raggruppamento fondamentale utilizzato per misurare dati e memorie è il **byte**.

Definizione: byte

Un **byte** è un gruppo formato da 8 bit. Il suo simbolo è la lettera maiuscola **B**.

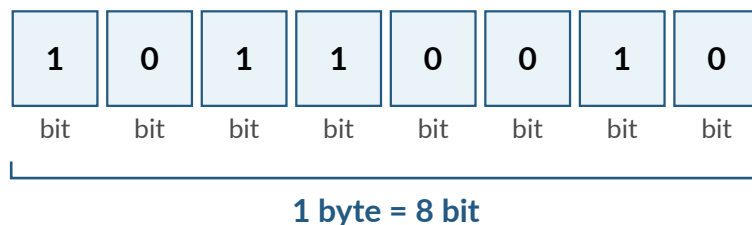


Figura 2.1: Un byte è formato da otto bit. La sequenza mostrata è soltanto una delle 256 combinazioni possibili.

Poiché ogni posizione può contenere 0 oppure 1, un byte può assumere:

$$2^8 = 256$$

configurazioni differenti. A seconda del sistema di codifica adottato, tali combinazioni possono rappresentare numeri, caratteri, parti di un'immagine o altri dati.

È essenziale distinguere i simboli:

- **b** indica il bit;
- **B** indica il byte.

Le due unità non sono equivalenti:

$$1 \text{ B} = 8 \text{ b}.$$

Esempio: conversione tra bit e byte

Per convertire 40 bit in byte si divide per 8:

$$40 \text{ b} \div 8 = 5 \text{ B}.$$

Per convertire 12 byte in bit si moltiplica per 8:

$$12 \text{ B} \times 8 = 96 \text{ b}.$$

Errore comune: confondere b e B

Le scritture 16 b e 16 B indicano quantità diverse: $16 \text{ b} = 2 \text{ B}$, mentre $16 \text{ B} = 128 \text{ b}$. Ignorare la differenza tra la lettera minuscola e quella maiuscola può produrre errori di un fattore 8.

2.1.3 I multipli del byte

Un byte è troppo piccolo per esprimere comodamente la dimensione dei file e la capacità delle memorie moderne. Si utilizzano quindi unità più grandi.

Per le capacità di memoria e le dimensioni dei file, in questo libro adottiamo una **convenzione didattica basata su 1024**: usiamo i simboli **KB**, **MB**, **GB** e **TB** e, a ogni passaggio tra due unità consecutive, il fattore 1024. Le equivalenze della tabella valgono per gli esempi e gli esercizi del libro.

Unità	Simbolo	Equivalenza	In byte
Kilobyte	KB	$1 \text{ KB} = 1024 \text{ B}$	2^{10} B
Megabyte	MB	$1 \text{ MB} = 1024 \text{ KB}$	2^{20} B
Gigabyte	GB	$1 \text{ GB} = 1024 \text{ MB}$	2^{30} B
Terabyte	TB	$1 \text{ TB} = 1024 \text{ GB}$	2^{40} B

Il fattore 1024 deriva dal fatto che $1024 = 2^{10}$ ed è quindi collegato alla numerazione binaria. Per esempio:

$$1 \text{ MB} = 1024 \text{ KB} = 1024 \times 1024 \text{ B} = 1\,048\,576 \text{ B}.$$

La sequenza completa delle unità usate nelle conversioni è **b**, **B**, **KB**, **MB**, **GB**, **TB**. La distinzione tra b minuscola e B maiuscola resta essenziale: il passaggio tra bit e byte utilizza il fattore 8.

2.1.4 La regola delle conversioni

Prima di eseguire una conversione è necessario stabilire se si sta passando verso un'unità più grande oppure verso un'unità più piccola.

Definizione: regola generale

- Da un'unità **più piccola** a un'unità **più grande** si **divide**.
- Da un'unità **più grande** a un'unità **più piccola** si **moltiplica**.

Nella convenzione del libro, tra B, KB, MB, GB e TB si utilizza il fattore 1024 per ogni passaggio tra unità consecutive. Tra B e b si utilizza il fattore 8.

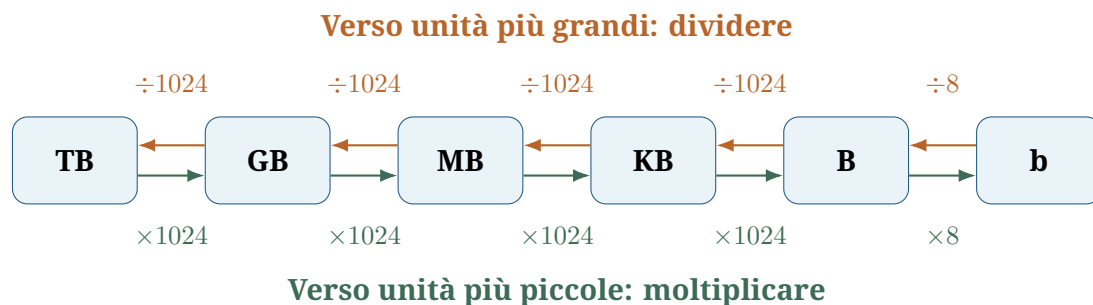


Figura 2.2: Scala delle unità di misura secondo la convenzione adottata nel libro. Il verso della conversione determina se occorre moltiplicare oppure dividere.

La regola consente anche di controllare se il risultato è plausibile:

- esprimendo la stessa quantità con un'unità più piccola, il valore numerico deve aumentare;
- esprimendo la stessa quantità con un'unità più grande, il valore numerico deve diminuire.

Conversione verso un'unità più piccola

Convertire 3 MB in KB significa passare a un'unità più piccola. Si moltiplica quindi per 1024:

$$3 \text{ MB} \times 1024 = 3072 \text{ KB.}$$

Esempio: conversione su più livelli

Convertire 2 GB in KB richiede due passaggi verso unità più piccole:

$$2 \text{ GB} \times 1024 = 2048 \text{ MB,}$$

$$2048 \text{ MB} \times 1024 = 2\,097\,152 \text{ KB.}$$

Il calcolo può essere scritto anche in forma compatta:

$$2 \times 1024 \times 1024 = 2\,097\,152 \text{ KB.}$$

Conversione verso un'unità più grande

Convertire 4096 KB in MB significa passare a un'unità più grande. Si divide quindi per 1024:

$$4096 \text{ KB} \div 1024 = 4 \text{ MB.}$$

Esempio: risultato decimale

Convertire 512 KB in MB:

$$512 \text{ KB} \div 1024 = 0,5 \text{ MB.}$$

Il risultato inferiore a 1 è corretto, perché 512 KB rappresentano metà di un megabyte.

Conversioni tra bit e byte

Il passaggio tra bit e byte utilizza il fattore 8:

$$B \longrightarrow b : \quad \times 8$$

$$b \longrightarrow B : \quad \div 8.$$

Se una quantità deve attraversare anche altri livelli, occorre applicare i fattori nell'ordine corretto.

Esempio: da kilobyte a bit

Convertire 2 KB in bit:

$$2 \text{ KB} \times 1024 = 2048 \text{ B,}$$

$$2048 \text{ B} \times 8 = 16\,384 \text{ b.}$$

Errore comune: usare sempre lo stesso fattore

Tra B, KB, MB, GB e TB si moltiplica o divide per 1024 a ogni passaggio tra unità consecutive. Tra byte e bit si moltiplica o divide per 8. Utilizzare 1024 anche nell'ultimo passaggio tra B e b produce un risultato errato.

2.1.5 Capacità, dimensione e spazio disponibile

La **capacità** indica la quantità massima di dati che una memoria può contenere. La **dimensione di un file**, spesso chiamata informalmente “peso”, indica invece lo spazio occupato dal file. Lo **spazio disponibile** è la parte della capacità che non è ancora utilizzata.

Per confrontare o sottrarre quantità espresse con unità differenti, è necessario convertirle prima nella stessa unità.

Esempio: spazio ancora disponibile

Una memoria ha una capacità di 8 GB e contiene dati per 3072 MB. Per calcolare lo spazio disponibile si convertono gli 8 GB in megabyte:

$$8 \text{ GB} \times 1024 = 8192 \text{ MB.}$$

Si sottrae lo spazio occupato:

$$8192 \text{ MB} - 3072 \text{ MB} = 5120 \text{ MB.}$$

Infine si può riconvertire il risultato in gigabyte:

$$5120 \text{ MB} \div 1024 = 5 \text{ GB.}$$

Esempio: numero teorico di file memorizzabili

Quante fotografie da 8 MB possono essere memorizzate in uno spazio di 2 GB?

$$2 \text{ GB} \times 1024 = 2048 \text{ MB,}$$

$$2048 \text{ MB} \div 8 \text{ MB} = 256.$$

Teoricamente possono essere memorizzate 256 fotografie. Nella realtà una parte dello spazio può essere occupata dal sistema di formattazione o da altri dati.

Esercizi: sezione 2.1

1. Completare le equivalenze:

- a) 1 B = _____ b;
- b) 1 KB = _____ B;
- c) 1 MB = _____ KB;
- d) 1 GB = _____ MB;
- e) 1 TB = _____ GB.

2. Convertire in byte: 16 bit, 72 bit, 256 bit e 2048 bit.

3. Convertire in bit: 4 B, 15 B, 128 B e 3 KB.

4. Convertire:

- a) 5 MB in KB;
- b) 12 GB in MB;
- c) 3 TB in GB;
- d) 7168 KB in MB;
- e) 20 480 MB in GB;
- f) 1536 GB in TB.

5. Convertire 4 GB in KB mostrando tutti i passaggi.

Esercizi: sezione 2.1 (continua)

6. Convertire 8 388 608 B in MB mostrando tutti i passaggi.

7. Completare la tabella.

B	KB	MB	GB
2048			
	4096		
		6	
			2

8. Stabilire se le seguenti affermazioni sono vere o false e correggere quelle false.

- a) Per passare da MB a KB si divide per 1024.
- b) Per passare da bit a byte si divide per 8.
- c) Una stessa quantità espressa in KB ha un valore numerico maggiore rispetto alla sua espressione in MB.
- d) 1024 MB corrispondono a 1 GB.
- e) 8 bit corrispondono a 8 byte.

9. Una chiavetta da 16 GB contiene file per 9216 MB. Calcolare lo spazio ancora disponibile in GB.

10. Un archivio contiene 350 fotografie da 6 MB ciascuna. Calcolare lo spazio totale occupato in MB e in GB.

11. Un video occupa 1,5 GB. Esprimere la sua dimensione in MB e in KB.

12. Una memoria dispone di 4096 MB liberi. Quanti file da 32 MB possono essere memorizzati teoricamente?

13. Correggere il seguente procedimento errato: $5 \text{ GB} \div 1024 = 0,00488 \text{ MB}$.

2.2 Il sistema binario

Il sistema di numerazione utilizzato quotidianamente è il sistema decimale, formato da dieci cifre. I computer, invece, rappresentano internamente numeri e dati attraverso il sistema binario, che utilizza soltanto i simboli 0 e 1. Per comprendere il passaggio tra i due sistemi è necessario richiamare il funzionamento dei sistemi di numerazione posizionali.

2.2.1 Sistemi di numerazione posizionali

In un **sistema di numerazione posizionale**, il valore di una cifra dipende sia dal simbolo utilizzato sia dalla posizione occupata nel numero. Ogni posizione corrisponde a una

potenza della **base** del sistema.

Nel sistema decimale la base è 10 e le cifre disponibili sono:

$$0, 1, 2, 3, 4, 5, 6, 7, 8, 9.$$

Per esempio, il numero 347 può essere scomposto nel modo seguente:

$$(347)_{10} = 3 \times 10^2 + 4 \times 10^1 + 7 \times 10^0.$$

Poiché $10^2 = 100$, $10^1 = 10$ e $10^0 = 1$, si ottiene:

$$347 = 300 + 40 + 7.$$

Il pedice 10 indica che il numero è scritto in base decimale. Quando il contesto è evidente, il pedice può essere omissso.

Definizione: base di un sistema di numerazione

La **base** indica quante cifre differenti vengono utilizzate e determina il valore delle posizioni. Nella parte intera di un numero in base b , le posizioni valgono b^0 , b^1 , b^2 , b^3 e così via, procedendo da destra verso sinistra.

2.2.2 Il sistema binario

Il **sistema binario** è un sistema posizionale in base 2. Utilizza soltanto due cifre:

$$0 \quad \text{e} \quad 1.$$

Per i numeri interi considerati in questa sezione, le posizioni da destra verso sinistra corrispondono alle potenze di 2:

$$2^0, 2^1, 2^2, 2^3, 2^4, \dots$$

I primi valori utili sono riportati nella tabella seguente.

Potenza	2^0	2^1	2^2	2^3	2^4	2^5	2^6	2^7	2^8	2^9	2^{10}
Valore	1	2	4	8	16	32	64	128	256	512	1024

Conoscere le potenze di 2 facilita notevolmente le conversioni e permette di controllare i risultati.

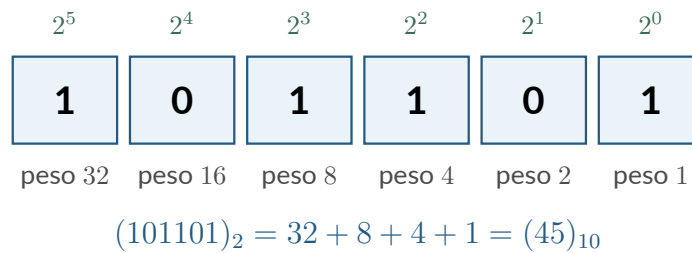


Figura 2.3: Nel sistema binario ogni cifra moltiplica una potenza di 2. Le posizioni contenenti 0 non contribuiscono al valore finale.

2.2.3 Perché i computer usano il sistema binario

I circuiti digitali devono distinguere in modo affidabile gli stati fisici utilizzati per rappresentare i dati. Utilizzare due soli stati offre diversi vantaggi:

- è più semplice distinguere due intervalli elettrici rispetto a un numero elevato di livelli;
- i circuiti possono essere costruiti utilizzando componenti che funzionano come interruttori elettronici;
- eventuali piccole variazioni e disturbi hanno minori probabilità di trasformare uno stato nell'altro;
- le operazioni logiche possono essere realizzate attraverso circuiti relativamente semplici.

I simboli 0 e 1 sono una rappresentazione astratta. Non significano necessariamente che in ogni circuito siano presenti esattamente zero volt e un volt: indicano due condizioni elettriche riconosciute dal sistema come stati logici differenti.

Approfondimento: transistor e stati logici

I processori moderni contengono miliardi di transistor, componenti elettronici che possono controllare il passaggio di corrente e realizzare funzioni logiche. Combinando un grandissimo numero di operazioni elementari è possibile eseguire calcoli, elaborare immagini, riprodurre suoni e far funzionare programmi complessi.



Figura 2.4: Esempio di transistor, componente elettronico utilizzato come elemento di commutazione nei circuiti digitali.

2.2.4 Conversione da binario a decimale

Per convertire un numero binario intero non negativo in decimale si applica il valore posizionale delle cifre.

Il procedimento è il seguente:

1. numerare le posizioni da destra verso sinistra, iniziando da 0;
2. associare a ogni posizione la corrispondente potenza di 2;
3. moltiplicare ogni cifra per il proprio peso;
4. sommare i risultati.

Esempio: conversione di $(110101)_2$

Si associano le potenze di 2 alle cifre:

cifra	1	1	0	1	0	1
potenza	2^5	2^4	2^3	2^2	2^1	2^0

Si moltiplica ogni cifra per il proprio peso:

$$(110101)_2 = 1 \times 32 + 1 \times 16 + 0 \times 8 + 1 \times 4 + 0 \times 2 + 1 \times 1.$$

Si sommano soltanto i pesi corrispondenti alle cifre uguali a 1:

$$32 + 16 + 4 + 1 = 53.$$

Pertanto:

$$(110101)_2 = (53)_{10}.$$

Errore comune: iniziare le potenze da 1

Nella parte intera, la posizione più a destra corrisponde a 2^0 , non a 2^1 . Poiché $2^0 = 1$, dimenticare questa posizione modifica il valore di tutti i pesi e produce una conversione errata.

Metodo rapido mediante la somma dei pesi

Dopo aver acquisito familiarità con le potenze di 2, è possibile leggere direttamente le posizioni che contengono 1. Per esempio:

$$(10011010)_2 = 128 + 16 + 8 + 2 = (154)_{10}.$$

Le posizioni contenenti 0 vengono ignorate, perché il loro contributo è nullo.

2.2.5 Conversione da decimale a binario

Per convertire un numero intero positivo dal sistema decimale al sistema binario si utilizza il metodo delle **divisioni successive per 2**.

Il procedimento è il seguente:

1. dividere il numero per 2;
2. annotare il resto, che può essere soltanto 0 oppure 1;
3. dividere nuovamente per 2 il quoziente ottenuto;
4. continuare finché il quoziente diventa 0;
5. leggere i resti dal basso verso l'alto.

$45 \div 2 = 22$	resto 1
$22 \div 2 = 11$	resto 0
$11 \div 2 = 5$	resto 1
$5 \div 2 = 2$	resto 1
$2 \div 2 = 1$	resto 0
$1 \div 2 = 0$	resto 1

↑
leggere i resti dal basso verso l'alto

$$(45)_{10} = (101101)_2$$

Figura 2.5: Conversione del numero decimale 45 in binario mediante divisioni successive per 2.

Esempio: conversione di $(26)_{10}$

Si eseguono le divisioni successive:

$$\begin{array}{rclcl}
 26 \div 2 & = & 13 & \text{resto } 0 \\
 13 \div 2 & = & 6 & \text{resto } 1 \\
 6 \div 2 & = & 3 & \text{resto } 0 \\
 3 \div 2 & = & 1 & \text{resto } 1 \\
 1 \div 2 & = & 0 & \text{resto } 1
 \end{array}$$

Leggendo i resti dal basso verso l'alto si ottiene:

$$(26)_{10} = (11010)_2.$$

Il risultato può essere controllato riconvertendolo in decimale:

$$(11010)_2 = 16 + 8 + 2 = (26)_{10}.$$

Errore comune: leggere i resti dall'alto verso il basso

Nel metodo delle divisioni successive il primo resto ottenuto rappresenta la cifra meno significativa, cioè quella più a destra. Per questo motivo i resti devono essere letti in ordine inverso, dal basso verso l'alto.

2.2.6 Zeri iniziali e numero di bit

Gli zeri collocati a sinistra di un numero binario non ne modificano il valore:

$$(101)_2 = (0101)_2 = (00000101)_2 = (5)_{10}.$$

Tuttavia, nei computer, il numero di bit utilizzato per una rappresentazione è importante. Uno stesso valore può essere scritto su 4, 8, 16 o più bit aggiungendo zeri iniziali. Per esempio, il numero 13 può essere rappresentato come:

$$(1101)_2 \quad \text{oppure} \quad (00001101)_2.$$

Con 8 bit esistono $2^8 = 256$ combinazioni. Se tutte vengono utilizzate per rappresentare numeri interi non negativi, i valori disponibili vanno da:

$$(00000000)_2 = (0)_{10}$$

fino a:

$$(11111111)_2 = (255)_{10}.$$

La rappresentazione dei numeri negativi richiederà una regola diversa e verrà affrontata nella sezione dedicata al complemento a 2.

Esercizi: sezione 2.2

1. Indicare la base e le cifre disponibili nei sistemi decimale e binario.
2. Scomporre in forma posizionale i numeri decimali 582, 1047 e 9306.
3. Completare la tabella delle potenze di 2 fino a 2^{12} .
4. Convertire in decimale:
 - a) $(101)_2$;
 - b) $(1110)_2$;
 - c) $(10001)_2$;
 - d) $(101101)_2$;
 - e) $(11001010)_2$;
 - f) $(11111111)_2$.
5. Per ciascuno dei numeri precedenti, indicare quali potenze di 2 contribuiscono al risultato.
6. Convertire in binario mediante divisioni successive:
 - a) $(7)_{10}$;
 - b) $(12)_{10}$;
 - c) $(25)_{10}$;

Esercizi: sezione 2.2 (continua)

- d) $(38)_{10}$;
 - e) $(64)_{10}$;
 - f) $(173)_{10}$.
7. Controllare le conversioni dell'esercizio precedente riconvertendo i risultati in decimale.
8. Scrivere su 8 bit i numeri decimali 3, 17, 64, 129 e 250.
9. Stabilire se le seguenti uguaglianze sono corrette. In caso contrario, correggerle.
- a) $(1000)_2 = (8)_{10}$;
 - b) $(10101)_2 = (19)_{10}$;
 - c) $(32)_{10} = (100000)_2$;
 - d) $(50)_{10} = (110010)_2$;
 - e) $(11111111)_2 = (256)_{10}$.
10. Spiegare perché gli zeri iniziali non cambiano il valore di un numero binario ma possono essere comunque importanti in un sistema informatico.
11. Qual è il massimo numero intero non negativo rappresentabile con 4 bit? E con 10 bit? Motivare la risposta.
12. Un dispositivo registra un valore mediante 6 bit senza segno. Quante combinazioni sono disponibili e quali sono il valore minimo e quello massimo rappresentabili?
13. Descrivere con parole proprie perché i circuiti digitali utilizzano due stati logici.
14. Individuare e correggere l'errore nel seguente procedimento: $13 \div 2 = 6$ resto 1; $6 \div 2 = 3$ resto 0; $3 \div 2 = 1$ resto 1; $1 \div 2 = 0$ resto 1; quindi $(13)_{10} = (1011)_2$ leggendo i resti dall'alto verso il basso.

2.3 I numeri binari con la virgola

Per convertire numeri con una parte frazionaria si usano le **moltiplicazioni successive per 2**. Se il risultato è troppo lungo, in questa sezione lo arrotondiamo a **10 cifre dopo la virgola in base 2**, cioè 10 bit frazionari.

2.3.1 Il valore delle cifre dopo la virgola

Finora abbiamo convertito soprattutto numeri interi. Un numero come 13,625 contiene una **parte intera**, pari a 13, e una **parte frazionaria**, pari a 0,625. Anche in binario queste due parti sono separate dalla virgola.

Nel sistema decimale le prime posizioni dopo la virgola valgono un decimo, un centesimo e un millesimo. In binario valgono invece **un mezzo, un quarto, un ottavo** e così via: a ogni passo verso destra il peso si dimezza.

Posizione	Potenza di 2	Frazione	Valore decimale
1 ^a	2^{-1}	$1/2$	0,5
2 ^a	2^{-2}	$1/4$	0,25
3 ^a	2^{-3}	$1/8$	0,125
4 ^a	2^{-4}	$1/16$	0,0625
5 ^a	2^{-5}	$1/32$	0,03125

Tabella 2.2: I pesi delle prime cinque cifre binarie dopo la virgola.

La posizione k dopo la virgola ha peso $2^{-k} = 1/2^k$. La decima posizione, per esempio, vale $1/1024 = 0,0009765625$.

Esempio: dal binario al decimale

Si moltiplica ogni cifra per il suo peso e si sommano i contributi:

$$\begin{aligned}(10,101)_2 &= 1 \cdot 2^1 + 0 \cdot 2^0 + 1 \cdot 2^{-1} + 0 \cdot 2^{-2} + 1 \cdot 2^{-3} \\ &= 2 + 0,5 + 0,125 = (2,625)_{10}.\end{aligned}$$

La virgola separa le due parti, ma **non è una cifra e non ha un peso**.

Errore comune: confondere la base

$(0,1)_2 = (0,5)_{10}$. La scrittura 0,1 in base 2 indica un mezzo; in base 10 indica un decimo. Le stesse cifre possono rappresentare valori differenti.

2.3.2 Le moltiplicazioni successive per 2

Per convertire un numero decimale non negativo, si trattano separatamente le due parti: **divisioni successive per 2 per la parte intera, moltiplicazioni successive per 2 per la parte frazionaria**.

Per la parte frazionaria il procedimento è il seguente:

1. Moltiplicare per 2 la parte frazionaria, che è compresa tra 0 incluso e 1 escluso.
2. Scrivere la parte intera del prodotto: può essere soltanto **0 oppure 1**. È la prossima cifra binaria dopo la virgola.
3. Conservare soltanto la nuova parte frazionaria del prodotto e moltiplicarla ancora per 2.
4. Ripetere fino a ottenere una parte frazionaria nulla oppure un numero di cifre sufficiente per l'approssimazione richiesta.

Errore comune: ordine di lettura delle cifre

Non arrotondare i risultati intermedi: l'approssimazione si esegue soltanto alla fine. Le cifre ottenute con le moltiplicazioni si leggono **dall'alto verso il basso**, nell'ordine in cui compaiono. La prima è subito dopo la virgola. I resti delle divisioni della parte intera si leggono invece dal basso verso l'alto.

Esempio: conversione di $(0,625)_{10}$

Passo	Moltiplicazione	Cifra ottenuta	Nuova parte frazionaria
1	$0,625 \times 2 = 1,25$	1	0,25
2	$0,25 \times 2 = 0,5$	0	0,5
3	$0,5 \times 2 = 1$	1	0

La parte frazionaria è diventata zero: il procedimento termina e il risultato è **esatto**:

$$(0,625)_{10} = (0,101)_2.$$

Il controllo con i pesi dà $1/2 + 1/8 = 0,5 + 0,125 = 0,625$.

Esempio: conversione di $(13,625)_{10}$

La parte intera è 13, che con le divisioni successive dà $(1101)_2$. La parte frazionaria è 0,625, appena convertita in $(0,101)_2$. Si uniscono i risultati:

$$(13,625)_{10} = (1101,101)_2.$$

Il controllo è $8 + 4 + 1 + 0,5 + 0,125 = 13,625$.

Perché il metodo funziona?

Moltiplicare per 2 sposta di una posizione a sinistra le cifre binarie: la prima cifra dopo la virgola diventa la parte intera del prodotto. Togliendola e ripetendo, si estraggono una alla volta le cifre successive.

Errore comune: ripartire dalla parte frazionaria

La parte frazionaria di 13,625 è 0,625, non 625. Dopo $0,625 \times 2 = 1,25$, si riparte da 0,25, non da 1,25.

2.3.3 Quando la conversione termina?

Alcuni numeri hanno una rappresentazione binaria finita, altri richiedono infinite cifre. **Un numero con poche cifre in base 10 può avere infinite cifre in base 2.**

Esempio: conversione del numero decimale 0,1

Passo	Moltiplicazione	Cifra ottenuta	Nuova parte frazionaria
1	$0,1 \times 2 = 0,2$	0	0,2
2	$0,2 \times 2 = 0,4$	0	0,4
3	$0,4 \times 2 = 0,8$	0	0,8
4	$0,8 \times 2 = 1,6$	1	0,6
5	$0,6 \times 2 = 1,2$	1	0,2

Alla fine del quinto passo ricompare 0,2, già usato all'inizio del secondo. Da quel punto le stesse moltiplicazioni e le stesse cifre si ripeteranno:

$$(0,1)_{10} = (0,0001100110011\dots)_2 = (0,0\overline{0011})_2.$$

La barra indica il **periodo**, cioè il gruppo di cifre che si ripete all'infinito. La parte frazionaria non diventa mai zero: per una scrittura di lunghezza limitata occorre un'approssimazione.

Una regola per prevederlo

Si scrive il valore come frazione e la si **riduce ai minimi termini**. Un numero razionale ha una rappresentazione binaria finita se e solo se il denominatore della frazione ridotta è una **potenza di 2**: 1, 2, 4, 8, 16, ...

Numero	Frazione ridotta	Sviluppo binario
0,625	5/8	Finito: $(0,101)_2$
0,75	3/4	Finito: $(0,11)_2$
0,1	1/10	Infinito periodico
0,2	1/5	Infinito periodico
1/3	1/3	Infinito periodico: $(0,\overline{01})_2$

Tabella 2.3: Rappresentazioni binarie finite e infinite periodiche.

La riduzione è essenziale: $0,5 = 5/10 = 1/2$ termina in binario, anche se il denominatore iniziale era 10. Il criterio funziona perché una sequenza finita di n cifre binarie dopo la virgola rappresenta una frazione con denominatore 2^n .

Approfondimento: sviluppi infiniti

I numeri **razionali** hanno uno sviluppo finito oppure infinito periodico, eventualmente dopo alcune cifre iniziali. I numeri **irrazionali**, come $\sqrt{2}$, hanno invece uno sviluppo infinito non periodico. Il criterio del denominatore riguarda i numeri razionali.

2.3.4 Arrotondare a 10 cifre binarie

In questa sezione, quando sono necessarie più di 10 cifre dopo la virgola, si conserva il valore più vicino rappresentabile con **10 bit frazionari**. In caso di esatta parità, si sceglie il valore con **maggiore valore assoluto**: per i numeri positivi si arrotonda quindi verso l'alto.

Definizione: arrotondamento a 10 bit frazionari

Per i numeri non negativi:

1. Calcolare **11 cifre binarie** dopo la virgola, se il procedimento non termina prima.
2. Conservare le prime 10 e osservare l'undicesima.
3. Se l'undicesima è **0**, lasciare invariate le prime 10 cifre.
4. Se l'undicesima è **1**, aggiungere $(0,0000000001)_2$, cioè un'unità nella decima posizione, gestendo gli eventuali riporti.

Questa regola comprende il caso di parità secondo la convenzione dichiarata. Se la conversione termina prima, le cifre mancanti valgono zero.

Esempio: l'undicesima cifra è 0

Per 0,1, le prime 11 cifre sono:

$$0, \underbrace{0001100110}_{10 \text{ cifre da conservare}} \underbrace{0}_{11^{\text{a}} \text{ cifra}} \dots$$

Non si incrementa l'ultima cifra conservata:

$$(0,1)_{10} \approx (0,0001100110)_2 = (0,099609375)_{10}.$$

L'errore assoluto è $|0,1 - 0,099609375| = 0,000390625$.

Esempio: l'undicesima cifra è 1

Per 0,2, le prime 11 cifre sono 0,00110011001... Si conserva 0,0011001100 e si aggiunge un'unità nell'ultima posizione:

$$(0,0011001100)_2 + (0,0000000001)_2 = (0,0011001101)_2.$$

Quindi $(0,2)_{10} \approx (0,0011001101)_2 = (0,2001953125)_{10}$.

Arrotondamento e troncamento sono diversi

Troncamento significa eliminare tutte le cifre oltre la decima senza modificare quelle conservate. Per 0,2 si otterrebbe $(0,0011001100)_2$, che è meno vicino al valore esatto rispetto al risultato arrotondato.

Con 10 bit frazionari, la distanza tra valori consecutivi è $2^{-10} = 1/1024$. L'errore assoluto dell'arrotondamento al più vicino è al massimo metà di questa distanza:

$$|\text{valore esatto} - \text{valore arrotondato}| \leq \frac{1}{2048} = 0,00048828125.$$

Dieci cifre binarie non sono dieci cifre decimali. Il limite riguarda le posizioni in base 2; per un risultato approssimato si usa $\approx$, non $=$.

2.3.5 Casi particolari da riconoscere

La parte frazionaria diventa zero

La conversione è terminata ed è esatta. Gli zeri aggiunti a destra non cambiano il valore: $(0,101)_2 = (0,1010000000)_2$. Anche un numero intero si può scrivere con soli zeri dopo la virgola. **Non occorre continuare le moltiplicazioni.**

Il riporto raggiunge la parte intera

Con $(0,9999)_{10}$, le prime 11 cifre binarie sono tutte 1. L'undicesima richiede un incremento che si propaga lungo tutte le dieci cifre conservate:

$$\begin{array}{r} 0,1111111111 \\ + 0,0000000001 \\ \hline 1,0000000000 \end{array} \quad (\text{somma in base 2})$$

Quindi $(0,9999)_{10} \approx (1,0000000000)_2$. Un numero minore di 1 può arrotondarsi a 1: **la parte intera può cambiare.**

Il valore è esattamente a metà

Il numero $1/2048 = 0,00048828125$ termina in binario, ma richiede **11 cifre** dopo la virgola: $(0,00000000001)_2$. È a metà fra $(0,0000000000)_2$ e $(0,0000000001)_2$. Con la nostra convenzione si arrotonda al secondo. Una conversione finita può dunque richiedere comunque un'approssimazione al limite scelto.

Un numero positivo molto piccolo può diventare zero

$0,0001$ è minore di $1/2048$: con 10 bit frazionari si arrotonda a $(0,0000000000)_2$. Il valore iniziale **non è zero**, ma la precisione scelta non consente di distinguerlo da zero mediante questo arrotondamento.

Il numero è negativo

Per la conversione matematica si lavora sul valore assoluto e si riapplica il segno: $(-2,625)_{10} = -(10,101)_2$. Per l'approssimazione si arrotonda il valore assoluto con la stessa regola; nei casi a metà ci si allontana quindi da zero. Il segno meno è qui un simbolo matematico: **non stiamo definendo la codifica in memoria** mediante complemento a 2 o virgola mobile.

Approfondimento: due scritture dello stesso valore

Una scrittura finita può avere anche una forma con infiniti 1:

$$(0,1000\dots)_2 = (0,0111\dots)_2 = (0,5)_{10}.$$

Infatti $1/4 + 1/8 + 1/16 + \dots = 1/2$. L'uguaglianza richiede **infiniti 1**: $(0,0111)_2$ è invece minore di 0,5. Nel procedimento usiamo sempre la forma finita quando esiste, fermandoci appena la parte frazionaria diventa zero.

2.3.6 Esercizi proposti**Esercizi: numeri binari con la virgola**

Indicare sempre la base e mostrare i passaggi. Usare = per le conversioni esatte e $\approx$ per i risultati arrotondati. Quando occorre, arrotondare a 10 cifre binarie dopo la virgola con la convenzione di questa sezione.

1. Convertire in decimale $(0,1011)_2$ usando i pesi posizionali.
2. Convertire in binario $(0,375)_{10}$ con le moltiplicazioni successive per 2 e controllare il risultato.
3. Convertire in binario $(6,6875)_{10}$, separando parte intera e parte frazionaria.
4. Stabilire quali valori hanno una rappresentazione binaria finita: 0,45; 0,125; 0,2; 0,75. Motivare mediante le frazioni ridotte ai minimi termini.
5. Convertire $(0,3)_{10}$ calcolando 11 cifre binarie e arrotondando a 10. Confrontare il valore arrotondato con quello esatto.
6. Arrotondare $(0,9999)_{10}$ a 10 bit frazionari. Mostrare il riporto e spiegare perché cambia la parte intera.
7. Arrotondare $1/2048$ e $0,0001$ a 10 bit frazionari. Spiegare perché si ottengono risultati diversi.
8. Individuare e correggere gli errori: «0,375 dà le cifre 0, 1, 1, che vanno lette dal basso verso l'alto. Ogni decimale finito termina anche in binario. Per arrotondare a 10 cifre basta calcolarne 10».

2.3.7 Soluzioni commentate

1. $1/2 + 1/8 + 1/16 = (0,6875)_{10}$.
2. $0,375 \times 2 = 0,75$; $0,75 \times 2 = 1,5$; $0,5 \times 2 = 1$. Le cifre sono 0, 1, 1: $(0,375)_{10} = (0,011)_2$.
3. $6 = (110)_2$; $0,6875 = (0,1011)_2$. Risultato: $(6,6875)_{10} = (110,1011)_2$.
4. Finite: $0,125 = 1/8$ e $0,75 = 3/4$. Periodiche: $0,45 = 9/20$ e $0,2 = 1/5$.
5. Prime 11 cifre: $0,01001100110\dots$. L'undicesima è 0: $(0,3)_{10} \approx (0,0100110011)_2$. Il valore ottenuto è 0,2998046875; l'errore assoluto è 0,0001953125.

6. $(0,9999)_{10} \approx (1,0000000000)_2$. Il riporto attraversa tutte le dieci cifre frazionarie, inizialmente pari a 1.
7. $1/2048 \approx (0,0000000001)_2$, per la regola dei casi a metà; $(0,0001)_{10} \approx (0,0000000000)_2$, perché è più vicino a zero.
8. Le cifre frazionarie si leggono dall'alto verso il basso; il risultato è $(0,011)_2$, non $(0,110)_2$. Un decimale finito può essere periodico in binario. Per l'arrotondamento servono 11 cifre, salvo termine anticipato.

2.4 Le operazioni con i numeri binari

I numeri binari possono essere utilizzati per eseguire le stesse operazioni aritmetiche studiate nel sistema decimale. Le regole risultano particolarmente semplici perché le uniche cifre disponibili sono 0 e 1. È però necessario prestare attenzione ai riporti, ai prestiti e all'allineamento delle cifre.

In questa sezione si considerano inizialmente numeri interi non negativi. La rappresentazione dei numeri negativi verrà introdotta nella sezione successiva mediante il complemento a 2.

Definizione: bit meno significativo e bit più significativo

In un numero binario intero non negativo, il bit collocato più a destra è il **bit meno significativo**, perché possiede peso 2^0 . Il bit più a sinistra è il **bit più significativo**, perché possiede il peso maggiore. Durante le operazioni in colonna le cifre devono essere allineate a partire dal bit meno significativo.

2.4.1 L'addizione binaria

L'addizione binaria segue lo stesso principio dell'addizione decimale: si procede da destra verso sinistra e, quando il risultato di una colonna non può essere scritto con una sola cifra, si produce un riporto nella colonna successiva.

Le regole fondamentali sono:

Operazione	Risultato	Osservazione
$0 + 0$	0	nessun riporto
$0 + 1$	1	nessun riporto
$1 + 0$	1	nessun riporto
$1 + 1$	$(10)_2$	si scrive 0 e si riporta 1
$1 + 1 + 1$	$(11)_2$	si scrive 1 e si riporta 1

La scrittura $(10)_2$ non rappresenta il numero decimale dieci: in binario equivale a $(2)_{10}$. Per questo motivo, quando si calcola $1 + 1$, si scrive 0 nella colonna corrente e si aggiunge un riporto pari a 1 nella colonna immediatamente a sinistra.

Esempio: addizione guidata: $(1011)_2 + (0110)_2$

Si allineano i numeri a destra e si procede dalla colonna meno significativa:

$$\begin{array}{r} \\ 0 1 1 \\ + 0 1 1 0 \\ \hline 1 0 0 0 1 \end{array}$$

Il calcolo può essere seguito colonna per colonna:

1. $1 + 0 = 1$;
2. $1 + 1 = (10)_2$: si scrive 0 e si riporta 1;
3. $0 + 1 + 1 = (10)_2$: si scrive 0 e si riporta 1;
4. $1 + 0 + 1 = (10)_2$: si scrive 0 e si riporta 1 nella nuova posizione a sinistra.

Quindi:

$$(1011)_2 + (0110)_2 = (10001)_2.$$

La verifica in decimale conferma il risultato:

$$11 + 6 = 17.$$

Esempio: esercizio guidato: $(11001)_2 + (00111)_2$

Si predispone l'operazione:

$$\begin{array}{r} 11001 \\ + 00111 \\ \hline \end{array}$$

Si completano progressivamente le colonne:

1. nella prima colonna si calcola $1 + 1 = (10)_2$;
2. nella seconda colonna si calcola $0 + 1$ e si aggiunge il riporto;
3. si prosegue fino alla colonna più a sinistra;
4. si controlla il risultato convertendo i due addendi in decimale.

Il risultato corretto è:

$$(11001)_2 + (00111)_2 = (100000)_2,$$

poiché $25 + 7 = 32$.

Errore comune: dimenticare il riporto

Il riporto prodotto in una colonna deve essere sommato alle cifre della colonna successiva. Non è sufficiente applicare separatamente le regole $0 + 0$, $0 + 1$ e $1 + 1$: in alcune colonne si devono sommare tre cifre, cioè i due bit degli addendi e il riporto.

Il riporto finale

Se l'addizione produce un riporto oltre la posizione più a sinistra, il risultato richiede un bit in più. Per esempio:

$$(1111)_2 + (0001)_2 = (10000)_2.$$

Il primo numero occupa 4 bit, mentre il risultato ne richiede 5. Quando il numero di bit disponibile è fissato, questo fatto può provocare un **overflow**. Il concetto verrà ripreso nella sezione sul complemento a 2.

2.4.2 La sottrazione binaria diretta

La sottrazione binaria si esegue in colonna da destra verso sinistra. Quando la cifra superiore è minore di quella inferiore, si richiede un prestito alla colonna a sinistra.

Le regole fondamentali sono:

Operazione	Risultato	Osservazione
$0 - 0$	0	nessun prestito
$1 - 0$	1	nessun prestito
$1 - 1$	0	nessun prestito
$0 - 1$	non eseguibile direttamente	occorre un prestito

Quando una colonna riceve un prestito, il valore 1 proveniente dalla colonna a sinistra vale $(10)_2$ nella colonna corrente. Si può quindi calcolare:

$$(10)_2 - (1)_2 = (1)_2.$$

La cifra che ha ceduto il prestito diminuisce invece di una unità.

Esempio: sottrazione guidata: $(1101)_2 - (0011)_2$

Si allineano i numeri:

$$\begin{array}{r} 1101 \\ -0011 \\ \hline 1010 \end{array}$$

Il procedimento è il seguente:

1. nella prima colonna: $1 - 1 = 0$;
2. nella seconda colonna: $0 - 1$ richiede un prestito; il valore superiore diventa $(10)_2$ e si calcola $(10)_2 - (1)_2 = (1)_2$;
3. la cifra della terza colonna, dopo aver ceduto il prestito, passa da 1 a 0; quindi $0 - 0 = 0$;
4. nell'ultima colonna: $1 - 0 = 1$.

Esempio: sottrazione guidata: $(1101)_2 - (0011)_2$ (continua)

Pertanto:

$$(1101)_2 - (0011)_2 = (1010)_2.$$

La verifica decimale è:

$$13 - 3 = 10.$$

Esempio: prestito attraverso più colonne

Nella sottrazione

$$(10000)_2 - (00001)_2$$

il bit 1 più a sinistra deve fornire il prestito attraversando le colonne intermedie contenenti 0. Il risultato è:

$$(10000)_2 - (00001)_2 = (01111)_2.$$

La verifica decimale è $16 - 1 = 15$.

Errore comune: modificare soltanto la colonna corrente

Quando si prende un prestito, non basta trasformare lo 0 corrente in $(10)_2$: bisogna anche diminuire la cifra che ha fornito il prestito. Se tra le due posizioni sono presenti più zeri, il prestito deve propagarsi attraverso tutte le colonne interessate.

In questa sezione la sottrazione diretta viene applicata soltanto quando il primo numero è maggiore o uguale al secondo. Per ottenere un risultato negativo verrà utilizzata la rappresentazione in complemento a 2.

2.4.3 La moltiplicazione binaria

La moltiplicazione binaria è basata su quattro regole molto semplici:

$$0 \times 0 = 0, \quad 0 \times 1 = 0, \quad 1 \times 0 = 0, \quad 1 \times 1 = 1.$$

Come nella moltiplicazione decimale, si calcolano i prodotti parziali procedendo dalle cifre del moltiplicatore, da destra verso sinistra. Moltiplicare per 0 produce una riga di zeri; moltiplicare per 1 riproduce il numero di partenza. Quando si passa alla seconda cifra del moltiplicatore si aggiunge uno zero a destra del prodotto parziale; con la terza cifra si aggiungono due zeri, e così via. Questi zeri realizzano lo spostamento verso sinistra richiesto dal valore posizionale delle cifre binarie.

Esempio: moltiplicazione guidata: $(101)_2 \times (11)_2$

Si dispongono i fattori in colonna:

$$\begin{array}{r} 101 \\ \times 11 \\ \hline 101 \\ +1010 \\ \hline 1111 \end{array}$$

Il primo bit del moltiplicatore è 1, quindi il primo prodotto parziale è $(101)_2$. Anche il secondo bit è 1, ma si trova nella posizione di peso 2^1 : si riscrive quindi il moltiplicando aggiungendo uno zero a destra, ottenendo $(1010)_2$. Se fosse presente una terza cifra uguale a 1, il relativo prodotto parziale avrebbe due zeri aggiunti a destra.

La somma dei prodotti parziali fornisce:

$$(101)_2 \times (11)_2 = (1111)_2.$$

In decimale:

$$5 \times 3 = 15.$$

Esempio: moltiplicazione con uno zero nel moltiplicatore

Calcolare $(1011)_2 \times (110)_2$:

$$\begin{array}{r} 1011 \\ \times 110 \\ \hline 0000 \\ 10110 \\ +101100 \\ \hline 1000010 \end{array}$$

Il bit meno significativo del moltiplicatore è 0 e genera una riga nulla. Gli altri due bit sono 1 e producono due copie del moltiplicando, spostate rispettivamente di una e di due posizioni. Il risultato è corretto perché $11 \times 6 = 66$ e $(66)_{10} = (1000010)_2$.

Approfondimento: spostamenti e potenze di 2

Spostare tutte le cifre di un numero binario di una posizione verso sinistra equivale a moltiplicarlo per 2, purché non venga perso alcun bit. Per esempio:

$$(1011)_2 = (11)_{10}, \quad (10110)_2 = (22)_{10}.$$

Uno spostamento di due posizioni verso sinistra equivale a moltiplicare per $2^2 = 4$. Questo principio è utilizzato nei prodotti parziali della moltiplicazione binaria.

2.4.4 Controllare il risultato delle operazioni

Durante l'apprendimento è utile verificare ogni operazione convertendo i numeri in decimale. Il controllo non sostituisce il procedimento binario, ma permette di individuare più facilmente riporti, prestiti o spostamenti dimenticati.

Per esempio:

$$(10110)_2 + (00101)_2 = (11011)_2$$

può essere controllato osservando che:

$$22 + 5 = 27.$$

Analogamente:

$$(11000)_2 - (00101)_2 = (10011)_2$$

corrisponde a:

$$24 - 5 = 19.$$

Esercizi: sezione 2.4

1. Completare le seguenti regole:

- a) $0 + 0 = \dots$;
- b) $1 + 1 = \dots$ con riporto $\dots$;
- c) $1 + 1 + 1 = \dots$ con riporto $\dots$;
- d) $(10)_2 - (1)_2 = \dots$;
- e) $1 \times 1 = \dots$

2. Eseguire le addizioni senza convertirle prima in decimale:

- a) $(101)_2 + (010)_2$;
- b) $(1001)_2 + (0010)_2$;
- c) $(10101)_2 + (00010)_2$;
- d) $(11000)_2 + (00111)_2$.

3. Eseguire le addizioni con riporti multipli:

- a) $(111)_2 + (001)_2$;
- b) $(1011)_2 + (0111)_2$;
- c) $(11101)_2 + (01111)_2$;
- d) $(101101)_2 + (110011)_2$.

4. Scrivere su 8 bit gli addendi dell'esercizio seguente, eseguire la somma e indicare se il risultato richiede un nono bit: $(173)_{10} + (99)_{10}$.

5. Eseguire le sottrazioni dirette:

- a) $(1011)_2 - (0010)_2$;
- b) $(1110)_2 - (0101)_2$;

Esercizi: sezione 2.4 (continua)

- c) $(10101)_2 - (00111)_2$;
d) $(100000)_2 - (000001)_2$.
6. Nelle sottrazioni dell'esercizio precedente, evidenziare le colonne in cui è stato necessario un prestito.
7. Eseguire le moltiplicazioni:
- a) $(101)_2 \times (10)_2$;
b) $(110)_2 \times (11)_2$;
c) $(1001)_2 \times (101)_2$;
d) $(1110)_2 \times (110)_2$.
8. Verificare in decimale tutti i risultati degli esercizi 2, 5 e 7.
9. Individuare e correggere l'errore:
- $$(1011)_2 + (0110)_2 = (11001)_2.$$
10. Individuare e correggere l'errore nella moltiplicazione: il secondo prodotto parziale di $(101)_2 \times (11)_2$ è stato scritto come $(101)_2$ senza spostarlo verso sinistra.
11. Spiegare perché aggiungere uno zero a destra di un numero binario intero equivale a moltiplicarlo per 2.
12. Preparare una breve procedura di controllo per verificare un'operazione binaria mediante la conversione in decimale.

2.5 Numeri negativi e complemento a 2

Nella sezione precedente abbiamo svolto operazioni con numeri interi non negativi. Un computer deve però rappresentare anche temperature sotto lo zero, differenze negative, spostamenti in direzioni opposte e risultati di calcoli in cui il sottraendo è maggiore del minuendo.

Per rappresentare correttamente numeri positivi e negativi è necessario stabilire in anticipo il numero di bit utilizzato. Nei sistemi informatici il metodo più diffuso per gli interi con segno è il **complemento a 2**.

In questa sezione una sequenza interpretata come numero con segno in complemento a 2 viene indicata con il pedice c2. Per esempio, $(11111011)_{c2}$ rappresenta -5 su 8 bit.

2.5.1 Perché occorre stabilire il numero di bit

Una sequenza binaria non possiede da sola un significato unico: il suo valore dipende dalla convenzione con cui viene interpretata. Per esempio, la sequenza a 8 bit 11111011 può essere interpretata in due modi:

$$(11111011)_2 = (251)_{10}, \quad (11111011)_{c2} = -5.$$

La prima scrittura considera il numero senza segno; la seconda specifica che la stessa sequenza deve essere interpretata in complemento a 2.

La lunghezza della rappresentazione è essenziale. Il complemento a 2 di un numero deve essere calcolato mantenendo sempre lo stesso numero di bit.

Errore comune: cambiare il numero di bit durante il calcolo

Non è corretto calcolare il complemento a 2 di una sequenza a 8 bit eliminando gli zeri iniziali o aggiungendo bit durante il procedimento. Tutte le operazioni devono essere svolte entro la lunghezza stabilita.

2.5.2 L'intervallo dei valori rappresentabili

Con n bit in complemento a 2 si possono rappresentare i valori compresi tra:

$$-2^{n-1}$$

e

$$2^{n-1} - 1.$$

Con 4 bit l'intervallo è quindi:

$$-2^3 = -8 \quad \text{fino a} \quad 2^3 - 1 = 7.$$

La tabella seguente mostra tutte le combinazioni a 4 bit.

Sequenza a 4 bit	Valore decimale in complemento a 2
(0000) _{c2}	0
(0001) _{c2}	1
(0010) _{c2}	2
(0011) _{c2}	3
(0100) _{c2}	4
(0101) _{c2}	5
(0110) _{c2}	6
(0111) _{c2}	7
(1000) _{c2}	−8
(1001) _{c2}	−7
(1010) _{c2}	−6
(1011) _{c2}	−5
(1100) _{c2}	−4
(1101) _{c2}	−3
(1110) _{c2}	−2
(1111) _{c2}	−1

Tabella 2.4: Tutte le combinazioni a 4 bit interpretate in complemento a 2.

Il bit più significativo permette una prima interpretazione:

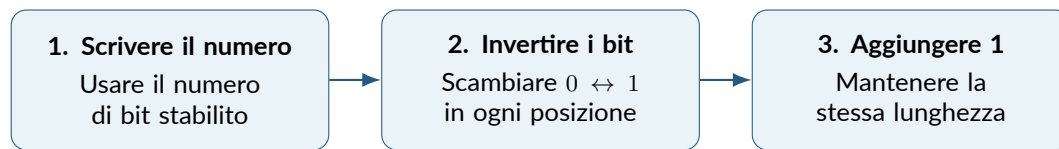
- se il primo bit è 0, il numero è non negativo e si legge normalmente;
- se il primo bit è 1, il numero è negativo e deve essere interpretato con la regola del complemento a 2.

Non si deve però considerare il primo bit come un semplice segno indipendente: tutti i bit partecipano alla rappresentazione del valore.

2.5.3 Ottenere l'opposto di un numero

Se l'opposto è rappresentabile con il numero di bit scelto, per ottenerlo in complemento a 2 si applicano due operazioni:

1. si invertono tutti i bit, sostituendo ogni 0 con 1 e ogni 1 con 0;
2. si aggiunge 1 al risultato, mantenendo lo stesso numero di bit.



Il risultato rappresenta l'opposto **solo se è rappresentabile** con gli stessi bit.
 Su 8 bit, -128 è l'eccezione: 10000000 torna 10000000 .

Figura 2.6: Procedimento per ottenere l'opposto in complemento a 2, purché sia rappresentabile con il numero di bit scelto.

Errore comune: il valore minimo non ha un opposto rappresentabile

Su 8 bit l'intervallo va da -128 a $+127$: $+128$ non è rappresentabile. Applicando “inverti e aggiungi 1” alla sequenza di -128 si ottiene ancora:

$$10000000 \rightarrow 01111111 \rightarrow 10000000.$$

Il risultato, letto in complemento a 2, resta -128 : non è il suo opposto. L'eccezione riguarda in generale il valore minimo -2^{n-1} su n bit.

Esempio: rappresentare -5 su 8 bit

Si parte dalla rappresentazione positiva di 5:

$$+5 = (00000101)_{c2}.$$

Si invertono tutti i bit:

$$(11111010)_{c2}.$$

Si aggiunge 1:

$$\begin{array}{r} 11111010 \\ +00000001 \\ \hline 11111011 \end{array}$$

Quindi, su 8 bit:

$$-5 = (11111011)_{c2}.$$

Esempio: esercizio guidato: rappresentare -18 su 8 bit

Si scrive inizialmente il valore positivo:

$$(18)_{10} = (00010010)_{c2}.$$

Si completano i due passaggi:

inversione : _____
 aggiunta di 1 : _____

Il risultato corretto è:

$$-18 = (11101110)_{c2}.$$

Curiosità: applicare due volte il complemento a 2

Se il procedimento “inverti e aggiungi 1” viene applicato due volte, si ritorna al numero di partenza, purché il numero di bit rimanga invariato. Il procedimento restituisce l'opposto quando questo è rappresentabile. Il valore minimo, invece, torna su sé stesso già dopo una sola applicazione; anche lo zero resta zero.

2.5.4 Interpretare un numero negativo

Quando una sequenza in complemento a 2 inizia con 1, per determinarne il valore si può applicare nuovamente lo stesso procedimento:

1. invertire tutti i bit;
2. aggiungere 1;
3. leggere il risultato **senza segno** e convertirlo in decimale: si ottiene il modulo;
4. assegnare il segno meno.

Questo metodo per ricavare il modulo vale anche per il minimo: da 10000000 si ottiene ancora 10000000 , che letto *senza segno* vale 128. Assegnando il segno meno si ricava correttamente -128 .

Esempio: interpretare $(11111011)_{c2}$

Il primo bit è 1, quindi il numero è negativo. Si invertono i bit:

$$(00000100)_{c2}.$$

Si aggiunge 1:

$$(00000101)_{c2} = (5)_{10}.$$

La sequenza iniziale rappresenta quindi:

$$(11111011)_{c2} = (-5)_{10}.$$

Errore comune: convertire direttamente come numero senza segno

Una sequenza con primo bit uguale a 1 non deve essere convertita sommando semplicemente tutti i pesi positivi, quando il contesto specifica che si tratta di complemento a 2. La stessa sequenza può avere un valore diverso se interpretata come numero senza segno.

2.5.5 Somma di numeri con segno

Uno dei principali vantaggi del complemento a 2 è che numeri positivi e negativi possono essere sommati mediante la normale addizione binaria. Non è necessario costruire un circuito separato per la sottrazione.

Il procedimento è:

1. rappresentare entrambi gli operandi con lo stesso numero di bit;
2. trasformare gli eventuali valori negativi in complemento a 2;
3. eseguire la normale addizione binaria;
4. eliminare l'eventuale riporto oltre il numero di bit stabilito;
5. interpretare il risultato in complemento a 2.

Esempio: somma $7 + (-3)$ su 8 bit

Le rappresentazioni sono:

$$+7 = (00000111)_{c2},$$

$$-3 = (11111101)_{c2}.$$

Si esegue la somma:

$$\begin{array}{r} 00000111 \\ +11111101 \\ \hline 1\,00000100 \end{array}$$

Il riporto esterno al gruppo di 8 bit viene eliminato. Rimane:

$$(00000100)_{c2} = (4)_{10}.$$

Pertanto:

$$7 + (-3) = 4.$$

Esempio: somma con risultato negativo: $3 + (-7)$

Su 8 bit:

$$+3 = (00000011)_{c2}, \quad -7 = (11111001)_{c2}.$$

La somma è:

$$\begin{array}{r} 00000011 \\ +11111001 \\ \hline 11111100 \end{array}$$

Il risultato inizia con 1. Applicando il complemento a 2 a $(11111100)_{c2}$ si ottiene $(00000100)_{c2}$, quindi:

$$(11111100)_{c2} = (-4)_{10}.$$

2.5.6 Eseguire una sottrazione usando soltanto la somma

In complemento a 2 la sottrazione viene trasformata nell'addizione dell'opposto:

$$A - B = A + (-B).$$

Per calcolare $A - B$ si rappresenta A , si costruisce il complemento a 2 di B e si sommano i due numeri.

Esempio: sottrazione con risultato positivo: $7 - 3$

Si trasforma la sottrazione in:

$$7 - 3 = 7 + (-3).$$

Su 8 bit:

$$\begin{array}{r} 00000111 \\ +11111101 \\ \hline 1\,00000100 \end{array}$$

Eliminando il riporto esterno si ottiene $(00000100)_{c2} = (4)_{10}$.

Esempio: sottrazione con risultato negativo: $3 - 7$

Si scrive:

$$3 - 7 = 3 + (-7).$$

Su 8 bit:

$$\begin{array}{r} 00000011 \\ +11111001 \\ \hline 11111100 \end{array}$$

Il risultato è negativo. Applicando nuovamente il complemento a 2 si ricava il modulo 4, quindi:

$$3 - 7 = -4.$$

Esempio: esercizio guidato: $12 - 19$ su 8 bit

Si rappresentano i due valori positivi:

$$12 = (00001100)_{c2}, \quad 19 = (00010011)_{c2}.$$

Si calcola l'opposto di 19:

$$00010011 \rightarrow 11101100 \rightarrow 11101101.$$

Si esegue quindi:

$$\begin{array}{r} 00001100 \\ +11101101 \\ \hline 11111001 \end{array}$$

Il risultato inizia con 1. Il suo modulo si ottiene invertendo e aggiungendo 1:

$$11111001 \rightarrow 00000110 \rightarrow 00000111 = 7.$$

Pertanto:

$$12 - 19 = -7.$$

2.5.7 L'overflow

Il numero di bit disponibile impone un intervallo limitato. Si verifica un **overflow** quando il risultato matematico è esterno all'intervallo rappresentabile.

Con 4 bit in complemento a 2 si possono rappresentare soltanto valori da -8 a $+7$. Consideriamo:

$$5 + 4 = 9.$$

Il numero 9 non è rappresentabile su 4 bit con segno. L'addizione produce:

$$\begin{array}{r} 0101 \\ +0100 \\ \hline 1001 \end{array}$$

La sequenza $(1001)_{c2}$, interpretata in complemento a 2, vale -7 e non 9. Il risultato è quindi invalido a causa dell'overflow.

Una regola pratica permette di riconoscere molti casi:

- sommando due numeri positivi, un risultato con primo bit 1 indica overflow;
- sommando due numeri negativi, un risultato con primo bit 0 indica overflow;
- sommando numeri di segno diverso non si verifica overflow.

Errore comune: confondere riporto e overflow

Il riporto finale e l'overflow non sono la stessa cosa. In complemento a 2 un riporto esterno può essere eliminato senza che vi sia overflow, come in $7 + (-3)$. L'overflow dipende invece dal fatto che il risultato non rientri nell'intervallo rappresentabile con il numero di bit scelto.

Esercizi: sezione 2.5

1. Indicare l'intervallo rappresentabile in complemento a 2 con 4, 5, 8 e 16 bit.
2. Rappresentare su 8 bit i seguenti numeri positivi e poi calcolarne l'opposto mediante il complemento a 2:
 - a) 1;
 - b) 6;
 - c) 15;
 - d) 37;
 - e) 100.
3. Interpretare in decimale le seguenti sequenze a 8 bit in complemento a 2:
 - a) $(11111111)_{c2}$;
 - b) $(11111010)_{c2}$;
 - c) $(11101100)_{c2}$;

Esercizi: sezione 2.5 (continua)

- d) $(10000000)_{c2}$;
 - e) $(01111111)_{c2}$.
4. Rappresentare su 4 bit in complemento a 2 i seguenti valori: -8 , -6 , -3 , -1 , 0 , 2 , 5 , 7 .
5. Eseguire su 8 bit le somme seguenti e interpretare il risultato:
- a) $5 + (-2)$;
 - b) $9 + (-4)$;
 - c) $3 + (-11)$;
 - d) $(-6) + (-5)$;
 - e) $(-12) + 7$.
6. Trasformare in addizioni ed eseguire su 8 bit:
- a) $14 - 6$;
 - b) $6 - 14$;
 - c) $25 - 9$;
 - d) $9 - 25$;
 - e) $31 - 31$.
7. Per ogni operazione dell'esercizio precedente, indicare l'eventuale riporto esterno e spiegare se deve essere conservato.
8. Stabilire quali delle seguenti operazioni producono overflow su 4 bit in complemento a 2:
- a) $3 + 2$;
 - b) $6 + 3$;
 - c) $-4 + (-3)$;
 - d) $-6 + (-4)$;
 - e) $7 + (-2)$.
9. Individuare e correggere l'errore: per ottenere -9 da $(00001001)_{c2}$ sono stati invertiti i bit ottenendo $(11110110)_{c2}$, ma non è stato aggiunto 1.
10. Spiegare perché la sequenza di bit 11111111 vale $(255)_{10}$ se interpretata senza segno, mentre $(11111111)_{c2}$ rappresenta -1 .
11. Verificare che applicando due volte il complemento a 2 a $(00101101)_{c2}$ si ritorna alla sequenza iniziale.
12. Preparare una procedura in cinque passaggi per calcolare una sottrazione mediante la sola addizione in complemento a 2.

2.6 Il sistema esadecimale

Le sequenze binarie descrivono con precisione i dati elaborati da un computer, ma diventano rapidamente lunghe e difficili da leggere. Il sistema esadecimale permette di rappresentare le stesse informazioni in una forma molto più compatta, mantenendo una corrispondenza immediata con il binario. In questa sezione le conversioni riguardano numeri interi non negativi.

2.6.1 Un sistema di numerazione in base 16

Il **sistema esadecimale** è un sistema di numerazione posizionale in base 16. Sono quindi necessari sedici simboli distinti. Le cifre da 0 a 9 conservano il loro significato abituale, mentre i valori da 10 a 15 vengono rappresentati dalle lettere da A a F.

Definizione: sistema esadecimale

Il sistema esadecimale utilizza sedici cifre: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E e F. La base viene indicata con il pedice 16, per esempio $(2F)_{16}$.

La tabella seguente riunisce le sedici cifre esadecimali e le corrispondenti rappresentazioni decimali e binarie.

Decimale	Binario su 4 bit	Esadecimale
0	0000	0
1	0001	1
2	0010	2
3	0011	3
4	0100	4
5	0101	5
6	0110	6
7	0111	7
8	1000	8
9	1001	9
10	1010	A
11	1011	B
12	1100	C
13	1101	D
14	1110	E
15	1111	F

Tabella 2.5: Corrispondenza tra valori decimali, gruppi binari di 4 bit e cifre esadecimali.

Dopo la cifra F non esiste un nuovo simbolo: come nel sistema decimale dopo 9 si scrive 10, nel sistema esadecimale dopo F si scrive $(10)_{16}$. Questo valore corrisponde a 16 in decimale.

Esempio: contare in esadecimale

La sequenza dei valori attorno al cambio di posizione è:

$\dots, 9, A, B, C, D, E, F, 10, 11, 12, \dots$

Pertanto:

$$(F)_{16} = (15)_{10}, \quad (10)_{16} = (16)_{10}, \quad (11)_{16} = (17)_{10}.$$

Curiosità: il nibble

Un gruppo di 4 bit viene talvolta chiamato *nibble*, termine traducibile in italiano come **semibyte** o **quartetto di bit**. Un nibble può assumere $2^4 = 16$ configurazioni, esattamente quante sono le cifre esadecimali. Per questo motivo ogni cifra esadecimale corrisponde a un gruppo di quattro bit.

2.6.2 Perché si usa il sistema esadecimale

Una cifra esadecimale sostituisce quattro cifre binarie. Una sequenza di 16 bit può quindi essere riscritta usando soltanto quattro cifre esadecimali:

$$(1101011010111110)_2 = (D6BE)_{16}.$$

La forma esadecimale è più breve, riduce la probabilità di trascrivere uno 0 o un 1 nella posizione sbagliata e rende più facile riconoscere gruppi di dati.

Il sistema esadecimale viene utilizzato, per esempio, per rappresentare:

- indirizzi e contenuti della memoria;
- istruzioni e dati a basso livello;
- codici dei colori nelle pagine Web e nei programmi di grafica;
- identificatori e sequenze binarie che sarebbero troppo lunghe da leggere direttamente.

Approfondimento: il prefisso 0x

In molti linguaggi di programmazione e strumenti informatici un numero esadecimale viene segnalato con il prefisso **0x**. Per esempio, **0x2F** indica lo stesso valore di $(2F)_{16}$. Nel capitolo verrà usato soprattutto il pedice 16, perché rende esplicita la base durante i calcoli.

2.6.3 Il valore posizionale delle cifre

Anche il sistema esadecimale è posizionale. Procedendo da destra verso sinistra, le posizioni hanno peso:

$$16^0, \quad 16^1, \quad 16^2, \quad 16^3, \dots$$

Le prime potenze utili sono:

$$16^0 = 1, \quad 16^1 = 16, \quad 16^2 = 256, \quad 16^3 = 4096.$$

Per evitare ambiguità, durante i calcoli le lettere devono essere sostituite con il loro valore decimale. Per esempio, A vale 10, B vale 11 e F vale 15.

Esempio: conversione guidata da esadecimale a decimale

Convertire $(2F)_{16}$ in decimale.

1. Si associano le cifre alle potenze di 16:

$$(2F)_{16} = 2 \cdot 16^1 + F \cdot 16^0.$$

2. Si sostituisce F con 15:

$$2 \cdot 16 + 15 \cdot 1.$$

3. Si eseguono i calcoli:

$$32 + 15 = 47.$$

Quindi:

$$(2F)_{16} = (47)_{10}.$$

Esempio: un numero di tre cifre

Convertire $(1C7)_{16}$ in decimale:

$$(1C7)_{16} = 1 \cdot 16^2 + 12 \cdot 16^1 + 7 \cdot 16^0.$$

$$256 + 192 + 7 = 455.$$

Pertanto:

$$(1C7)_{16} = (455)_{10}.$$

Errore comune: interpretare le lettere come simboli separati

In un numero esadecimale la lettera B non indica una variabile e non deve essere letta come una parola. È una cifra con valore 11. Analogamente, F è una sola cifra con valore 15.

2.6.4 Conversione da decimale a esadecimale

Per convertire un numero decimale intero non negativo in esadecimale si applica un procedimento simile a quello usato per il binario, ma si effettuano divisioni successive per 16.

1. Dividere il numero per 16.
2. Annotare il resto, che sarà compreso tra 0 e 15.
3. Dividere nuovamente per 16 il quoziente ottenuto.

4. Continuare fino a ottenere quoziente 0.
5. Leggere i resti dal basso verso l'alto, sostituendo 10–15 con A–F.

Esempio: conversione guidata di $(190)_{10}$

Si eseguono le divisioni:

$$190 : 16 = 11 \quad \text{con resto } 14,$$

$$11 : 16 = 0 \quad \text{con resto } 11.$$

I resti devono essere letti dal basso verso l'alto. Il valore 11 corrisponde a B e il valore 14 corrisponde a E:

$$(190)_{10} = (BE)_{16}.$$

Il controllo conferma il risultato:

$$B \cdot 16 + E = 11 \cdot 16 + 14 = 190.$$

Esempio: conversione guidata di $(756)_{10}$

$$756 : 16 = 47 \quad \text{con resto } 4,$$

$$47 : 16 = 2 \quad \text{con resto } 15,$$

$$2 : 16 = 0 \quad \text{con resto } 2.$$

Leggendo i resti dal basso verso l'alto e sostituendo 15 con F:

$$(756)_{10} = (2F4)_{16}.$$

Verifica:

$$2 \cdot 256 + 15 \cdot 16 + 4 = 512 + 240 + 4 = 756.$$

2.6.5 Conversione da binario a esadecimale

La conversione tra binario ed esadecimale non richiede divisioni. È sufficiente suddividere la sequenza binaria in gruppi di quattro bit, iniziando da destra, e sostituire ogni gruppo con la cifra esadecimale corrispondente.

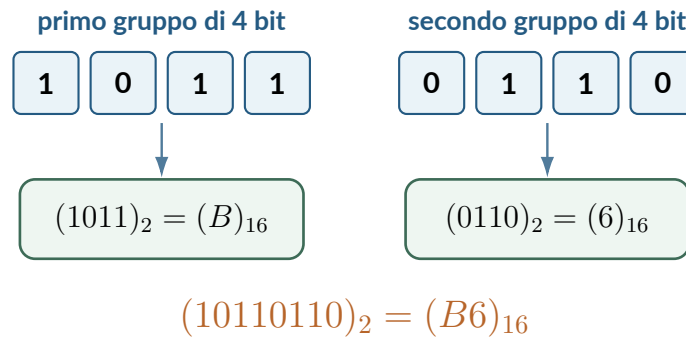


Figura 2.7: Ogni gruppo di quattro bit viene sostituito da una sola cifra esadecimale.

Esempio: conversione diretta

Convertire $(10110110)_2$ in esadecimale.

1011 0110

Dalla tabella di corrispondenza:

$$(1011)_2 = (B)_{16}, \quad (0110)_2 = (6)_{16}.$$

Quindi:

$$(10110110)_2 = (B6)_{16}.$$

Se il numero di bit non è multiplo di quattro, si aggiungono zeri a sinistra fino a completare il primo gruppo.

Esempio: completare il gruppo a sinistra

$(1101011)_2$

La sequenza contiene sette bit. Si aggiunge uno zero a sinistra:

0110 1011.

Pertanto:

$$(1101011)_2 = (6B)_{16}.$$

Errore comune: raggruppare partendo da sinistra

Per i numeri interi, i gruppi devono essere formati partendo dalla cifra più a destra. Gli eventuali zeri necessari vanno aggiunti soltanto a sinistra. Cambiare il punto da cui inizia il raggruppamento modifica il valore del numero.

2.6.6 Conversione da esadecimale a binario

Per effettuare il percorso inverso si sostituisce ogni cifra esadecimale con il gruppo binario di quattro bit indicato nella tabella 2.5.

Esempio: da esadecimale a binario

Convertire $(A7)_{16}$ in binario:

$$(A)_{16} = (1010)_2, \quad (7)_{16} = (0111)_2.$$

Unendo i due gruppi:

$$(A7)_{16} = (10100111)_2.$$

Esempio: conservare quattro bit per ogni cifra

Convertire $(3F2)_{16}$:

$$3 \rightarrow 0011, \quad F \rightarrow 1111, \quad 2 \rightarrow 0010.$$

Quindi:

$$(3F2)_{16} = (001111110010)_2.$$

Lo zero iniziale dell'intera sequenza può essere omissso ottenendo $(1111110010)_2$, ma gli zeri interni ai gruppi non devono essere eliminati durante la conversione.

2.6.7 Conversioni tra le tre basi

Binario, decimale ed esadecimale rappresentano gli stessi valori usando simboli differenti. La scelta del percorso più conveniente riduce i calcoli:

- tra binario ed esadecimale conviene usare i gruppi di quattro bit;
- da binario o esadecimale a decimale si usano i pesi posizionali;
- da decimale a binario o esadecimale si usano le divisioni successive per la base di arrivo.

Esempio: conversione completa

Convertire $(D3)_{16}$ in binario e in decimale.

Per il binario:

$$(D3)_{16} = (1101\ 0011)_2.$$

Per il decimale:

$$(D3)_{16} = 13 \cdot 16 + 3 = (211)_{10}.$$

Il controllo mediante i pesi binari produce lo stesso valore:

$$(11010011)_2 = 128 + 64 + 16 + 2 + 1 = (211)_{10}.$$

Esercizi: sezione 2.6

1. Scrivere in ordine crescente tutte le cifre del sistema esadecimale e indicare il valore decimale delle lettere A–F.
2. Completare la tabella senza consultare la tabella 2.5:

Decimale	Binario su 4 bit	Esadecimale
5	_____	_____
10	_____	_____
13	_____	_____
15	_____	_____

3. Convertire in decimale: $(18)_{16}$, $(2A)_{16}$, $(7F)_{16}$, $(B4)_{16}$, $(10C)_{16}$ e $(3E7)_{16}$.
4. Convertire in esadecimale mediante divisioni successive: $(26)_{10}$, $(63)_{10}$, $(128)_{10}$, $(255)_{10}$, $(300)_{10}$ e $(1024)_{10}$.
5. Convertire direttamente in esadecimale: $(1010)_2$, $(1111)_2$, $(10011100)_2$, $(11101011)_2$ e $(10110101101)_2$.
6. Convertire direttamente in binario: $(5)_{16}$, $(C)_{16}$, $(2D)_{16}$, $(A0)_{16}$, $(7F3)_{16}$ e $(BEEF)_{16}$.
7. Aggiungere gli zeri iniziali necessari e convertire: $(101)_2$, $(11011)_2$, $(1001011)_2$ e $(111001101)_2$.
8. Convertire $(156)_{10}$ prima in binario e poi in esadecimale. Verificare il risultato convertendo direttamente il numero decimale in esadecimale.
9. Convertire $(9C)_{16}$ in binario e poi in decimale. Verificare il valore usando direttamente i pesi di 16.
10. Individuare e correggere l'errore nel procedimento $1011 \mid 01 \rightarrow B1$ usato per convertire $(101101)_2$ in esadecimale. I bit sono stati raggruppati da sinistra e il gruppo finale incompleto è stato letto come una cifra autonoma. Raggruppare correttamente i bit e verificare il risultato in decimale.
11. Spiegare perché due cifre esadecimali possono rappresentare esattamente un byte.
12. Ordinare dal più piccolo al più grande, dopo averli convertiti in una base comune: $(101101)_2$, $(35)_{10}$, $(2F)_{16}$ e $(111000)_2$.

2.7 I colori in formato esadecimale

Il sistema esadecimale trova un'applicazione quotidiana nella rappresentazione dei colori utilizzati da schermi, pagine Web e programmi di grafica. Un codice breve come #3A7FD5 contiene le intensità delle tre componenti luminose rossa, verde e blu.

2.7.1 Il modello RGB

Gli schermi producono i colori combinando luce rossa, verde e blu. Questo metodo viene chiamato **modello RGB**, dalle iniziali inglesi di *Red*, *Green* e *Blue*.

Ogni componente può essere assente, presente con intensità intermedia oppure alla massima intensità. Nella rappresentazione più comune ogni canale utilizza 8 bit e può quindi assumere:

$$2^8 = 256$$

valori, compresi tra 0 e 255.

Definizione: colore RGB a 24 bit

Un colore RGB a 24 bit utilizza 8 bit per la componente rossa, 8 bit per la componente verde e 8 bit per la componente blu:

$$8 + 8 + 8 = 24 \text{ bit.}$$

Il numero teorico delle combinazioni possibili è:

$$256 \cdot 256 \cdot 256 = 2^{24} = 16\,777\,216.$$

Quando tutte le componenti valgono 0 non viene emessa luce e si ottiene il nero. Quando tutte valgono 255 si ottiene il bianco. Le combinazioni intermedie producono gli altri colori.

Colore	Rosso	Verde	Blu
Nero	0	0	0
Rosso	255	0	0
Verde	0	255	0
Blu	0	0	255
Giallo	255	255	0
Ciano	0	255	255
Magenta	255	0	255
Bianco	255	255	255

Tabella 2.6: Alcune combinazioni fondamentali del modello RGB.

Approfondimento: RGB e stampa

Il modello RGB descrive la combinazione di luci ed è adatto agli schermi. La stampa utilizza normalmente un modello differente, basato sugli inchiostri ciano, magenta, giallo e nero. Per questo motivo un colore visualizzato sul monitor può non apparire perfettamente identico una volta stampato.

2.7.2 La notazione #RRGGBB

Poiché ciascun canale contiene 8 bit, può essere rappresentato esattamente con due cifre esadecimali:

$$(00000000)_2 = (00)_{16}, \quad (11111111)_2 = (FF)_{16}.$$

Un codice colore è quindi formato dal simbolo # seguito da sei cifre esadecimali:

#RRGGBB.

Le prime due cifre indicano il rosso, le due centrali il verde e le ultime due il blu.

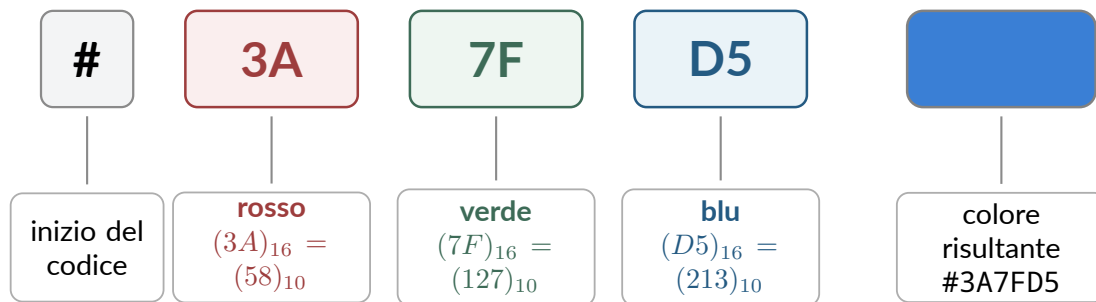


Figura 2.8: Struttura del codice #3A7FD5: ogni coppia esadecimale descrive un canale RGB.

Colore	RGB decimale	Codice	Campione
Nero	(0, 0, 0)	#000000	
Bianco	(255, 255, 255)	#FFFFFF	
Rosso	(255, 0, 0)	#FF0000	
Verde	(0, 255, 0)	#00FF00	
Blu	(0, 0, 255)	#0000FF	
Giallo	(255, 255, 0)	#FFFF00	
Ciano	(0, 255, 255)	#00FFFF	
Magenta	(255, 0, 255)	#FF00FF	
Grigio	(128, 128, 128)	#808080	

Tabella 2.7: Codici esadecimali di alcuni colori fondamentali.

2.7.3 Interpretare un codice colore

Per interpretare un codice si separano le sei cifre in tre coppie e si converte ogni coppia da esadecimale a decimale.

Esempio: interpretazione guidata di #D2691E

Si separano i tre canali:

D2 69 1E.

Si converte ciascun valore:

$$(D2)_{16} = 13 \cdot 16 + 2 = 210,$$

$$(69)_{16} = 6 \cdot 16 + 9 = 105,$$

$$(1E)_{16} = 1 \cdot 16 + 14 = 30.$$

Il codice corrisponde quindi a:

$$RGB(210, 105, 30).$$

La componente rossa è dominante, quella verde è intermedia e quella blu è bassa: il risultato è una tonalità calda tendente all'arancione-bruno.

Errore comune: confondere coppie e singole cifre

Nel codice #FF8000 i canali non sono F, F e 8. Le sei cifre devono essere divise in tre coppie: FF per il rosso, 80 per il verde e 00 per il blu.

2.7.4 Da valori RGB decimali a codice esadecimale

Per costruire il codice esadecimale a partire dai valori RGB si converte separatamente ogni componente. Ogni risultato deve essere scritto usando esattamente due cifre. Se il valore esadecimale contiene una sola cifra, si aggiunge uno zero a sinistra.

Esempio: conversione guidata di $RGB(45, 160, 210)$

Si converte la componente rossa:

$$(45)_{10} = (2D)_{16}.$$

Si converte la componente verde:

$$(160)_{10} = (A0)_{16}.$$

Si converte la componente blu:

$$(210)_{10} = (D2)_{16}.$$

Unendo le tre coppie nell'ordine rosso, verde e blu:

$$RGB(45, 160, 210) = \#2DA0D2.$$

Esempio: aggiungere lo zero iniziale

Convertire $RGB(7, 32, 15)$.

$$(7)_{10} = (7)_{16} = (07)_{16},$$

$$(32)_{10} = (20)_{16},$$

$$(15)_{10} = (F)_{16} = (0F)_{16}.$$

Il codice completo è:

#07200F.

Senza gli zeri iniziali non sarebbe più possibile distinguere correttamente le tre componenti.

2.7.5 Da codice esadecimale a valori RGB

Il procedimento inverso consiste nel separare il codice in tre coppie e convertire ciascuna coppia in decimale.

Esempio: conversione guidata di #7BC043

$$(7B)_{16} = 7 \cdot 16 + 11 = 123,$$

$$(C0)_{16} = 12 \cdot 16 + 0 = 192,$$

$$(43)_{16} = 4 \cdot 16 + 3 = 67.$$

Pertanto:

$$\#7BC043 = RGB(123, 192, 67).$$

2.7.6 Prevedere e modificare un colore

Il codice esadecimale permette di ragionare sulle componenti prima ancora di osservare il campione:

- un valore vicino a 00 indica una componente poco intensa;
- un valore vicino a FF indica una componente molto intensa;
- se i tre canali hanno lo stesso valore si ottiene una tonalità di grigio;
- aumentando tutti e tre i valori il colore tende a diventare più chiaro;
- diminuendo tutti e tre i valori il colore tende a diventare più scuro;
- aumentando un solo canale si rafforza la presenza della componente corrispondente.

Esempio: una scala di grigi

Nei codici seguenti le tre coppie sono sempre uguali:

#202020, #808080, #E0E0E0.

Il primo colore è un grigio molto scuro, il secondo un grigio intermedio e il terzo un grigio chiaro.

Curiosità: forme abbreviate e trasparenza

In alcuni contesti il codice #RGB è una forma abbreviata di #RRGGBB: per esempio #3A7 equivale a #33AA77. Esistono inoltre formati con una quarta componente, chiamata canale alfa, che descrive la trasparenza. Queste notazioni non verranno utilizzate negli esercizi principali.

2.7.7 Applicazione facoltativa: costruire una tavolozza

La conoscenza dei codici esadecimali può essere applicata alla progettazione grafica di una semplice interfaccia, di un'etichetta o di una scheda informativa. Una tavolozza equilibrata deve contenere colori leggibili e coerenti con il messaggio da comunicare.

Esempio: tavolozza per una scheda informativa

Si può scegliere:

- un verde scuro per i titoli, per esempio #2F5D50;
- un verde chiaro per gli sfondi, per esempio #DCE9E2;
- un arancione per mettere in evidenza avvisi, per esempio #D9822B;
- un grigio molto scuro per il testo, per esempio #303030;
- il bianco per le aree principali, #FFFFFF.

Per ogni codice si possono convertire le tre coppie in valori RGB e verificare quali componenti risultano dominanti.

Esercizi: sezione 2.7

1. Spiegare il significato delle lettere R, G e B e indicare quanti bit vengono usati per ciascun canale in un colore RGB a 24 bit.
2. Indicare il colore prodotto dalle seguenti combinazioni RGB fondamentali: (0, 0, 0), (255, 0, 0), (0, 255, 255), (255, 255, 255).
3. Separare nei tre canali i codici: #F08020, #12ABEF, #808080 e #00FF7F.
4. Convertire in valori RGB decimali: #FF8000, #3366CC, #7F7F7F, #00A040 e #C81464.
5. Convertire in codice esadecimale: $RGB(255, 255, 0)$, $RGB(128, 0, 255)$, $RGB(16, 32, 64)$, $RGB(5, 10, 15)$ e $RGB(200, 150, 100)$.
6. Spiegare perché $RGB(7, 32, 15)$ deve essere scritto #07200F e non #720F.

Esercizi: sezione 2.7 (continua)

7. Completare i codici:
 - a) rosso massimo, verde assente, blu massimo: #_____;
 - b) rosso assente, verde massimo, blu assente: #_____;
 - c) tutti i canali a 128: #_____.
8. Stabilire quali codici rappresentano tonalità di grigio: #303030, #30A030, #BEBEBE, #101112.
9. Tra #204060 e #80A0C0, prevedere quale colore apparirà generalmente più chiaro e motivare la risposta osservando i tre canali.
10. Modificare #804020 per ottenere:
 - a) una componente rossa più intensa;
 - b) una componente blu più intensa;
 - c) una tonalità complessivamente più scura.
11. Individuare e correggere l'errore nell'affermazione: "Nel codice #1AF040, le componenti sono 1, A e F".
12. Progettare una tavolozza di quattro colori per una presentazione, un'etichetta di un prodotto agricolo oppure una scheda di laboratorio di chimica. Riportare per ogni colore il codice esadecimale, i valori RGB decimali e la funzione prevista.

2.8 Quanto pesa una fotografia digitale?

Una fotografia visualizzata sullo schermo appare come un'immagine continua, ma in realtà è formata da una grande quantità di piccoli elementi disposti in una griglia. Per comprendere perché un file fotografico occupa alcuni megabyte è necessario collegare quattro concetti già incontrati nel capitolo: il numero di pixel, la quantità di bit usata per descrivere ciascun pixel, il byte e i suoi multipli.

Il calcolo presentato in questa sezione permette di stimare la **dimensione teorica dei dati dell'immagine non compressa**. Il peso effettivo del file può essere differente, perché i formati fotografici applicano tecniche di compressione e aggiungono altre informazioni.

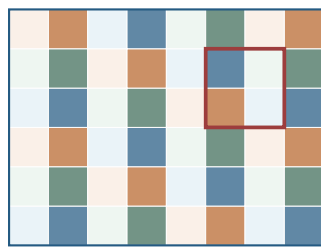
2.8.1 Il pixel: il più piccolo elemento dell'immagine

Il termine **pixel** deriva dall'espressione inglese *picture element*, cioè "elemento dell'immagine". Un'immagine digitale di tipo raster è composta da una griglia di pixel. Ciascun pixel occupa una posizione precisa e contiene un valore che ne descrive il colore.

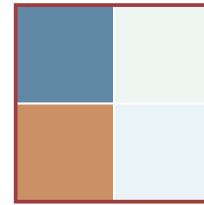
Definizione: pixel

Il **pixel** è il più piccolo elemento indirizzabile di un'immagine digitale raster. L'insieme ordinato dei pixel forma la griglia dell'immagine.

IMMAGINE DIGITALE

Selezione di 2×2 pixel

DETTAGLIO INGRANDITO



Un pixel

Ingrandimento degli
stessi 4 pixel

Figura 2.9: Un'immagine raster è composta da una griglia di pixel. Ingrandendo una porzione diventano visibili i singoli elementi.

Se l'immagine viene ingrandita oltre un certo limite, i singoli pixel possono diventare visibili come piccoli quadrati. Questo effetto viene spesso chiamato **sgranatura** o **pixelatura**.

Errore comune: il pixel non ha una dimensione fisica sempre uguale

Un pixel non misura necessariamente un millimetro o una quantità fissa di centimetri. La sua dimensione visibile dipende dallo schermo, dalla stampa e dal livello di ingrandimento. Nella fotografia digitale il pixel viene soprattutto considerato come una posizione della griglia dell'immagine.

2.8.2 La risoluzione e il numero totale di pixel

La **risoluzione** di un'immagine indica il numero di pixel presenti lungo la larghezza e lungo l'altezza. Viene normalmente scritta nella forma:

larghezza $\times$ altezza.

Per esempio, una fotografia con risoluzione 4000×3000 contiene 4000 pixel in ogni riga e 3000 pixel in ogni colonna. Il numero totale di pixel si ottiene moltiplicando le due dimensioni:

$$4000 \cdot 3000 = 12\,000\,000 \text{ pixel.}$$

Definizione: megapixel

Un **megapixel**, abbreviato in MP, corrisponde a un milione di pixel. Il valore in megapixel si ottiene dividendo il numero totale di pixel per 1 000 000.

La fotografia dell'esempio possiede quindi:

$$\frac{12\,000\,000}{1\,000\,000} = 12 \text{ MP.}$$

MP e MB misurano quantità diverse: i primi esprimono il numero di pixel, i secondi la quantità di dati. Per i MB vale la convenzione della sezione 2.1.

Esempio: calcolare i megapixel

Un'immagine ha una risoluzione di 6000×4000 pixel.

$$6000 \cdot 4000 = 24\,000\,000 \text{ pixel}$$

$$\frac{24\,000\,000}{1\,000\,000} = 24 \text{ MP.}$$

La fotografia contiene quindi 24 milioni di pixel ed è comunemente indicata come immagine da 24 megapixel.

Curiosità: risoluzione e qualità

Un numero maggiore di pixel permette di conservare più dettagli e di ritagliare o stampare l'immagine a dimensioni maggiori. Tuttavia la qualità finale dipende anche dall'obiettivo, dal sensore, dalla messa a fuoco, dal rumore, dalla compressione e dalle condizioni di ripresa. Un numero elevato di megapixel, da solo, non garantisce una fotografia migliore.

2.8.3 Quanti bit servono per descrivere un pixel?

Il numero di bit utilizzato per rappresentare il colore di un pixel viene chiamato **profondità di colore**. Maggiore è il numero di bit per pixel, maggiore è il numero di colori differenti che possono essere rappresentati.

Tipo di immagine	Bit/pixel	Colori o livelli	Descrizione
Binaria	1	2	Ogni pixel può assumere soltanto due stati, per esempio bianco o nero.
Scala di grigi	8	256	Ogni pixel descrive uno tra 256 livelli di luminosità.
RGB	24	2^{24}	Ogni pixel usa 8 bit per il rosso, 8 per il verde e 8 per il blu.
RGBA	32	2^{24} colori	Ai 24 bit RGB si aggiungono 8 bit per la trasparenza.

Tabella 2.8: Alcune profondità di colore comunemente utilizzate.

Nel caso più comune trattato negli esercizi, una fotografia RGB utilizza:

$$8 \text{ bit per il rosso} + 8 \text{ bit per il verde} + 8 \text{ bit per il blu} = 24 \text{ bit per pixel.}$$

Poiché un byte contiene 8 bit:

$$24 \text{ bit} = 3 \text{ byte.}$$

Pertanto, nell'ipotesi RGB a 24 bit, ogni pixel richiede teoricamente 3 byte.

Approfondimento: quanti colori si possono rappresentare?

Con 24 bit si ottengono:

$$2^{24} = 16\,777\,216$$

combinazioni possibili. Questo valore deriva da 256 livelli per ciascuno dei tre canali RGB:

$$256 \cdot 256 \cdot 256 = 16\,777\,216.$$

2.8.4 Calcolare la dimensione teorica dell'immagine

Per un'immagine non compressa, la quantità di dati necessaria per descrivere tutti i pixel può essere stimata con la formula:

$$\text{dimensione in bit} = \text{larghezza} \cdot \text{altezza} \cdot \text{bit per pixel}.$$

In alternativa, se si conoscono i byte necessari per ogni pixel:

$$\text{dimensione in byte} = \text{numero di pixel} \cdot \text{byte per pixel}.$$

Per esprimere il risultato in megabyte si applica la convenzione adottata nel libro:

$$1 \text{ MB} = 1024 \cdot 1024 \text{ B} = 1\,048\,576 \text{ B}.$$

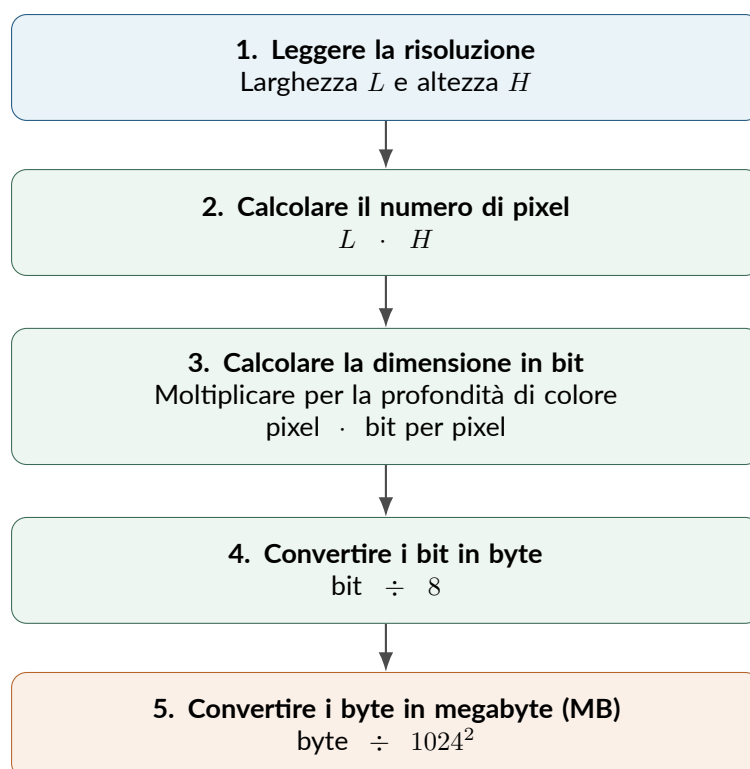


Figura 2.10: Procedimento generale per calcolare la dimensione teorica di un'immagine digitale.

Esempio: fotografia da 12 megapixel

Si consideri una fotografia RGB a 24 bit con risoluzione 4000×3000 pixel.

1. Calcolo del numero di pixel

$$4000 \cdot 3000 = 12\,000\,000 \text{ pixel.}$$

2. Calcolo della dimensione in bit

$$12\,000\,000 \cdot 24 = 288\,000\,000 \text{ bit.}$$

3. Conversione in byte

$$288\,000\,000 \div 8 = 36\,000\,000 \text{ B.}$$

4. Conversione in megabyte

$$36\,000\,000 \div 1024 \div 1024 \approx 34,33 \text{ MB.}$$

La dimensione teorica dei soli dati RGB non compressi è quindi circa **34,33 MB**.

Errore comune: saltare un passaggio da B a MB

Per convertire i byte in MB non basta dividere una sola volta per 1024: il primo passaggio porta ai KB, il secondo ai MB. Per esempio:

$$36\,000\,000 \text{ B} \div 1024 = 35\,156,25 \text{ KB,}$$

$$35\,156,25 \text{ KB} \div 1024 \approx 34,33 \text{ MB.}$$

Si può ottenere lo stesso risultato dividendo direttamente il numero di byte per 1024^2 .

Esempio: immagine Full HD

Un'immagine RGB a 24 bit ha risoluzione 1920×1080 pixel.

$$1920 \cdot 1080 = 2\,073\,600 \text{ pixel.}$$

Ogni pixel richiede 3 byte:

$$2\,073\,600 \cdot 3 = 6\,220\,800 \text{ B.}$$

Si converte in megabyte:

$$6\,220\,800 \div 1024 \div 1024 \approx 5,93 \text{ MB.}$$

La dimensione teorica non compressa è quindi circa **5,93 MB**.

Risoluzione	Pixel totali	Megapixel	RGB 24 bit
800 × 600	480 000	0,48 MP	1,37 MB
1920 × 1080	2 073 600	2,07 MP	5,93 MB
4000 × 3000	12 000 000	12 MP	34,33 MB
6000 × 4000	24 000 000	24 MP	68,66 MB

Tabella 2.9: Dimensione teorica dei dati RGB non compressi per alcune risoluzioni. I valori sono arrotondati.

2.8.5 Perché il file reale può essere più piccolo?

Il valore calcolato finora descrive la quantità teorica di dati necessaria per memorizzare tutti i pixel senza considerare la compressione. Una fotografia salvata sul telefono o sulla fotocamera può occupare molto meno spazio perché il formato del file cerca di rappresentare gli stessi contenuti usando una quantità inferiore di dati.

Definizione: compressione

La **compressione** è un insieme di tecniche che riduce la quantità di dati necessaria per memorizzare o trasmettere un file.

Si distinguono due categorie principali:

- nella **compressione senza perdita** i dati originali possono essere ricostruiti esattamente;
- nella **compressione con perdita** alcune informazioni vengono eliminate per ottenere file più piccoli, cercando di mantenere una qualità visiva accettabile.

Formato	Compressione	Perdita	Uso tipico
JPEG	Sì	Sì	Fotografie, quando è importante ridurre la dimensione del file.
PNG	Sì	No	Grafica, schermate, loghi e immagini che richiedono conservazione esatta dei dati.
BMP	Spesso assente o limitata	No	Immagini semplici e casi in cui la compatibilità è più importante dello spazio.
RAW	Dipende dal formato	Generalmente no	Conservazione di molti dati acquisiti dal sensore per la successiva elaborazione.

Tabella 2.10: Confronto generale tra alcuni formati di immagine.

La dimensione reale dipende inoltre da numerosi fattori:

- qualità di compressione selezionata;
- quantità di dettagli presenti nella scena;
- rumore e variazioni di colore;
- formato utilizzato;
- profondità di colore;
- presenza di miniature, profili colore e metadati;
- eventuale trasparenza.

Una fotografia con aree uniformi, come un cielo limpido, può essere compressa più facilmente di una fotografia ricca di foglie, erba, capelli o altri dettagli minuti.

Esempio: confrontare dimensione teorica e reale

La fotografia da 4000×3000 pixel dell'esempio precedente ha una dimensione teorica RGB di circa 34,33 MB. Il file JPEG ha invece una dimensione di 4,8 MB, espressa nella stessa unità per rendere confrontabili i valori.

Il rapporto tra i due valori è:

$$34,33 \div 4,8 \approx 7,15.$$

Il file reale è quindi circa 7,15 volte più piccolo della rappresentazione RGB non compressa. Questo confronto non indica da solo la qualità dell'immagine, ma mostra l'effetto della compressione.

Approfondimento: la formula è una stima didattica

Un file reale contiene anche intestazioni, metadati e informazioni necessarie per interpretare i dati. Alcuni formati possono inoltre usare una profondità di colore diversa da 24 bit oppure organizzare i dati in modo differente. La formula presentata resta comunque un metodo efficace per comprendere da quali fattori dipende l'ordine di grandezza della dimensione di un'immagine.

2.8.6 Ridimensionare una fotografia

Ridurre larghezza e altezza diminuisce il numero totale di pixel e quindi la quantità di dati da memorizzare. È importante osservare che dimezzare entrambe le dimensioni non dimezza il numero di pixel: lo riduce a un quarto.

Esempio: dimezzare larghezza e altezza

Una fotografia misura 4000×3000 pixel e contiene:

$$4000 \cdot 3000 = 12\,000\,000 \text{ pixel.}$$

Dimezzando entrambe le dimensioni si ottiene 2000×1500 :

$$2000 \cdot 1500 = 3\,000\,000 \text{ pixel.}$$

Il numero di pixel è passato da 12 milioni a 3 milioni, cioè a un quarto del valore iniziale. Anche la dimensione teorica RGB si riduce da circa 34,33 MB a circa 8,58 MB.

Errore comune: modificare la risoluzione senza mantenere le proporzioni

Se larghezza e altezza vengono cambiate con fattori differenti, la fotografia può risultare deformata. Per conservare le proporzioni originali, entrambe le dimensioni devono essere moltiplicate o divise per lo stesso fattore.

Esercizi: sezione 2.8

1. Definire i termini *pixel*, *risoluzione*, *megapixel* e *profondità di colore*.
2. Calcolare il numero totale di pixel e i megapixel delle immagini seguenti:
 - a) 1280×720 ;
 - b) 1920×1080 ;
 - c) 3000×2000 ;
 - d) 6000×4000 .
3. Calcolare la dimensione teorica in byte, KB e MB di un'immagine RGB a 24 bit con risoluzione 1000×1000 pixel.
4. Calcolare la dimensione teorica in MB di un'immagine 1920×1080 :
 - a) binaria a 1 bit per pixel;
 - b) in scala di grigi a 8 bit per pixel;
 - c) RGB a 24 bit per pixel;
 - d) RGBA a 32 bit per pixel.
5. Una fotografia 4000×3000 è salvata in JPEG e occupa 5,2 MB. Calcolare la dimensione teorica RGB a 24 bit e confrontarla con il peso reale.
6. Una fotografia 6000×4000 viene ridimensionata a 3000×2000 . Di quale fattore diminuisce il numero di pixel? Calcolare anche le due dimensioni teoriche RGB.
7. Un'immagine contiene 8 000 000 pixel e utilizza 24 bit per pixel. Calcolare la dimensione teorica in MB.
8. Una fotografia RGB non compressa occupa teoricamente 18 000 000 byte. Determinare quanti pixel contiene e, sapendo che è larga 3000 pixel, ricavare l'altezza.
9. Spiegare perché due fotografie con la stessa risoluzione e lo stesso formato JPEG possono avere dimensioni differenti.
10. Indicare quale formato, tra JPEG e PNG, è generalmente più adatto:
 - a) a una fotografia destinata al web;
 - b) a un logo con aree di colore uniforme;
 - c) a un'immagine che deve conservare esattamente ogni pixel.
11. Individuare l'errore: "Una fotografia da 12 megapixel occupa sempre esattamente 12 MB".

Esercizi riepilogativi

Il linguaggio digitale

Capitolo 2

1. Misurare l'informazione

Completa senza usare la calcolatrice. Usa le equivalenze presentate nel capitolo.

$$\begin{array}{ll} 8 \text{ b} = \underline{\hspace{2cm}} \text{ B} & 2048 \text{ B} = \underline{\hspace{2cm}} \text{ KB} \\ 4 \text{ KB} = \underline{\hspace{2cm}} \text{ B} & 3 \text{ MB} = \underline{\hspace{2cm}} \text{ KB} \\ 2 \text{ GB} = \underline{\hspace{2cm}} \text{ MB} & 8192 \text{ b} = \underline{\hspace{2cm}} \text{ B} \end{array}$$

2. Conversioni tra le basi

Completa la tabella indicando tutti i passaggi sul quaderno.

Decimale	Binario	Esadecimale
13	<u> </u>	<u> </u>
<u> </u>	0010 1101	<u> </u>
175	<u> </u>	<u> </u>
<u> </u>	1111 0000	<u> </u>
<u> </u>	<u> </u>	7B

3. Operazioni binarie

Esegui le operazioni e controlla ogni risultato convertendo gli operandi e il risultato in decimale.

Addizioni

$$\begin{array}{r} 101101_2 \\ + 011011_2 \\ \hline \end{array}$$

$$\begin{array}{r} 110010_2 \\ + 001111_2 \\ \hline \end{array}$$

$$\begin{array}{r} 1010111_2 \\ + 0101101_2 \\ \hline \end{array}$$

Sottrazioni

$$\begin{array}{r} 110010_2 \\ - 001101_2 \\ \hline \end{array}$$

$$\begin{array}{r} 101101_2 \\ - 010011_2 \\ \hline \end{array}$$

$$\begin{array}{r} 111000_2 \\ - 001111_2 \\ \hline \end{array}$$

Moltiplicazioni

$$\begin{array}{r} 1011_2 \\ \times 101_2 \\ \hline \end{array}$$

$$\begin{array}{r} 1101_2 \\ \times 11_2 \\ \hline \end{array}$$

$$\begin{array}{r} 1001_2 \\ \times 110_2 \\ \hline \end{array}$$

Esercizi riepilogativi**Capitolo 2****Numeri binari con la virgola**

Ripassa la sezione 2.3 e svolgi i passaggi sul quaderno. Usa $=$ per i risultati esatti e $\approx$ per quelli approssimati; non arrotondare i risultati intermedi.

4. Conversioni esatte con la virgola

1. Converti in decimale $(0,01101)_2$ e $(101,1101)_2$ usando i pesi delle cifre.
2. Converti in binario 0,3125, 3,875 e 10,6875. Separa parte intera e parte frazionaria; applica a quest'ultima le moltiplicazioni successive per 2. Controlla i risultati riconvertendoli in decimale.
3. Confronta $(0,1)_2$, $(0,1000)_2$ e $(0,1)_{10}$. Quali rappresentano lo stesso valore? Spiega il ruolo della base e degli zeri finali.

5. Rappresentazioni finite e periodiche

1. Stabilisci quali valori hanno una rappresentazione binaria finita: 0,45; 0,0625; 0,35; $1/6$; $7/32$. Riduci ogni frazione ai minimi termini e motiva la risposta osservando il denominatore.
2. Converti 0,4 in binario fino a ritrovare una parte frazionaria già usata. Individua il periodo e spiega perché le moltiplicazioni successive ripeteranno le stesse cifre.

6. Arrotondamenti e casi particolari

Conserva **10 cifre binarie dopo la virgola**, calcolando l'undicesima per decidere l'arrotondamento. Nei casi esattamente a metà scegli il valore con **maggiore valore assoluto**, come nella sezione 2.3.

1. Converti 0,15 calcolando le prime 11 cifre binarie. Scrivi sia il risultato troncato a 10 cifre sia quello arrotondato, mostrando gli eventuali riporti.
2. Arrotonda 2,9999 a 10 bit frazionari. Mostra il riporto e indica che cosa accade alla parte intera.
3. Applica lo stesso arrotondamento a $1/2048$, 0,0001 e $-0,15$. Riconosci il caso di parità, il valore che diventa zero e il caso in cui devi lavorare sul valore assoluto.
4. Calcola l'errore assoluto del risultato arrotondato ottenuto per 0,15 e confrontalo con il limite massimo $1/2048$.

Esercizi riepilogativi

Numeri con segno e colori digitali

Capitolo 2**7. Numeri con segno e complemento a 2**

Usa rappresentazioni a 8 bit.

1. Rappresenta $+37$ e -37 in complemento a 2.
2. Interpreta il numero `1110 1011` come valore con segno.
3. Esegui $75 + 68$ e stabilisci se si verifica overflow. Motiva la risposta.

8. Dal modello RGB al codice esadecimale

1. Converti $(255, 128, 0)$ nel codice `#RRGGBB`.
2. Ricava i valori RGB decimali dal codice `#3A7FC0`.
3. Senza effettuare calcoli completi, indica quale colore dovrebbe risultare da `#00FFFF` e spiega il ragionamento.
4. Modifica `#804020` per ottenere un colore con una componente blu più intensa, lasciando invariate rosso e verde.

Esercizi riepilogativi**Capitolo 2**

Dimensioni e archiviazione delle immagini

9. Quanto pesa una fotografia?

Una fotografia non compressa misura 4000×3000 pixel e usa 24 bit per pixel.

1. Calcola il numero totale di pixel.
2. Calcola la dimensione teorica in bit e in byte.
3. Converti il risultato in MB.
4. La fotografia viene ridimensionata a 2000×1500 pixel, mantenendo 24 bit per pixel. Calcola la nuova dimensione teorica e confrontala con quella iniziale.
5. Il file JPEG reale occupa soltanto 4,8 MB. Spiega perché il valore è molto inferiore alla dimensione teorica.

10. Scegliere lo spazio di archiviazione

Una classe produce 280 fotografie da 7,5 MB ciascuna e 12 video da 420 MB ciascuno.

1. Stima lo spazio totale necessario in MB e in GB.
2. Una memoria da 8 GB è sufficiente? Considera che una parte dello spazio non è disponibile per i file dell'utente.
3. Indica almeno due motivi per cui la stima può differire dallo spazio realmente occupato.

11. Spiega con parole tue

Qual è la differenza tra **risoluzione**, **profondità di colore** e **compressione** di un'immagine?

Parte III

Reti e Internet

3. Reti e Internet

Quando si invia una fotografia, si apre una pagina Web o si ascolta musica in streaming, l'azione appare immediata. In realtà i dati attraversano dispositivi, collegamenti, reti di operatori e centri di elaborazione distribuiti anche a grande distanza. Comprendere questa infrastruttura permette di usare Internet in modo più consapevole e di riconoscere il ruolo svolto da hardware, indirizzi e protocolli.

Il capitolo inizia seguendo il viaggio di una fotografia inviata da Bob, collegato alla rete Wi-Fi di casa, ad Alice, connessa al mare attraverso la rete mobile. A partire da questo esempio verranno studiati i mezzi trasmissivi, i dispositivi di indirizzamento, le topologie e le diverse estensioni delle reti. La parte successiva sarà dedicata agli indirizzi IP, ai protocolli, alla navigazione Web, al DNS e ai modelli client-server e peer-to-peer.

Conoscenze	Abilità	Competenze
• mezzi trasmissivi in rame, fibra e wireless; • hub, switch e router; • topologie e tipologie di rete; • indirizzi IPv4, IP privati, IP pubblici e IPv6; • protocolli di rete e di Internet; • URL, HTTP, HTTPS e DNS; • modelli client-server e peer-to-peer.	• ricostruire il percorso seguito da un dato; • riconoscere mezzi trasmissivi e dispositivi di indirizzamento; • confrontare vantaggi e limiti delle tecnologie; • classificare una rete per topologia ed estensione; • interpretare un indirizzo IP e un URL; • descrivere le fasi della navigazione Web.	• comprendere la complessità dell'infrastruttura di Internet; • scegliere collegamenti adatti a semplici situazioni reali; • spiegare come dispositivi e protocolli collaborano; • analizzare problemi elementari di connettività; • utilizzare i servizi online con maggiore consapevolezza.

3.1 Il viaggio di una fotografia

Bob si trova a casa e vuole mostrare ad Alice una fotografia appena scattata. Apre WhatsApp, seleziona l'immagine e preme il pulsante di invio. Alice si trova al mare e utilizza la connessione 5G del proprio smartphone. Dopo pochi istanti la fotografia compare sul suo schermo.

L'operazione sembra semplicissima: un tocco sullo smartphone di Bob e una notifica su quello di Alice. La fotografia, tuttavia, non passa direttamente da un telefono all'altro. I dati attraversano una catena di dispositivi e reti che l'utente normalmente non vede.

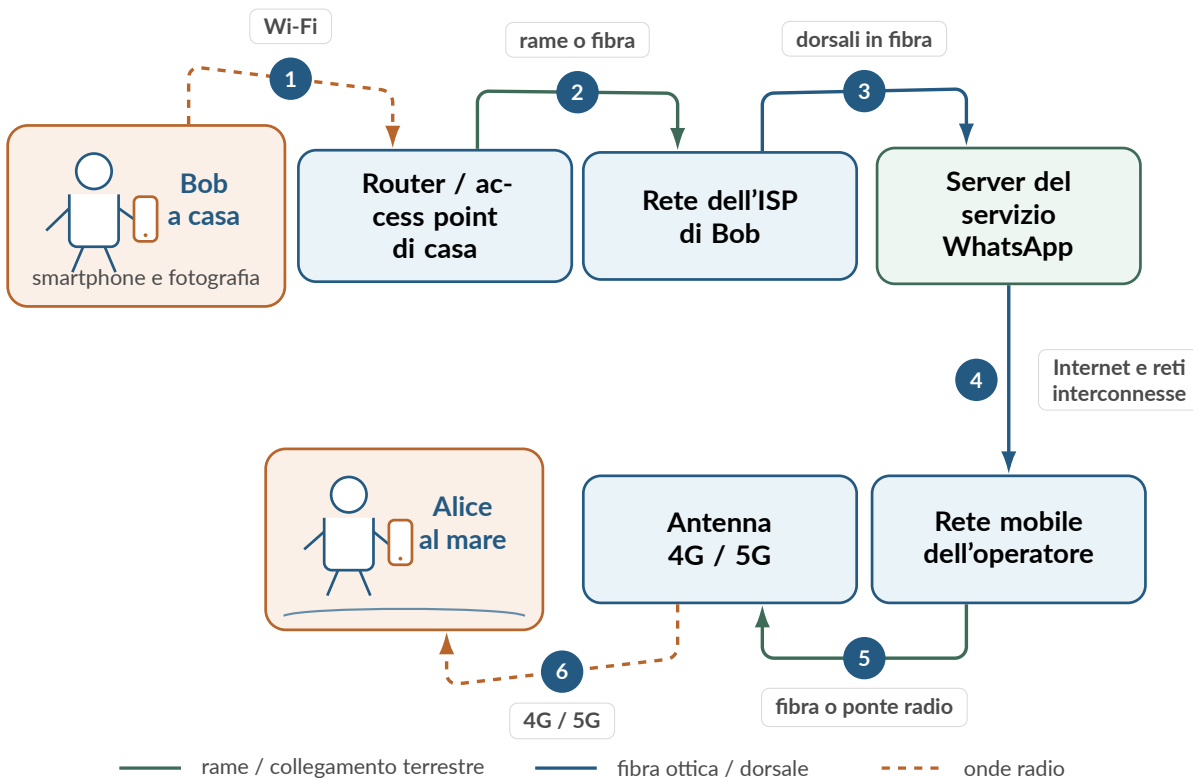


Figura 3.1: Rappresentazione semplificata del viaggio della fotografia da Bob ad Alice. Il percorso reale può comprendere molti altri router, collegamenti e server.

Approfondimento: uno schema necessariamente semplificato

Una grande piattaforma di messaggistica non utilizza normalmente un solo computer, ma numerosi server e centri di elaborazione. Anche le reti degli operatori contengono molti apparati intermedi. La figura rappresenta quindi le tappe principali, utili per comprendere il funzionamento generale, e non l'esatto percorso fisico seguito da ogni singolo pacchetto.

3.1.1 Prima tappa: dallo smartphone di Bob al router

Lo smartphone di Bob trasforma la fotografia in dati digitali e li prepara per la trasmissione. Poiché il telefono è collegato alla rete domestica, i dati raggiungono il router attraverso il **Wi-Fi**.

Il Wi-Fi utilizza onde elettromagnetiche. Tra il telefono e il router non è presente un cavo,

ma il segnale deve comunque attraversare lo spazio e può essere attenuato da pareti, distanza, interferenze e ostacoli.



Figura 3.2: Prima tappa: la fotografia passa dallo smartphone di Bob al router domestico attraverso il Wi-Fi.

Il dispositivo chiamato comunemente **router Wi-Fi** svolge spesso più funzioni contemporaneamente:

- crea la rete wireless domestica;
- collega tra loro i dispositivi della casa;
- inoltra verso Internet i dati destinati a reti esterne;
- riceve da Internet i dati diretti ai dispositivi locali.

Errore comune: Internet e Wi-Fi non sono sinonimi

Il Wi-Fi è una tecnologia usata per collegare senza fili un dispositivo a una rete locale. Internet è invece l'insieme mondiale di reti interconnesse. Una rete Wi-Fi può esistere anche senza accesso a Internet e, al contrario, uno smartphone può accedere a Internet tramite la rete mobile senza usare il Wi-Fi.

3.1.2 Seconda tappa: dal router alla rete dell'operatore

Il router domestico deve consegnare i dati all'azienda che fornisce la connessione. Questa azienda viene chiamata **ISP**, dall'inglese *Internet Service Provider*, cioè fornitore di servizi Internet.

Il tratto tra l'abitazione e la rete dell'ISP può utilizzare tecnologie differenti. Per esempio:

- cavo in rame, nelle connessioni che sfruttano almeno in parte la rete telefonica;
- cavo coassiale, in alcune reti via cavo;
- fibra ottica fino all'abitazione;
- collegamento radio, in alcune zone servite da accesso wireless fisso;
- collegamento satellitare, quando non sono disponibili infrastrutture terrestri adeguate.

Il router di Bob non conosce direttamente il percorso completo verso Alice. Consegna i dati al primo apparato della rete dell'ISP; saranno poi i router successivi a inoltrarli da una rete all'altra.

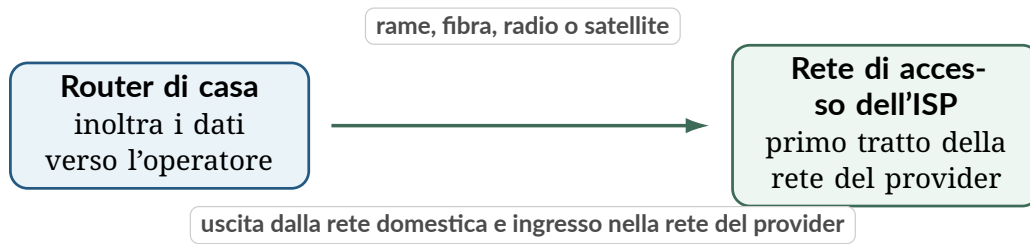


Figura 3.3: Seconda tappa: il router consegna i dati alla rete del fornitore di accesso a Internet.

Definizione: ISP

Un **Internet Service Provider** è un'organizzazione che collega utenti e reti a Internet e gestisce una parte dell'infrastruttura necessaria al trasporto dei dati.

3.1.3 Terza tappa: attraverso la rete di Internet

All'interno della rete dell'ISP, i dati vengono inoltrati attraverso numerosi apparati. I collegamenti principali, chiamati spesso **dorsali**, devono trasportare contemporaneamente il traffico prodotto da moltissimi utenti. Per questo motivo utilizzano in larga parte fibra ottica ad alta capacità.

Il percorso può attraversare:

- reti locali dell'operatore;
- router di raccolta e di dorsale;
- punti di interconnessione tra operatori diversi;
- collegamenti terrestri in fibra;
- cavi sottomarini, quando i dati devono raggiungere altri Paesi o continenti;
- centri di elaborazione nei quali funzionano i server del servizio.

Internet viene definita una **rete di reti**: non appartiene a un unico soggetto e non è costituita da un solo collegamento. È formata da molte reti autonome che accettano di scambiarsi dati secondo regole comuni.

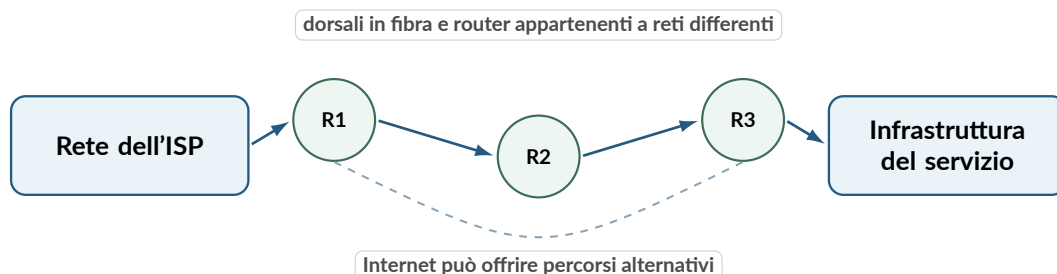


Figura 3.4: Terza tappa: i dati attraversano router e dorsali appartenenti a reti interconnesse.

Curiosità: il percorso non è necessariamente il più breve

I dati non seguono sempre la linea geograficamente più corta. Il percorso dipende dalle connessioni disponibili, dagli accordi tra operatori, dal traffico presente e da eventuali guasti. Inoltre pacchetti appartenenti alla stessa comunicazione possono, in determinate condizioni, attraversare percorsi differenti.

3.1.4 Quarta tappa: il server del servizio

I dati raggiungono l'infrastruttura del servizio WhatsApp. In un modello semplificato, il **server** riceve il messaggio inviato dal client di Bob e lo rende disponibile per il client di Alice.

Il server non è necessariamente un singolo computer. I grandi servizi online distribuiscono normalmente le proprie funzioni tra numerosi sistemi, in modo da:

- servire contemporaneamente molti utenti;
- evitare che un singolo guasto interrompa l'intero servizio;
- conservare temporaneamente i messaggi quando il destinatario non è raggiungibile;
- scegliere infrastrutture vicine agli utenti o adatte a gestire il traffico.

La fotografia è stata quindi trasformata in dati e trasportata fino al servizio di messaggistica. Da questo punto il viaggio prosegue verso la rete utilizzata da Alice.



Figura 3.5: Quarta tappa: l'infrastruttura del servizio riceve il messaggio e lo rende disponibile ad Alice.

3.1.5 Quinta tappa: dall'operatore mobile all'antenna 5G

Alice non è collegata a una rete Wi-Fi. Il suo smartphone comunica via radio con una stazione dell'operatore mobile. Prima di arrivare all'antenna, i dati attraversano la rete dell'operatore, che comprende collegamenti in fibra, router, sistemi di gestione e apparati dedicati alla rete mobile.

L'antenna copre una determinata area, chiamata comunemente **cella**. Quando Alice si sposta, il collegamento può essere trasferito da una cella a un'altra senza richiedere una nuova configurazione manuale da parte dell'utente.

Tra la rete dell'operatore e l'antenna viene spesso utilizzata la fibra ottica; in alcune situazioni possono essere impiegati anche ponti radio. Soltanto l'ultimo tratto, dall'antenna allo smartphone, utilizza necessariamente la comunicazione mobile senza fili.

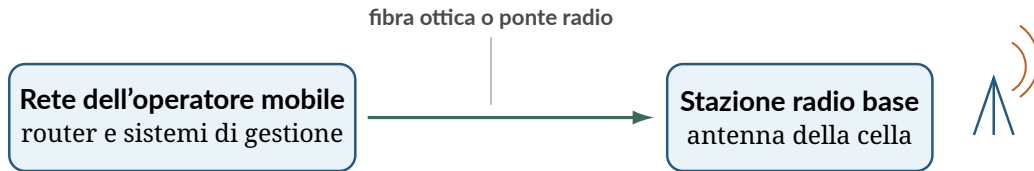


Figura 3.6: Quinta tappa: la rete dell'operatore trasporta i dati fino alla stazione radio base che serve la zona di Alice.

3.1.6 Sesta tappa: la fotografia arriva ad Alice

L'antenna trasmette i dati attraverso la rete 4G o 5G. Lo smartphone di Alice li riceve, li ricompone e l'applicazione visualizza la fotografia.



Figura 3.7: Sesta tappa: l'antenna trasmette i dati via 4G o 5G e lo smartphone di Alice ricostruisce la fotografia.

L'intero percorso ha richiesto la collaborazione di:

- dispositivi finali, come gli smartphone;
- apparati di rete, come router e antenne;
- mezzi trasmissivi differenti;
- reti appartenenti a operatori diversi;
- server che forniscono il servizio;
- protocolli che stabiliscono come rappresentare, indirizzare e trasportare i dati.

Definizione: rete informatica

Una **rete informatica** è un insieme di dispositivi collegati tra loro per scambiare dati e condividere risorse o servizi.

3.1.7 Un viaggio composto da molti piccoli trasferimenti

È utile immaginare il viaggio non come un unico salto da Bob ad Alice, ma come una successione di trasferimenti tra dispositivi vicini. Ogni apparato riceve i dati da un collegamento e li inoltra verso il collegamento successivo.

Tappa	Da / verso	Tecnologia principale mostrata nello schema
1	Smartphone di Bob → router di casa	Wi-Fi, onde radio
2	Router → rete dell'ISP	rame, fibra o collegamento radio
3	ISP → infrastruttura del servizio	dorsali in fibra e router
4	Servizio → operatore di Alice	Internet e reti interconnesse
5	Operatore → antenna	fibra o ponte radio
6	Antenna → smartphone di Alice	rete mobile 4G/5G

Approfondimento: dal contenuto ai pacchetti

Durante la trasmissione la fotografia non viene generalmente trattata come un unico blocco indivisibile. I dati vengono organizzati in unità più piccole che possono essere trasportate e controllate più facilmente. Nei prossimi sottocapitoli verrà introdotto il concetto di pacchetto e si vedrà come gli indirizzi e i protocolli consentono di consegnarlo alla destinazione corretta.

3.1.8 Quando qualcosa non funziona

Il percorso permette anche di comprendere perché un problema di rete può avere cause molto diverse. Se Alice non riceve la fotografia, il guasto potrebbe riguardare:

- lo smartphone o l'applicazione di Bob;
- il collegamento Wi-Fi domestico;
- il router o la linea dell'ISP;
- un collegamento della rete Internet;
- i server del servizio;
- la rete dell'operatore mobile;
- l'antenna o la copertura nella zona di Alice;
- lo smartphone o l'applicazione di Alice.

Per diagnosticare il problema è quindi utile procedere per tappe, verificando prima gli elementi più vicini all'utente e successivamente quelli esterni.

Esempio: una prima diagnosi

Bob riesce ad aprire altri siti dal proprio smartphone, ma l'applicazione di messaggistica non invia la fotografia. Questo rende meno probabile un guasto generale del Wi-Fi o della connessione Internet e suggerisce di controllare l'applicazione o il servizio.

Alice, invece, non riesce a utilizzare nessuna applicazione online e sul telefono non compare il simbolo della rete mobile. In questo caso il problema può riguardare la copertura, la modalità aereo, la SIM o la configurazione del dispositivo.

Esercizi: sezione 3.1

1. Riordinare le seguenti tappe: antenna 5G, router di Bob, smartphone di Alice, ISP di Bob, server del servizio, smartphone di Bob.
2. Spiegare perché la fotografia non passa direttamente dallo smartphone di Bob a quello di Alice.
3. Individuare nello schema della [figura 3.1](#) i tratti wireless e quelli cablati.
4. Indicare in quali tratti può essere utilizzata la fibra ottica.
5. Definire i termini *rete informatica*, *ISP*, *dorsale* e *server*.
6. Spiegare perché Internet viene definita una “rete di reti”.
7. Elencare almeno quattro possibili cause per cui Alice potrebbe non ricevere la fotografia.
8. Bob disattiva il Wi-Fi ma dispone di una connessione mobile. Descrivere come cambia la prima parte del percorso.
9. Alice è collegata al Wi-Fi dell'albergo invece che alla rete 5G. Descrivere come cambia l'ultima parte del percorso.
10. Individuare l'errore nella frase: “La fotografia viaggia tutta insieme lungo un unico cavo che collega i due smartphone”.
11. Disegnare uno schema semplificato del percorso seguito da un'email inviata dal computer della scuola allo smartphone di un familiare.
12. Scegliere un servizio online utilizzato abitualmente e indicare quali dispositivi, reti e collegamenti potrebbero essere coinvolti.

3.2 I mezzi trasmissivi

Per collegare due dispositivi è necessario un mezzo attraverso il quale il segnale possa propagarsi. Questo mezzo può essere un cavo oppure lo spazio circostante. La forma fisica del segnale cambia in base alla tecnologia utilizzata:

- nei cavi di rame i dati vengono rappresentati mediante variazioni di un segnale elettrico;
- nella fibra ottica vengono utilizzati impulsi luminosi;
- nelle comunicazioni senza fili vengono utilizzate onde elettromagnetiche.

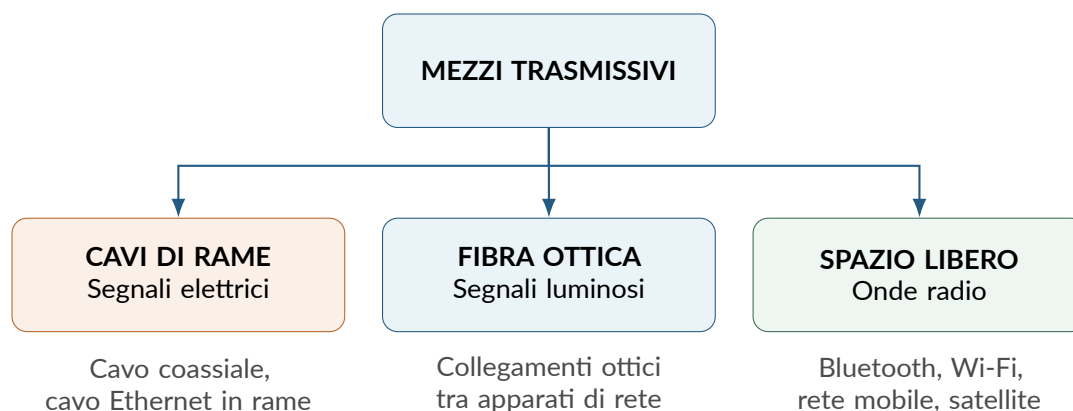


Figura 3.8: Classificazione semplificata dei mezzi trasmissivi in base alla forma del segnale e al supporto utilizzato.

Definizione: mezzo trasmissivo

Un **mezzo trasmissivo** è il supporto fisico o lo spazio attraverso cui si propaga il segnale che trasporta i dati tra due dispositivi.

Non esiste un mezzo migliore in ogni situazione. La scelta dipende da distanza, velocità richiesta, ambiente, costi, mobilità, affidabilità e facilità di installazione.

3.2.1 Trasmettere attraverso il rame

Il rame è un buon conduttore elettrico. Nei cavi di rete i dati vengono trasportati mediante variazioni controllate del segnale elettrico. I due esempi principali considerati in questo capitolo sono il cavo coassiale e il cavo Ethernet a coppie intrecciate.

Il cavo coassiale

Il **cavo coassiale** è formato da più strati disposti intorno allo stesso asse:

1. un conduttore centrale in rame;
2. uno strato isolante;
3. una schermatura metallica;
4. un rivestimento esterno protettivo.

La schermatura protegge il segnale dalle interferenze elettromagnetiche e riduce la dispersione verso l'esterno. Il cavo risulta quindi robusto e adatto a trasportare segnali anche in ambienti nei quali sono presenti disturbi elettrici.

Velocità tipica indicativa: nelle connessioni Internet via cavo può offrire, in base alla tecnologia dell'operatore, velocità da circa 100 Mbps fino a 1 Gbps. Il valore reale dipende dall'impianto e dal servizio condiviso tra gli utenti.

Esempi di utilizzo

- collegamento delle antenne televisive;
- reti televisive e Internet via cavo;
- impianti di videosorveglianza;
- vecchie reti Ethernet a bus.

Vantaggi

- buona protezione dalle interferenze;
- struttura robusta;
- possibilità di trasportare segnali su distanze maggiori rispetto ad alcuni semplici cavi non schermati.

Svantaggi

- è più rigido e ingombrante di un normale cavo Ethernet;
- i connettori e le derivazioni possono essere meno comodi da installare;
- nelle moderne reti locali è stato quasi completamente sostituito dai cavi a coppie intrecciate e dalla fibra.

Il cavo Ethernet a coppie intrecciate

Nelle reti locali moderne il collegamento cablato più comune utilizza un cavo formato da **quattro coppie di fili di rame**. I due fili di ogni coppia sono intrecciati tra loro. L'intrecciatura aiuta a ridurre gli effetti delle interferenze e permette di trasportare i segnali in modo più affidabile.

Il connettore viene comunemente chiamato **RJ45**. Il cavo può collegare, per esempio, un computer a uno switch oppure un access point al resto della rete.

Esistono cavi:

- non schermati, più flessibili ed economici;
- schermati, maggiormente protetti dalle interferenze;
- appartenenti a categorie differenti, progettate per prestazioni diverse.

In questa fase è sufficiente sapere che la categoria del cavo e la qualità dell'installazione influenzano la velocità e l'affidabilità del collegamento.

Velocità tipica indicativa: 1 Gbps è oggi un valore molto comune nelle reti locali; collegamenti meno recenti possono funzionare a 100 Mbps, mentre impianti e apparati adeguati possono raggiungere 2,5, 5 o 10 Gbps. Una singola tratta in rame è normalmente limitata a circa 100 metri.

Esempi di utilizzo

- computer di un laboratorio scolastico;
- stampanti di rete;
- switch e router;
- access point Wi-Fi;
- telecamere di rete;
- smart TV e console.

Vantaggi

- costo contenuto;
- installazione relativamente semplice;
- collegamento stabile e prevedibile;
- prestazioni elevate nelle distanze tipiche di una rete locale;
- possibilità, in alcuni impianti, di trasportare anche alimentazione elettrica insieme ai dati.

Svantaggi

- lunghezza limitata per ogni tratta;
- sensibilità alle interferenze, soprattutto nei cavi e negli impianti meno protetti;
- necessità di posare fisicamente il cavo;
- minore capacità e distanza rispetto alla fibra ottica nelle dorsali.



Cavo coassiale



Cavo Ethernet

Figura 3.9: Esempi di mezzi trasmissivi in rame. Nel cavo coassiale gli strati sono concentrici; nel cavo Ethernet i conduttori sono organizzati in coppie intrecciate.

Errore comune: Ethernet non indica soltanto il cavo

Nel linguaggio comune si parla spesso di “cavo Ethernet”. Ethernet è però un insieme di tecnologie e regole per le reti locali. Il cavo a coppie intrecciate è uno dei mezzi fisici più utilizzati per realizzare collegamenti Ethernet.

3.2.2 Trasmettere attraverso la luce: la fibra ottica

La **fibra ottica** trasporta dati mediante impulsi luminosi. Una fibra è estremamente sottile ed è costituita principalmente da:

- un **nucleo**, nel quale si propaga la luce;
- un **rivestimento ottico**, che contribuisce a mantenere la luce all'interno del nucleo;
- strati protettivi, che rendono il cavo più resistente durante l'installazione e l'utilizzo.

La sorgente luminosa accende e spegne o modifica rapidamente il segnale. All'estremità opposta un ricevitore converte gli impulsi luminosi in segnali elettrici interpretabili dagli apparati di rete.

Velocità tipica indicativa: nelle connessioni FTTH per utenti domestici sono comuni valori da alcune centinaia di Mbps fino a 1 Gbps; nelle dorsali professionali una singola fibra può trasportare decine o centinaia di Gbps. La distanza può variare da centinaia di metri a molti chilometri, in base al tipo di fibra e agli apparati.

Approfondimento: perché la luce rimane nella fibra

Il nucleo e il rivestimento sono realizzati con proprietà ottiche differenti. Quando la luce entra con un'angolazione adatta, viene guidata lungo il nucleo attraverso successive riflessioni. Il principio permette di trasportare il segnale anche quando il cavo segue curve moderate.

Esempi di utilizzo

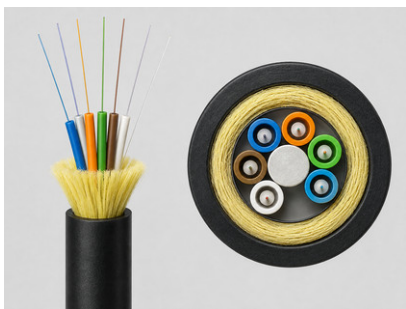
- collegamenti Internet FTTH fino all'abitazione;
- dorsali degli operatori;
- collegamenti tra edifici o piani di una scuola;
- reti dei centri di elaborazione dati;
- cavi sottomarini tra continenti;
- collegamenti tra apparati che richiedono alta capacità o grande distanza.

Vantaggi

- elevata capacità di trasmissione;
- possibilità di coprire grandi distanze;
- ridotta attenuazione rispetto al rame;
- immunità alle interferenze elettromagnetiche;
- dimensioni ridotte delle singole fibre;
- assenza di corrente elettrica nel mezzo ottico.

Svantaggi

- maggiore delicatezza durante posa e terminazione;
- necessità di strumenti e competenze specifiche;
- connettori e apparati ottici generalmente più complessi;
- riparazione meno immediata rispetto a un semplice cavo di rame;
- impossibilità di trasportare alimentazione elettrica attraverso la fibra stessa.



Cavo in fibra ottica



Connettore ottico

Figura 3.10: A sinistra, un cavo in fibra ottica; a destra, un connettore ottico.

FTTC e FTTH

Nelle offerte di connettività si incontrano spesso le sigle FTTC e FTTH:

- **FTTC**, *Fiber To The Cabinet*: la fibra raggiunge un armadio stradale; l'ultimo tratto verso l'abitazione utilizza normalmente il rame;
- **FTTH**, *Fiber To The Home*: la fibra raggiunge direttamente la singola abitazione o unità immobiliare dell'utente. Non va confusa con la **FTTB**, *Fiber To The Building*: in questo caso la fibra termina in un punto comune dell'edificio e il tratto finale verso le singole unità può utilizzare il rame.

In generale, portare la fibra più vicino al dispositivo finale riduce il tratto in rame e permette prestazioni più elevate e stabili. Le prestazioni effettive dipendono comunque dall'impianto, dagli apparati, dal contratto e dal traffico presente.

Errore comune: la fibra non rende istantanea qualsiasi comunicazione

Una connessione in fibra può aumentare notevolmente la capacità del collegamento, ma il tempo necessario per utilizzare un servizio dipende anche dai server, dal percorso attraverso Internet, dalla rete locale e dal dispositivo dell'utente.

3.2.3 Trasmettere senza fili

Nelle comunicazioni wireless non è presente un cavo tra trasmettitore e ricevitore. Le antenne convertono i segnali elettrici in onde elettromagnetiche e compiono l'operazione inversa quando ricevono il segnale.

L'espressione "trasmissione nell'etere" viene talvolta usata nel linguaggio comune. Più precisamente, le onde si propagano nello spazio e possono essere influenzate da distanza, ostacoli, altre trasmissioni e condizioni ambientali.

Le comunicazioni senza fili permettono mobilità e installazioni più flessibili, ma il mezzo è condiviso: più dispositivi possono utilizzare le stesse frequenze e devono coordinarsi per evitare interferenze.

Bluetooth

Il **Bluetooth** è progettato principalmente per collegamenti personali a breve distanza e con consumi contenuti. Viene utilizzato per creare piccole reti intorno a una persona o a un dispositivo.

Velocità e portata indicative: circa 1–3 Mbps nelle applicazioni comuni e una portata tipica di circa 10 metri. Dispositivi e versioni particolari possono raggiungere distanze maggiori, ma pareti e interferenze riducono il risultato.

Esempi di utilizzo

- cuffie e altoparlanti;
- mouse e tastiere;
- smartwatch e dispositivi per l'attività fisica;
- sensori e dispositivi per la domotica;
- trasferimenti occasionali tra dispositivi vicini.

Vantaggi

- consumi generalmente contenuti;
- collegamento semplice tra dispositivi vicini;
- assenza di cavi;
- ampia diffusione su smartphone e computer.

Svantaggi

- portata limitata;
- capacità inferiore rispetto a Ethernet, fibra e alle migliori connessioni Wi-Fi;
- possibili interferenze con altre tecnologie radio;
- necessità di associare e proteggere correttamente i dispositivi.

Wi-Fi

Il **Wi-Fi** viene utilizzato per collegare dispositivi a una rete locale senza posare un cavo per ogni terminale. I dispositivi comunicano generalmente con un **access point**, che li collega al resto della rete.

Un router domestico integra spesso la funzione Wi-Fi insieme all'instradamento tra la rete locale e Internet.

Velocità e portata indicative: nella pratica una buona rete Wi-Fi può offrire da circa 100 a 800 Mbps. In un edificio la copertura utile è spesso nell'ordine di 20–30 metri per punto di accesso; all'aperto e senza ostacoli può essere maggiore.

Esempi di utilizzo

- smartphone, tablet e notebook in casa;
- aule e laboratori scolastici;
- uffici, biblioteche e alberghi;
- dispositivi domestici connessi;
- reti temporanee per eventi.

Vantaggi

- mobilità nell'area coperta;
- facilità di collegamento di nuovi dispositivi;
- riduzione del cablaggio fino ai terminali;
- possibilità di servire dispositivi privi di porta Ethernet.

Svantaggi

- prestazioni variabili in base a distanza e ostacoli;
- banda condivisa tra gli utenti;
- interferenze con reti vicine e altri dispositivi;
- necessità di configurare correttamente sicurezza e password;
- copertura limitata rispetto alle reti mobili.



Dispositivi Bluetooth



Access point Wi-Fi

Figura 3.11: Bluetooth e Wi-Fi utilizzano entrambi onde elettromagnetiche, ma sono progettati per esigenze, distanze e modalità di collegamento differenti.

Esempio: cavo o Wi-Fi in un laboratorio

Un computer fisso vicino allo switch può essere collegato con un cavo Ethernet per ottenere stabilità e prestazioni prevedibili. Un tablet utilizzato dagli studenti deve invece potersi spostare e viene collegato tramite Wi-Fi. Nella stessa rete possono quindi convivere mezzi trasmissivi differenti.

Reti mobili: 3G, 4G e 5G

Le reti mobili dividono il territorio in aree chiamate **celle**. Ogni cella è servita da una stazione radio base. Lo smartphone comunica con l'antenna più adatta e la rete gestisce il passaggio da una cella all'altra durante gli spostamenti.

Le sigle 3G, 4G e 5G indicano generazioni successive di tecnologie mobili:

- il **3G** ha reso comune l'accesso a Internet da smartphone;
- il **4G** ha aumentato notevolmente la capacità per navigazione, video e servizi multimediali;
- il **5G** è progettato per offrire maggiore capacità, tempi di risposta ridotti e supporto a un numero elevato di dispositivi, quando rete e copertura lo consentono.

Il numero della generazione non garantisce da solo una determinata velocità. Le prestazioni dipendono da qualità del segnale, distanza dall'antenna, ostacoli, frequenze utilizzate, numero di utenti e collegamento dell'antenna alla rete dell'operatore.

Velocità e portata indicative:

- 3G: circa 1–10 Mbps; una cella può coprire alcuni chilometri;
- 4G: circa 20–150 Mbps; la portata varia da centinaia di metri a diversi chilometri;
- 5G: circa 100 Mbps–1 Gbps nelle condizioni favorevoli; la portata può andare da poche centinaia di metri, con frequenze elevate, a diversi chilometri con frequenze più basse.

Esempi di utilizzo

- navigazione e messaggistica su smartphone;
- router mobili e connessioni di emergenza;
- tablet e computer dotati di SIM;
- sensori, veicoli e dispositivi distribuiti sul territorio;
- accesso a Internet in movimento.

Vantaggi

- ampia copertura geografica;
- mobilità tra celle differenti;
- collegamento disponibile senza rete locale dell'utente;
- rapidità di attivazione in presenza di copertura.

Svantaggi

- qualità variabile in base alla zona e all'affollamento;
- possibili limiti di traffico previsti dal contratto;
- maggiore consumo energetico del dispositivo in condizioni di segnale debole;
- dipendenza dall'infrastruttura e dalla copertura dell'operatore.



Figura 3.12: Una stazione radio base collega gli smartphone presenti nella cella alla rete dell'operatore.

Comunicazioni satellitari

Un collegamento satellitare utilizza onde radio per comunicare tra una stazione a terra e uno o più satelliti. Il satellite inoltra il segnale verso un'altra stazione o verso l'infrastruttura del servizio.

Per una trattazione introduttiva è utile distinguere due modelli.

Satelliti tradizionali ad alta quota I servizi satellitari tradizionali possono utilizzare satelliti posti a grande distanza dalla superficie terrestre, capaci di coprire aree molto estese. Una singola posizione può servire regioni molto ampie, ma il segnale deve compiere un percorso lungo.

Vantaggi

- copertura di vaste aree;
- utilità in località isolate, montagne, mare e zone rurali;
- indipendenza dalla posa di un cavo fino a ogni utente.

Svantaggi

- ritardo maggiore dovuto alla distanza percorsa dal segnale;
- necessità di un'antenna orientata e con adeguata visibilità del cielo;
- possibili effetti delle condizioni meteorologiche;

- costi e capacità condivisa tra molti utenti.

Costellazioni in orbita bassa Sistemi come **Starlink** utilizzano numerosi satelliti in orbita più vicina alla Terra. Il terminale dell'utente cambia il satellite con cui comunica mentre la costellazione si muove.

La minore distanza può ridurre il ritardo rispetto ai tradizionali collegamenti satellitari ad alta quota. Per garantire continuità sono però necessari molti satelliti, stazioni a terra, sistemi di controllo e terminali specifici.

Velocità e portata indicative: i servizi satellitari tradizionali possono offrire circa 25–150 Mbps, mentre i sistemi in orbita bassa possono raggiungere indicativamente 50–250 Mbps. Il collegamento radio copre distanze da centinaia a decine di migliaia di chilometri e permette di servire aree molto vaste; ostacoli, meteo e congestione possono però ridurre le prestazioni.

Vantaggi

- disponibilità anche dove mancano connessioni terrestri adeguate;
- ritardo generalmente inferiore rispetto ai sistemi satellitari posti a quota molto maggiore;
- installazione possibile senza scavare o posare lunghi cavi locali.

Svantaggi

- costo del terminale e del servizio;
- necessità di una visuale ampia del cielo;
- possibili interruzioni dovute a ostacoli o condizioni ambientali;
- prestazioni variabili in base alla copertura e al numero di utenti;
- dipendenza da una complessa infrastruttura spaziale e terrestre.



Antenna satellitare tradizionale



Terminale per costellazione LEO

Figura 3.13: Confronto visivo tra una parabola satellitare tradizionale e un terminale utilizzato da servizi basati su satelliti in orbita bassa.

Approfondimento: anche il satellite si collega a una rete terrestre

Il satellite non sostituisce completamente Internet terrestre. I dati devono normalmente raggiungere stazioni, gateway e dorsali collegate agli altri operatori e ai server. Il tratto via satellite è quindi una parte di un percorso più ampio.

Approfondimento: mezzi guidati e non guidati

Un'altra classificazione distingue i mezzi trasmissivi in due gruppi:

- **guidati:** il segnale segue un supporto fisico, come un cavo in rame o una fibra ottica;
- **non guidati:** il segnale si propaga nello spazio attraverso onde elettromagnetiche, come nel Bluetooth, nel Wi-Fi, nelle reti mobili e nei collegamenti satellitari.

I mezzi guidati rendono il percorso fisico più definito e sono spesso preferiti quando servono stabilità e capacità elevata. I mezzi non guidati permettono invece mobilità e collegamenti nei luoghi in cui la posa dei cavi è difficile o poco conveniente. Questa classificazione descrive il modo in cui il segnale si propaga e non indica automaticamente quale tecnologia sia più veloce o migliore.

3.2.4 Confrontare i mezzi trasmissivi

La tabella seguente riassume le caratteristiche generali. Le prestazioni reali dipendono dalla versione della tecnologia, dalla qualità degli apparati e dall'ambiente.

Mezzo	Velocità tipica indicativa	Distanza o portata indicativa
Coassiale	100 Mbps–1 Gbps	decine o centinaia di metri, secondo l'impianto
Ethernet su rame	1 Gbps molto comune	circa 100 m per tratta
Fibra ottica	1 Gbps nelle FTTH; molto oltre nelle dorsali	centinaia di metri o molti chilometri
Bluetooth	1–3 Mbps	normalmente circa 10 m
Wi-Fi	100–800 Mbps nella pratica	circa 20–30 m in edificio
3G/4G/5G	da pochi Mbps fino a circa 1 Gbps	centinaia di metri o diversi chilometri
Satellite	circa 25–250 Mbps	collegamenti radio di centinaia o migliaia di km

Mezzo	Principali vantaggi	Principali limiti
Coassiale	schermatura e robustezza	rigidità e uso ridotto nelle LAN moderne
Ethernet su rame	economico, stabile, semplice	posa del cavo e interferenze
Fibra ottica	velocità, distanza, immunità ai disturbi	posa e terminazione più complesse
Bluetooth	bassi consumi e semplicità	breve distanza e capacità limitata
Wi-Fi	mobilità e assenza di cavi	interferenze, ostacoli e banda condivisa
3G/4G/5G	copertura e mobilità	prestazioni dipendenti dalla cella
Satellite	copertura senza cavo locale	costo, ostacoli, ritardo e variabilità

Tabella 3.1: Confronto generale tra i principali mezzi trasmissivi, suddiviso in prestazioni e aspetti pratici. I valori sono ordini di grandezza indicativi e non velocità garantite.

3.2.5 Come scegliere il mezzo adatto

La scelta deve essere collegata al problema concreto. Occorre considerare:

- **distanza:** un collegamento all'interno di un'aula ha esigenze diverse da una dorsale tra città;
- **mobilità:** smartphone e tablet richiedono normalmente un collegamento wireless;
- **capacità:** video, backup e molti utenti contemporanei richiedono maggiore banda;
- **affidabilità:** un servizio essenziale può richiedere collegamenti stabili o ridondanti;
- **ambiente:** pareti, disturbi elettrici, condizioni atmosferiche e ostacoli influenzano la scelta;
- **costo:** devono essere considerati materiali, posa, apparati e manutenzione;
- **sicurezza:** il controllo fisico del percorso e la corretta configurazione possono essere importanti.

Esempio: collegare tre edifici scolastici

Per collegare stabilmente tre edifici vicini potrebbe essere utilizzata la fibra ottica, adatta a distanze maggiori e non sensibile ai disturbi elettrici. All'interno di ogni edificio i computer fissi potrebbero usare Ethernet, mentre notebook e tablet potrebbero accedere tramite Wi-Fi. Un collegamento mobile potrebbe essere mantenuto come soluzione temporanea di emergenza.

Esempio: sensore in un campo agricolo

Un sensore posto vicino a una centralina potrebbe utilizzare Bluetooth o un'altra tecnologia radio a corto raggio. Se il dispositivo è isolato e deve inviare dati a distanza, potrebbe servirsi della rete mobile. In un'area priva di copertura terrestre potrebbe essere valutato un collegamento satellitare, considerando costi, energia disponibile e visibilità del cielo.

Errore comune: wireless non significa automaticamente più lento

Le prestazioni non dipendono soltanto dalla presenza o assenza del cavo. Una tecnologia wireless moderna in buone condizioni può essere più veloce di un vecchio collegamento cablato. Per confrontare due soluzioni occorre considerare versione, apparati, distanza, interferenze, numero di utenti e qualità dell'installazione.

Esercizi: sezione 3.2

1. Definire il termine *mezzo trasmissivo*.
2. Associare a ciascun mezzo la forma del segnale: cavo Ethernet, fibra ottica, Wi-Fi, cavo coassiale, rete 5G.
3. Descrivere i principali strati di un cavo coassiale e spiegare la funzione della schermatura.
4. Perché i fili di un cavo Ethernet sono organizzati in coppie intrecciate?
5. Indicare tre vantaggi e tre svantaggi della fibra ottica.
6. Spiegare la differenza generale tra FTTC e FTTH.
7. Confrontare Bluetooth e Wi-Fi indicando almeno due utilizzi adatti a ciascuna tecnologia.
8. Spiegare perché una connessione 5G può avere prestazioni differenti in due luoghi diversi.
9. Confrontare un satellite tradizionale ad alta quota con una costellazione in orbita bassa.
10. Individuare l'errore nella frase: "Una connessione satellitare non utilizza mai cavi o reti terrestri".
11. Classificare rame, fibra, Bluetooth e 4G come mezzi guidati o non guidati.
12. Scegliere il mezzo più adatto e motivare la risposta:
 - a) collegare un computer fisso allo switch nella stessa aula;
 - b) collegare due edifici scolastici distanti 300 metri;
 - c) collegare cuffie a uno smartphone;
 - d) fornire Internet temporaneo durante un'uscita didattica;
 - e) collegare una struttura isolata priva di rete terrestre.

Esercizi: sezione 3.2 (continua)

13. In un laboratorio il Wi-Fi è lento quando tutti gli studenti guardano un video. Formulare almeno tre possibili spiegazioni.
14. Preparare una tabella personale con un esempio, un vantaggio e uno svantaggio per ciascun mezzo studiato.
15. Tornare alla [figura 3.1](#) e indicare quali mezzi trasmissivi potrebbero essere utilizzati in ogni tratto.

3.3 I dispositivi di indirizzamento

I mezzi trasmissivi permettono al segnale di propagarsi, ma una rete ha bisogno anche di apparati che ricevano i dati e stabiliscano dove inoltrarli. In questo capitolo tali apparati verranno indicati, in senso didattico, come **dispositivi di indirizzamento**.

L'espressione comprende apparati con capacità differenti: l'hub non legge realmente un indirizzo e ripete il segnale verso tutti, mentre switch e router utilizzano informazioni di destinazione per effettuare una scelta.

Definizione: dispositivo di indirizzamento

Un **dispositivo di indirizzamento** è un apparato che collega dispositivi o reti e contribuisce a ripetere oppure inoltrare i dati verso la destinazione.

3.3.1 Dalla trasmissione all'inoltro

Quando un dato raggiunge un apparato di rete, il comportamento dipende dal dispositivo utilizzato:

- un **hub** ripete il segnale verso tutte le altre porte;
- uno **switch** inoltra il dato verso la porta del destinatario nella rete locale;
- un **router** collega reti differenti e inoltra il pacchetto verso la rete corretta.

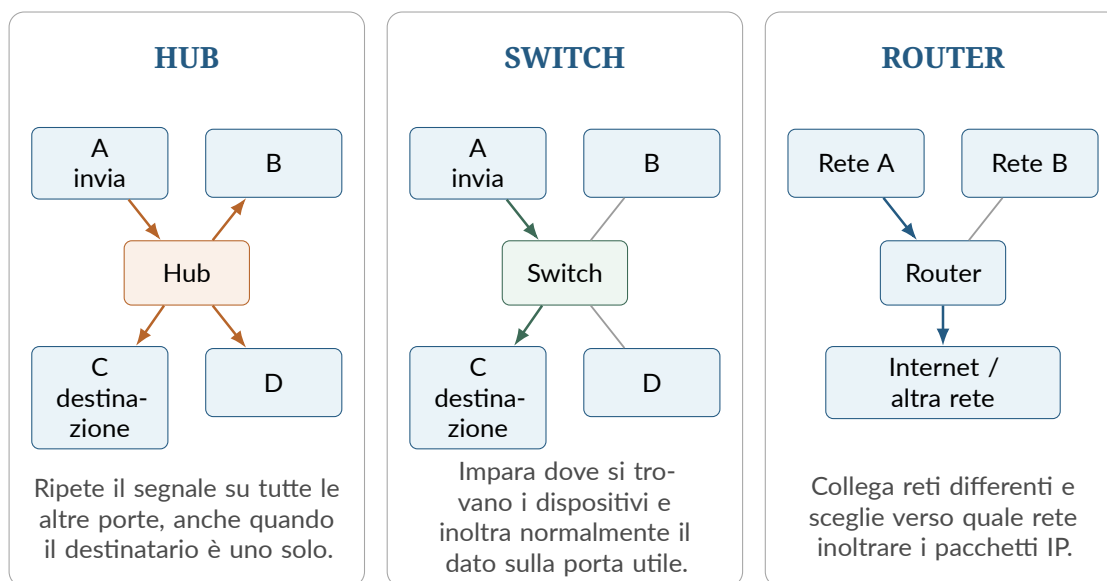


Figura 3.14: Confronto semplificato tra hub, switch e router.

Errore comune: dispositivo e mezzo trasmissivo non sono la stessa cosa

Il cavo Ethernet, la fibra ottica e le onde radio sono mezzi attraverso cui il segnale si propaga. Hub, switch e router sono invece apparati che ricevono e inoltrano i dati. Un router domestico può utilizzare contemporaneamente più mezzi: fibra o rame verso l'operatore, Ethernet verso un computer e Wi-Fi verso uno smartphone.

3.3.2 L'hub: ripetere verso tutti

L'**hub** è un apparato semplice utilizzato nelle vecchie reti locali cablate. Quando riceve un segnale su una porta, lo ripete verso tutte le altre porte, senza controllare quale dispositivo sia il destinatario.

Se il computer A invia un dato al computer C, anche B e D ricevono il segnale e devono ignorarlo. Tutti i dispositivi condividono inoltre la capacità disponibile.

Vantaggi

- funzionamento molto semplice;
- nessuna configurazione per l'inoltro ordinario.

Svantaggi

- produce traffico inutile;
- condivide la capacità tra tutti i dispositivi;
- offre prestazioni e riservatezza inferiori rispetto a uno switch.

Gli hub sono oggi quasi completamente sostituiti dagli switch, ma sono utili per comprendere l'evoluzione delle reti Ethernet.



Hub Ethernet



Switch Ethernet

Figura 3.15: Hub e switch possono avere un aspetto simile, ma inoltrano i dati in modo differente.

3.3.3 Lo switch: scegliere la porta corretta

Lo **switch** collega i dispositivi della stessa rete locale. È comune nei laboratori, negli uffici e negli armadi di rete.

Ogni scheda di rete possiede un **indirizzo MAC**. Lo switch impara progressivamente quale dispositivo è collegato a ogni porta. Quando conosce la posizione del destinatario, inoltra il dato soltanto sulla porta corretta.

Il funzionamento essenziale può essere riassunto così:

1. riceve il dato da una porta;
2. legge l'indirizzo MAC del destinatario;
3. consulta la propria tabella;
4. inoltra il dato verso la porta associata al destinatario.

Vantaggi

- riduce il traffico inutile;
- permette più comunicazioni contemporanee;
- amplia facilmente una rete locale.

Svantaggi

- richiede alimentazione;
- il guasto dello switch centrale può isolare molti dispositivi;
- non collega da solo la LAN a Internet.

Esempio: lo switch del laboratorio

Quando un computer invia un documento alla stampante di rete, lo switch inoltra normalmente i dati soltanto verso la porta alla quale è collegata la stampante, senza trasmetterli a tutti gli altri computer.

Curiosità: alimentazione attraverso Ethernet

Alcuni switch supportano il **PoE**, dall'inglese *Power over Ethernet*. Lo stesso cavo può trasportare dati ed energia verso dispositivi compatibili, come telecamere o punti di accesso Wi-Fi.

3.3.4 Il router: collegare reti differenti

Il **router** collega reti differenti. In una casa o in una scuola permette alla rete locale di comunicare con la rete dell'operatore e quindi con Internet.

Per scegliere dove inoltrare un pacchetto legge l'**indirizzo IP** del destinatario e individua la rete verso cui inviarlo. Lo stesso procedimento viene ripetuto dai router successivi fino alla destinazione.

Un router domestico riunisce spesso nello stesso apparecchio più funzioni:

- collega la LAN alla linea dell'operatore;
- offre alcune porte Ethernet, comportandosi anche come un piccolo switch;
- crea la rete Wi-Fi, svolgendo la funzione di punto di accesso;
- assegna automaticamente le impostazioni di rete ai dispositivi;
- applica semplici controlli di sicurezza;
- può integrare la parte necessaria per una linea in rame oppure collegarsi alla terminazione della fibra.

Non tutti i router comprendono ogni funzione. Nelle reti scolastiche o aziendali router, switch e punti di accesso possono essere apparati separati; in una rete domestica sono spesso integrati per rendere l'installazione più semplice.

Vantaggi

- collega la rete locale a Internet;
- separa il traffico locale da quello diretto verso altre reti;
- permette a molti dispositivi di condividere la connessione;
- può integrare Ethernet, Wi-Fi e funzioni di sicurezza.

Svantaggi

- una configurazione errata può impedire l'accesso a Internet;
- il guasto del router di accesso isola la LAN dai servizi esterni;
- un unico apparecchio integrato concentra più funzioni nello stesso punto.



Router professionale



Router domestico multifunzione

Figura 3.16: Il router professionale può essere separato dagli altri apparati; quello domestico integra spesso porte Ethernet, Wi-Fi e collegamento alla linea dell'operatore.

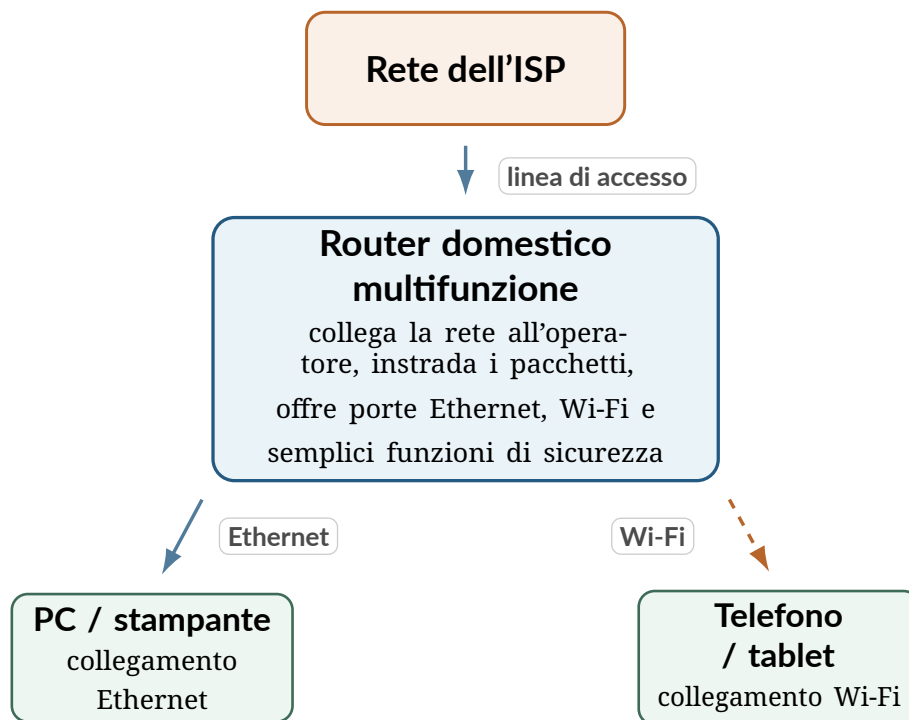


Figura 3.17: Esempio semplificato delle funzioni che possono essere integrate o coordinate da un router domestico.

Errore comune: il router non crea Internet

Il router collega la LAN alla rete dell'operatore, ma non produce da solo la connessione. Se la linea dell'ISP è interrotta, i dispositivi locali possono ancora comunicare tra loro, ma non raggiungono i servizi esterni.

3.3.5 Confrontare i dispositivi di indirizzamento

Dispositivo	Funzione principale	Informazione utilizzata	Esempio tipico
Hub	ripete verso tutte le porte	non seleziona il destinatario	vecchia rete Ethernet condivisa
Switch	inoltra nella rete locale	indirizzo MAC e porta associata	laboratorio informatico
Router	collega reti differenti	indirizzo IP e rete di destinazione	collegamento tra LAN e Internet

Tabella 3.2: Confronto generale tra i principali dispositivi di indirizzamento.

Esempio: una piccola rete scolastica

Dodici computer e una stampante possono essere collegati a uno switch. Il router collega questa LAN all'operatore e può anche offrire il Wi-Fi, oppure coordinare punti di accesso separati nelle diverse aule.

Esercizi: sezione 3.3

1. Spiegare la differenza tra mezzo trasmissivo e dispositivo di indirizzamento.
2. Che cosa fa un hub quando riceve un segnale?
3. Perché lo switch riduce il traffico inutile rispetto all'hub?
4. Quale indirizzo utilizza lo switch per riconoscere il destinatario?
5. Qual è la differenza fondamentale tra switch e router?
6. Quale indirizzo utilizza il router per scegliere la rete di destinazione?
7. Elencare almeno quattro funzioni che possono essere integrate in un router domestico.
8. Perché nelle reti scolastiche router, switch e Wi-Fi possono essere gestiti da apparati separati?
9. Associare ogni situazione al dispositivo più adatto:
 - a) collegare venti computer nella stessa aula;
 - b) collegare la LAN della scuola a Internet;
 - c) ripetere indiscriminatamente un segnale su tutte le porte.
10. In un laboratorio i computer comunicano tra loro, ma non accedono a Internet. Quale dispositivo o collegamento potrebbe avere un problema?
11. Spiegare perché il guasto dello switch centrale produce conseguenze diverse dal guasto del router verso Internet.
12. Disegnare una rete domestica indicando router, dispositivi Ethernet, dispositivi Wi-Fi e collegamento dell'ISP.

3.4 Le topologie di rete

Una rete non è descritta soltanto dai dispositivi e dai mezzi trasmissivi utilizzati. Un **nodo di rete** è un dispositivo o punto capace di inviare, ricevere o inoltrare dati. È importante anche osservare *come* i nodi sono collegati tra loro. La disposizione dei collegamenti prende il nome di **topologia di rete**.

La topologia influenza:

- la quantità di cablaggio necessaria;
- il costo dell'installazione;
- la facilità con cui la rete può essere ampliata;
- le conseguenze di un guasto;
- la presenza di percorsi alternativi;
- la semplicità della manutenzione.

Definizione: topologia di rete

La **topologia di rete** descrive la disposizione dei nodi e dei collegamenti attraverso i quali i dati possono circolare.

3.4.1 Topologia a bus

Nella **topologia a bus** tutti i nodi condividono un unico collegamento principale, chiamato dorsale o bus. Ogni dispositivo si collega al cavo comune e riceve i segnali che lo attraversano.

Nelle implementazioni storiche era necessario terminare correttamente le estremità del cavo per evitare riflessioni del segnale. Un guasto o un'interruzione del bus poteva compromettere l'intera rete.

Vantaggi

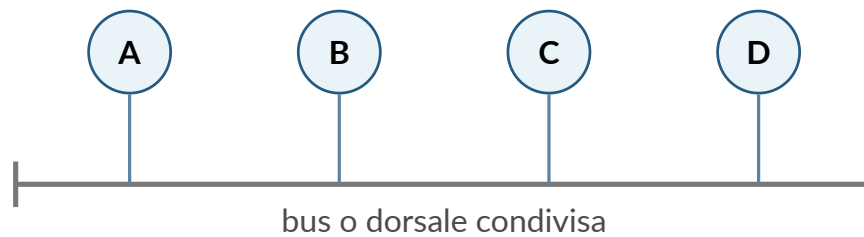
- utilizza una quantità ridotta di cavo;
- non richiede necessariamente un apparato centrale;
- struttura iniziale relativamente semplice.

Svantaggi

- tutti i nodi condividono lo stesso mezzo;
- le prestazioni peggiorano con l'aumento del traffico;
- un guasto sul collegamento principale può bloccare molti dispositivi;
- individuare il punto del guasto può essere difficile;
- aggiungere o rimuovere nodi può disturbare la rete.

Esempi di utilizzo

La topologia a bus è associata soprattutto alle prime reti Ethernet su cavo coassiale e ad alcuni sistemi industriali o di comunicazione interna. Nelle LAN moderne è stata in gran parte sostituita dalla stella con switch.



Tutti i nodi condividono lo stesso mezzo: il traffico e gli eventuali guasti della dorsale possono coinvolgere l'intera rete.

Figura 3.18: Topologia a bus con un unico collegamento condiviso.

3.4.2 Topologia ad anello

Nella **topologia ad anello** ogni nodo è collegato a due nodi vicini e l'insieme dei collegamenti forma un circuito chiuso. I dati possono circolare in una sola direzione oppure in entrambe, a seconda della tecnologia utilizzata.

Vantaggi

- percorso ordinato tra i nodi;
- possibilità di gestire l'accesso al mezzo in modo prevedibile;
- nelle versioni con doppio anello può essere disponibile un percorso alternativo.

Svantaggi

- in un anello semplice un guasto può interrompere la comunicazione;
- l'aggiunta di un nodo può richiedere la modifica del circuito;
- manutenzione e ampliamento possono essere più complessi rispetto alla stella.

Esempi di utilizzo

La topologia ad anello è stata impiegata in reti storiche come Token Ring. Strutture ad anello, spesso ridondanti, vengono ancora utilizzate in alcune reti industriali, metropolitane e dorsali in fibra, perché permettono di raggiungere una destinazione da direzioni differenti.

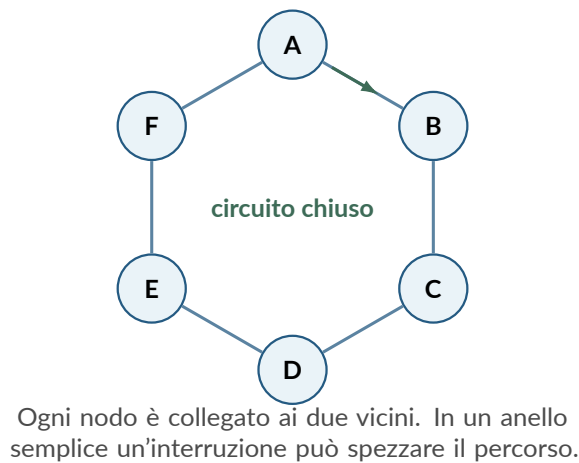


Figura 3.19: Topologia ad anello: ogni nodo è collegato ai due nodi vicini.

3.4.3 Topologia a stella

Nella **topologia a stella** ogni dispositivo è collegato con un proprio collegamento a un apparato centrale. Nelle moderne reti Ethernet l'apparato centrale è normalmente uno switch; nelle reti Wi-Fi può essere un access point.

Vantaggi

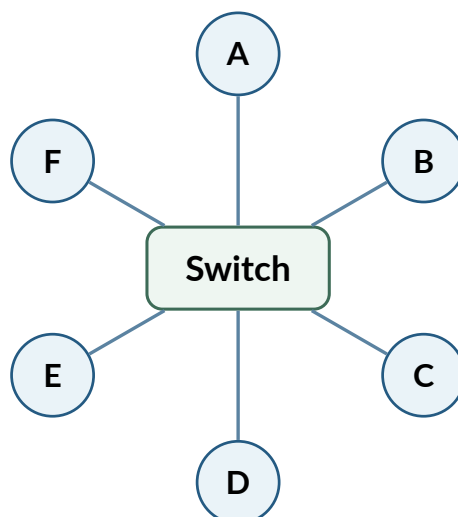
- il guasto del cavo di un dispositivo isola normalmente soltanto quel dispositivo;
- aggiungere o rimuovere un nodo è relativamente semplice;
- i guasti sono più facili da localizzare;
- lo switch permette comunicazioni efficienti e contemporanee;
- la struttura è adatta alla gestione centralizzata.

Svantaggi

- richiede un collegamento separato per ogni nodo;
- utilizza più cavo rispetto a una topologia a bus;
- il guasto dell'apparato centrale può interrompere l'intera rete;
- il numero di dispositivi dipende dalle porte o dalla capacità dell'apparato centrale.

Esempi di utilizzo

- laboratorio informatico con computer collegati a uno switch;
- rete domestica con dispositivi collegati al router;
- rete Wi-Fi con client associati a un access point;
- piccolo ufficio con stampanti, computer e telefoni di rete.



Ogni nodo possiede un collegamento separato verso l'apparato centrale. Un guasto al centro può fermare l'intera rete.

Figura 3.20: Topologia a stella con uno switch centrale.

Esempio: guasto in una rete a stella

Se si interrompe il cavo che collega il computer 7 allo switch, soltanto il computer 7 perde la connessione. Se invece si guasta lo switch centrale, tutti i dispositivi collegati a quello switch smettono di comunicare. La stessa topologia reagisce quindi in modo diverso a seconda del componente danneggiato.

3.4.4 Topologia ad albero

La **topologia ad albero** organizza più reti a stella secondo una struttura gerarchica. Uno switch centrale può essere collegato a switch di piano, di reparto o di edificio; questi ultimi collegano i dispositivi finali.

Vantaggi

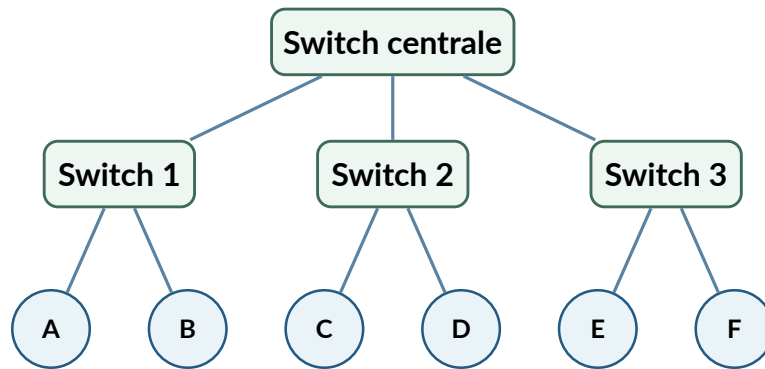
- permette di organizzare reti con molti dispositivi;
- facilita la suddivisione per edifici, piani, laboratori o reparti;
- consente di ampliare la rete aggiungendo nuovi rami;
- rende più ordinata la distribuzione del cablaggio.

Svantaggi

- un guasto in un apparato di livello superiore può isolare un intero ramo;
- i collegamenti centrali devono sopportare il traffico prodotto dai rami inferiori;
- progettazione e gestione sono più complesse rispetto a una singola stella;
- un errore nella gerarchia può creare colli di bottiglia.

Esempi di utilizzo

- rete scolastica con uno switch centrale e switch nei diversi edifici;
- azienda organizzata per piani o reparti;
- rete universitaria composta da più dipartimenti;
- infrastruttura di un centro dati organizzata su più livelli.



Più stelle sono organizzate in livelli gerarchici: un guasto in alto può isolare un intero ramo.

Figura 3.21: Topologia ad albero organizzata su più livelli gerarchici.

3.4.5 Topologia a maglia

Nella **topologia a maglia** un nodo dispone di collegamenti verso più nodi. In questo modo possono esistere percorsi alternativi tra origine e destinazione.

Si distingue tra:

- **maglia completa**, nella quale ogni nodo è collegato direttamente a tutti gli altri;
- **maglia parziale**, nella quale soltanto alcuni nodi possiedono più collegamenti.

Una maglia completa offre molti percorsi, ma il numero di collegamenti cresce rapidamente. Per questo motivo le reti reali utilizzano più spesso maglie parziali.

Vantaggi

- presenza di percorsi alternativi;
- maggiore tolleranza ai guasti;
- possibilità di distribuire il traffico su collegamenti differenti;
- assenza di dipendenza assoluta da un solo collegamento centrale.

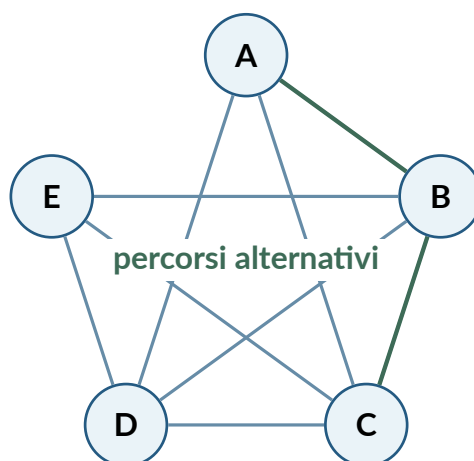
Svantaggi

- costo elevato di collegamenti e apparati;
- progettazione e configurazione più complesse;

- maggiore difficoltà nel controllo e nella manutenzione;
- una maglia completa diventa poco pratica con molti nodi.

Esempi di utilizzo

- dorsali di Internet e reti degli operatori;
- collegamenti ridondanti tra centri dati;
- reti wireless mesh;
- sistemi nei quali la continuità del servizio è essenziale.



La presenza di più collegamenti permette di raggiungere una destinazione anche quando uno dei percorsi non è disponibile.

Figura 3.22: Esempio di topologia a maglia con percorsi alternativi.

Approfondimento: ridondanza

La **ridondanza** consiste nel predisporre risorse o percorsi alternativi. Se un collegamento si interrompe, i dati possono essere inoltrati attraverso un'altra strada. La ridondanza aumenta affidabilità e continuità, ma richiede maggiori costi e una configurazione capace di gestire correttamente i percorsi multipli.

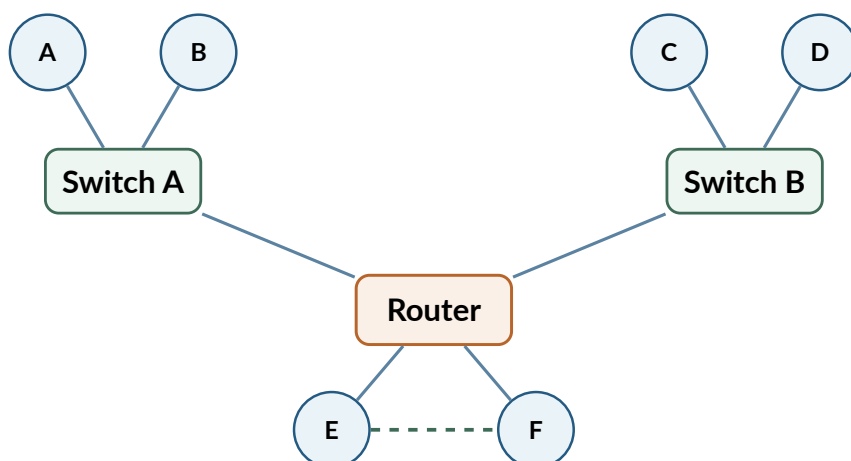
3.4.6 Topologie ibride

Le reti reali raramente corrispondono a una sola forma geometrica. Una **topologia ibrida** combina strutture differenti per adattarsi alle esigenze dell'ambiente.

Una scuola, per esempio, può utilizzare:

- una stella in ogni laboratorio;
- una struttura ad albero per collegare piani ed edifici;
- due collegamenti ridondanti tra gli apparati principali;
- reti Wi-Fi a stella attorno a numerosi access point;

- un collegamento punto-punto in fibra verso una sede distaccata.



La rete combina stelle, collegamenti gerarchici e un percorso ridondante: le topologie reali sono spesso ibride.

Figura 3.23: Esempio di topologia ibrida ottenuta combinando strutture differenti.

Errore comune: una forma non è sempre visibile osservando i cavi

La topologia rappresentata nei diagrammi è un modello. Nella realtà i cavi possono seguire pareti, canaline e percorsi irregolari senza formare visivamente una stella o un albero. La topologia dipende da quali nodi sono collegati tra loro, non dalla forma geometrica assunta fisicamente dal cavo nell'edificio.

3.4.7 Confrontare le topologie

Topologia	Elemento caratteristico	Principali vantaggi	Principali limiti	Esempio
Bus	unico collegamento condiviso	poco cablaggio	guasto e traffico sul bus influenzano tutti	vecchia Ethernet coassiale
Anello	ogni nodo ha due vicini	percorso ordinato, possibile ridondanza	interruzioni e ampliamenti complessi	anello in fibra
Stella	apparato centrale	gestione semplice e guasti locali isolabili	dipendenza dal centro	laboratorio con switch
Albero	stelle organizzate in livelli	scalabile e gerarchico	guasti dei livelli alti isolano rami	rete di un istituto
Maglia	più percorsi tra i nodi	affidabilità e ridondanza	costo e complessità	dorsale di un operatore
Ibrida	combinazione di topologie	adattabile alle esigenze	progettazione più articolata	rete aziendale reale

Tabella 3.3: Confronto generale tra le principali topologie di rete.

3.4.8 Scegliere una topologia

Non esiste una topologia migliore in assoluto. La scelta dipende dal contesto e deve considerare:

- numero e distribuzione dei dispositivi;
- distanza tra gli ambienti;
- importanza della continuità del servizio;
- costo dei collegamenti e degli apparati;
- possibilità di ampliare la rete in futuro;
- facilità di manutenzione;
- quantità di traffico prevista;
- conseguenze accettabili in caso di guasto.

Esempio: rete di una scuola su tre piani

Ogni aula e laboratorio può utilizzare una topologia a stella attorno a uno switch di piano. Gli switch dei piani vengono collegati a uno switch centrale, formando nel complesso una topologia ad albero. Se due collegamenti distinti uniscono gli apparati principali, una parte della rete assume anche caratteristiche di maglia e aumenta la ridondanza.

Esempio: rete temporanea per una serra

Alcuni sensori possono comunicare con un punto di raccolta centrale, formando una stella wireless. Se i sensori sono molto distanti o ostacolati, una rete mesh può permettere ai nodi di inoltrare i dati l'uno per l'altro. La soluzione più affidabile dipende da distanze, energia disponibile, ostacoli e importanza dei dati raccolti.

Esercizi: sezione 3.4

1. Definire il termine *topologia di rete*.
2. Perché una rete Wi-Fi con access point può essere rappresentata come una stella?
3. Descrivere la topologia a bus e indicare due suoi limiti.
4. Descrivere la topologia ad anello e spiegare l'effetto di un'interruzione in un anello semplice.
5. Perché la topologia a stella è molto utilizzata nelle LAN moderne?
6. Qual è la differenza tra guasto di un cavo periferico e guasto dell'apparato centrale in una stella?
7. Spiegare come è organizzata una topologia ad albero.
8. Distinguere maglia completa e maglia parziale.
9. Che cosa significa ridondanza e perché è utile?
10. Spiegare perché le reti reali sono spesso ibride.
11. Associare ogni situazione alla topologia più adatta e motivare:
 - a) venti PC collegati a uno switch di laboratorio;
 - b) rete con più percorsi alternativi tra centri dati;

Esercizi: sezione 3.4 (continua)

- c) rete di un istituto composta da più piani e laboratori;
 - d) vecchia rete che condivide un unico cavo coassiale.
12. In una rete a stella non comunica più un solo computer. Quali elementi è opportuno controllare per primi?
 13. In una rete ad albero si guasta lo switch di un piano. Quali dispositivi possono essere interessati? Che cosa accadrebbe invece con il guasto dello switch centrale?
 14. Disegnare una rete composta da due stelle collegate tra loro e indicare se può essere considerata anche una semplice struttura ad albero.
 15. Progettare la topologia di una piccola azienda con due uffici, un magazzino e una connessione Internet. Indicare dispositivi centrali, collegamenti e possibili punti di guasto.
 16. Tornare alla [figura 3.1](#): individuare almeno una struttura a stella e una parte della rete che può essere considerata a maglia.

3.5 Tipologie di rete per estensione

Le reti possono essere classificate anche in base alla loro **estensione geografica**, cioè all'area nella quale sono distribuiti i dispositivi e i collegamenti. Questa classificazione non descrive soltanto una distanza: al crescere dell'estensione aumentano normalmente il numero di apparati, le organizzazioni coinvolte e la complessità necessaria per trasportare i dati.

Una piccola rete personale può essere gestita interamente da un solo utente. Una rete mondiale, invece, nasce dall'interconnessione di moltissime reti amministrate da soggetti differenti. Per questo motivo una LAN, una MAN e una WAN possono utilizzare alcuni mezzi trasmissivi simili, ma cambiano organizzazione, responsabilità e apparati impiegati.

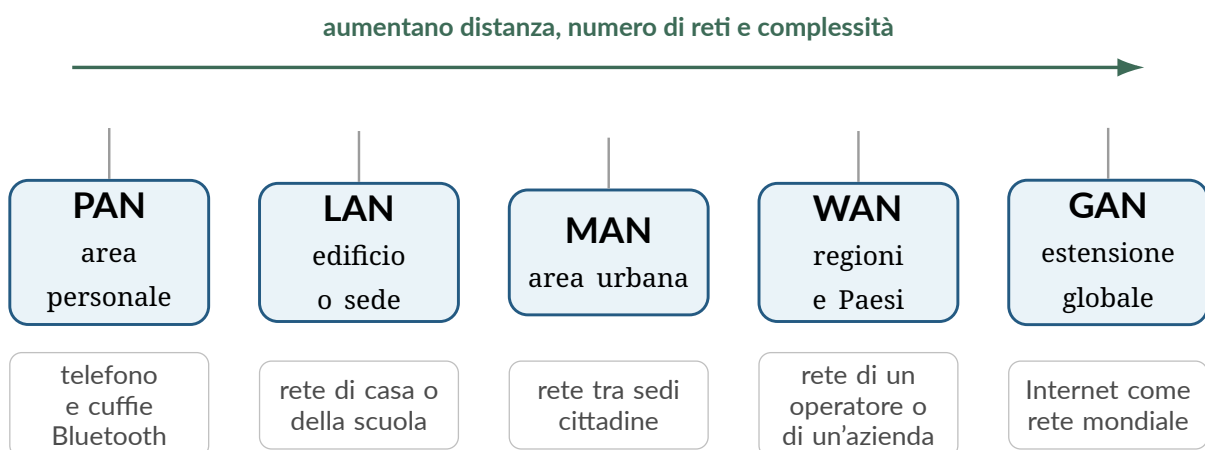


Figura 3.24: Classificazione semplificata delle reti in base all'estensione geografica. Le categorie descrivono scale crescenti, ma i loro confini non sono sempre rigidi.

Approfondimento: categorie utili, non confini assoluti

Le sigle PAN, LAN, MAN, WAN e GAN sono modelli utili per descrivere la scala di una rete. Non esiste però una distanza universale oltre la quale una LAN diventa automaticamente una MAN. Si considerano anche proprietà, gestione, tecnologia e scopo della rete. Una grande azienda, per esempio, può chiamare LAN la rete interna di un'intera sede anche se questa comprende più edifici vicini.

3.5.1 PAN: la rete personale

La sigla **PAN** deriva da *Personal Area Network*. Indica una rete costruita attorno a una persona e ai dispositivi che utilizza direttamente. L'area coperta è normalmente limitata a pochi metri o a un singolo ambiente.

Una PAN può comprendere:

- smartphone e auricolari Bluetooth;
- smartwatch e telefono;
- computer portatile e mouse wireless;
- sensori personali o sanitari collegati a un dispositivo di raccolta;
- collegamenti diretti tramite USB o Wi-Fi Direct.

I mezzi più comuni sono Bluetooth, collegamenti USB e, in alcuni casi, Wi-Fi Direct. Di solito non è necessario uno switch. Il telefono o il computer può svolgere il ruolo di dispositivo centrale e, quando condivide l'accesso a Internet, può comportarsi anche come piccolo router o hotspot.

Vantaggi

- configurazione relativamente semplice;
- mobilità e bassi consumi nei collegamenti Bluetooth;
- costo contenuto;
- collegamento diretto tra dispositivi vicini.

Limiti

- copertura ridotta;
- numero limitato di dispositivi e risorse condivise;
- prestazioni generalmente inferiori a una LAN cablata;
- possibili problemi di compatibilità o associazione.

Esempio: la PAN di Bob

Mentre invia la fotografia ad Alice, Bob indossa auricolari Bluetooth collegati allo smartphone. Telefono e auricolari formano una PAN. Contemporaneamente lo smartphone appartiene anche alla LAN Wi-Fi domestica: lo stesso dispositivo può quindi partecipare a reti di categorie differenti.

3.5.2 LAN: la rete locale

La sigla **LAN** deriva da *Local Area Network*. Una LAN collega dispositivi all'interno di un'area limitata e sotto una gestione comune, come un'abitazione, un laboratorio, un ufficio, un edificio o una sede scolastica.

I mezzi trasmissivi prevalenti sono:

- cavi Ethernet in rame;
- fibra ottica per collegamenti interni ad alta capacità o tra edifici vicini;
- Wi-Fi per i dispositivi mobili.

Gli apparati tipici sono switch, access point e router. Lo switch collega i dispositivi cablati, l'access point permette l'accesso wireless e il router collega la LAN ad altre reti, tra cui Internet.

Vantaggi

- velocità elevate e tempi di risposta ridotti;
- condivisione di stampanti, file e servizi;
- gestione centralizzata degli utenti e della sicurezza;
- possibilità di combinare Ethernet e Wi-Fi.

Limiti

- copertura limitata alla sede;
- necessità di progettare cablaggio e copertura wireless;
- dipendenza dagli apparati centrali;
- gestione più impegnativa quando aumentano utenti e dispositivi.

Esempio: la rete di un laboratorio scolastico

I computer del laboratorio sono collegati mediante cavi Ethernet a uno switch. Una stampante è condivisa tra le postazioni e un router permette di raggiungere Internet. Gli access point della scuola aggiungono la connettività Wi-Fi. L'insieme costituisce una LAN anche se utilizza mezzi trasmissivi diversi.

3.5.3 MAN: la rete metropolitana

La sigla **MAN** deriva da *Metropolitan Area Network*. Indica una rete che collega più sedi distribuite in una città o in un'area urbana. Può essere gestita da un ente, da un'università, da una grande organizzazione o da un operatore di telecomunicazioni.

Una MAN utilizza spesso:

- anelli o dorsali in fibra ottica;
- collegamenti punto-punto tra edifici;
- ponti radio quando il cablaggio non è conveniente;
- router e switch ad alte prestazioni nei punti principali.

Esempi reali

- collegamento tra sedi comunali nella stessa città;
- rete tra edifici e dipartimenti universitari;
- infrastruttura metropolitana di un operatore;
- rete che unisce ospedali o biblioteche cittadine.

Vantaggi

- collega più LAN mantenendo elevate capacità;
- permette di condividere servizi tra sedi vicine;
- può utilizzare percorsi ridondanti;
- riduce la dipendenza da collegamenti Internet pubblici per il traffico interno.

Limiti

- costi di posa, affitto e manutenzione dei collegamenti;
- necessità di accordi e autorizzazioni sul territorio;
- maggiore complessità rispetto a una LAN;
- possibili guasti su infrastrutture esterne agli edifici.

3.5.4 WAN: la rete geografica

La sigla **WAN** deriva da *Wide Area Network*. Una WAN collega reti e sedi distribuite su territori molto ampi: regioni, Stati o continenti. Non unisce soltanto singoli computer, ma spesso intere LAN e MAN.

I mezzi trasmissivi possono comprendere:

- dorsali terrestri in fibra ottica;
- cavi sottomarini tra continenti;

- collegamenti radio e satellitari;
- linee fornite o affittate da operatori differenti.

L'inoltro dei dati richiede numerosi router, che collegano reti amministrate anche da organizzazioni diverse. Rispetto a una LAN, i percorsi sono più lunghi e i tempi di risposta possono essere maggiori.

Esempi reali

- rete di un'azienda con sedi in più Paesi;
- rete nazionale di un operatore telefonico;
- collegamento tra centri dati situati in regioni differenti;
- rete che collega filiali bancarie o punti vendita distribuiti sul territorio.

Curiosità: i cavi sottomarini

Una parte molto importante del traffico internazionale attraversa cavi in fibra ottica posati sui fondali marini. I satelliti sono essenziali in particolari contesti, ma non costituiscono l'unico né il principale mezzo per collegare tutti i continenti. Le grandi capacità richieste da Internet vengono trasportate soprattutto dalle dorsali in fibra.

3.5.5 GAN: una rete di estensione globale

La sigla **GAN**, da *Global Area Network*, viene utilizzata in alcuni contesti didattici per indicare una rete o un insieme di reti con estensione mondiale. L'esempio più importante è Internet.

Internet non è una singola WAN posseduta da un'unica organizzazione. È una **rete di reti** nella quale LAN, MAN e WAN appartenenti a soggetti diversi comunicano grazie a collegamenti e protocolli comuni.

Una rete globale può utilizzare contemporaneamente:

- reti locali Ethernet e Wi-Fi;
- reti mobili 4G e 5G;
- dorsali nazionali e internazionali in fibra;
- cavi sottomarini;
- collegamenti satellitari;
- router e punti di interconnessione tra operatori.

Errore comune: Internet e Web non sono la stessa cosa

Internet è l'infrastruttura mondiale di reti interconnesse. Il Web è uno dei servizi che utilizza Internet, insieme alla posta elettronica, alla messaggistica, alle videochiamate, ai giochi online e a molti altri. È quindi possibile usare Internet anche senza aprire una pagina Web.

3.5.6 Confrontare le tipologie di rete

Tipo	Area indicativa	Mezzi frequenti	Apparati e organizzazione	Esempio
PAN	area personale	Bluetooth, USB, Wi-Fi Direct	collegamento diretto o dispositivo centrale	telefono e smart-watch
LAN	casa, edificio o sede	Ethernet, Wi-Fi, fibra interna	switch, access point e router sotto gestione comune	rete scolastica
MAN	città o area urbana	fibra e ponti radio	apparati che collegano più LAN cittadine	rete universitaria urbana
WAN	regioni, Stati e continenti	fibra terrestre e sottomarina, radio, satelliti	molti router e reti di operatori	rete di un'impresa internazionale
GAN	estensione mondiale	combinazione di tutti i mezzi	interconnessione di moltissime reti autonome	Internet

Tabella 3.4: Confronto generale tra le tipologie di rete classificate per estensione.

Esempio: il viaggio della fotografia attraverso reti differenti

Nel percorso studiato nella [sezione 3.1](#), lo smartphone di Bob comunica con il router attraverso la LAN domestica. Il router inoltra i dati nella WAN dell'operatore. Il messaggio attraversa reti nazionali o internazionali e raggiunge l'infrastruttura del servizio. Infine passa nella rete mobile dell'operatore di Alice. Una sola comunicazione può quindi attraversare più categorie di rete.

Esercizi: sezione 3.5

1. Che cosa indica l'estensione geografica di una rete?
2. Spiegare perché i confini tra LAN, MAN e WAN non sono sempre determinati da una distanza precisa.
3. Scrivere per esteso le sigle PAN, LAN, MAN, WAN e GAN.
4. Classificare come PAN, LAN, MAN, WAN o GAN:
 - a) smartphone collegato a cuffie Bluetooth;
 - b) computer di un laboratorio collegati allo stesso switch;
 - c) sedi comunali collegate in una città;
 - d) filiali di un'azienda distribuite in Europa;
 - e) insieme mondiale delle reti che formano Internet.
5. Quali mezzi trasmissivi sono comuni in una LAN?
6. Quali apparati sono normalmente presenti in una LAN scolastica?
7. Descrivere un possibile esempio di MAN e indicare i mezzi impiegati.
8. Perché una WAN richiede normalmente numerosi router?
9. Qual è il ruolo dei cavi sottomarini nelle comunicazioni mondiali?
10. Spiegare perché Internet può essere descritta come una rete di reti.
11. Distinguere Internet e Web.

Esercizi: sezione 3.5 (continua)

12. Uno smartphone è collegato a uno smartwatch via Bluetooth e a Internet tramite Wi-Fi. A quali reti partecipa?
13. Una scuola possiede due edifici vicini, collegati in fibra, e una sede distaccata in un'altra regione. Classificare i diversi collegamenti e motivare.
14. Completare una tabella personale indicando, per ogni categoria, estensione, mezzi, apparati ed esempio.
15. Tornare alla [figura 3.1](#) e individuare le parti che possono essere considerate LAN, WAN e rete globale.

3.6 Il protocollo IP

Per consegnare una fotografia da Bob ad Alice non basta disporre di cavi, antenne e router. Ogni dispositivo e ogni apparato deve seguire regole comuni. Inoltre i dati devono contenere informazioni che permettano di riconoscere la destinazione.

Il **protocollo IP**, dall'inglese *Internet Protocol*, fornisce un sistema di indirizzamento e stabilisce come i pacchetti vengono inoltrati attraverso reti differenti. È uno dei fondamenti di Internet.

3.6.1 La similitudine del servizio postale

Il funzionamento generale può essere confrontato con una spedizione postale. Per consegnare un pacco sono necessari:

- un mittente;
- un destinatario identificato da un indirizzo;
- un contenuto da trasportare;
- centri di smistamento che scelgono il passaggio successivo;
- una rete di strade, mezzi e uffici.

Nella comunicazione IP si trovano elementi simili:

- il dispositivo mittente prepara i dati;
- i dati vengono inseriti in pacchetti;
- ogni pacchetto contiene un indirizzo IP di origine e uno di destinazione;
- i router svolgono una funzione paragonabile allo smistamento;
- i collegamenti fisici trasportano il pacchetto fino alla rete successiva.

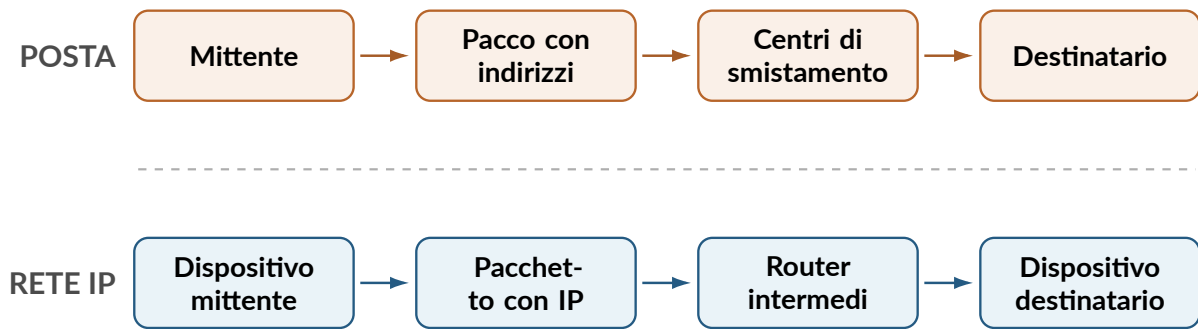


Figura 3.25: Analogia semplificata tra una spedizione postale e l'inoltro di un pacchetto in una rete IP.

L'analogia aiuta a comprendere l'indirizzamento, ma non deve essere interpretata alla lettera. Un pacchetto IP non è un oggetto fisico autonomo: è una struttura di bit trasmessa sotto forma di segnali elettrici, luminosi o radio.

3.6.2 Che cosa fa il protocollo IP

Il protocollo IP permette ai dati di attraversare reti differenti grazie a due funzioni essenziali:

1. **fornire un sistema di indirizzamento:** nei pacchetti vengono indicati l'indirizzo IP del mittente e quello del destinatario;
2. **guidare l'inoltro:** ogni router legge l'indirizzo di destinazione e invia il pacchetto verso la rete successiva.

Il router non deve conoscere in anticipo ogni tratto del viaggio. Deve soltanto scegliere il passaggio successivo più adatto. Il pacchetto viene così inoltrato da un router all'altro fino alla rete del destinatario.

Errore comune: IP non svolge tutte le funzioni della comunicazione

IP si occupa soprattutto degli indirizzi e del percorso tra reti. Altri protocolli gestiscono, per esempio, il controllo della consegna, la sicurezza, la posta elettronica o il trasferimento delle pagine Web.

3.6.3 L'indirizzo IPv4

La versione del protocollo IP ancora molto diffusa è **IPv4**. Un indirizzo IPv4 è formato da **32 bit**, suddivisi in quattro gruppi da 8 bit. Ogni gruppo viene chiamato **ottetto**.

Per rendere l'indirizzo più facile da leggere, gli ottetti vengono convertiti in decimale e separati da punti. Un esempio è:

192.168.1.34

Ogni ottetto può rappresentare un valore compreso tra 0 e 255, perché con 8 bit si ottengono $2^8 = 256$ combinazioni, numerate da 0 a 255.

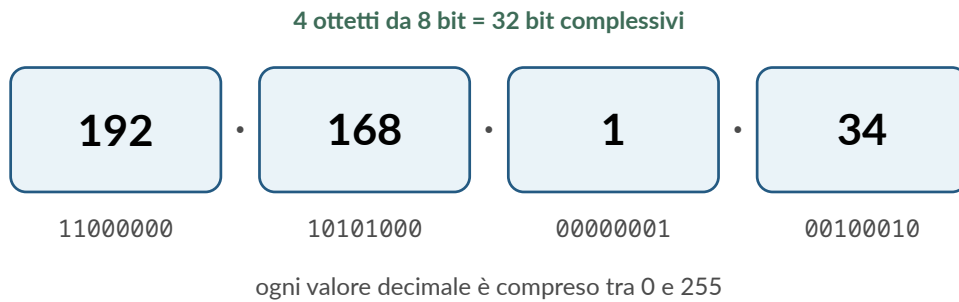


Figura 3.26: Un indirizzo IPv4 è composto da quattro ottetti, per un totale di 32 bit.

Esempio: controllare un indirizzo IPv4

L'indirizzo 172.20.4.8 contiene quattro valori, tutti compresi tra 0 e 255, e rispetta quindi la forma generale di IPv4. La scrittura 192.168.300.4 non è valida, perché 300 non può essere rappresentato con un singolo ottetto.

Approfondimento: indirizzo IP e interfaccia di rete

In modo semplificato si dice che l'indirizzo IP identifica un dispositivo. Più precisamente, viene assegnato a un'interfaccia collegata a una rete. Un computer dotato sia di Ethernet sia di Wi-Fi può avere indirizzi differenti sulle due interfacce. L'indirizzo può inoltre cambiare nel tempo o quando il dispositivo si collega a un'altra rete.

3.6.4 Indirizzi privati e indirizzi pubblici

Se ogni telefono, computer, televisore, sensore e apparecchio domestico utilizzasse sempre un indirizzo IPv4 pubblico esclusivo, gli indirizzi disponibili non sarebbero sufficienti. Per ridurre il numero di indirizzi pubblici necessari vengono utilizzati **indirizzi privati** nelle reti locali e uno o più **indirizzi pubblici** per comunicare su Internet.

Approfondimento: un numero di dispositivi in continua crescita

Non esiste un conteggio unico, perché le stime dipendono da quali apparecchi vengono inclusi. Nel solo settore dell'Internet of Things, escludendo quindi computer, smartphone e tablet, IoT Analytics ha stimato circa 21,1 miliardi di dispositivi connessi alla fine del 2025 e circa 39 miliardi nel 2030. La crescita di sensori, veicoli, elettrodomestici e impianti industriali rende sempre più importante disporre di uno spazio di indirizzamento molto ampio. Questa è una delle ragioni principali del passaggio graduale a IPv6.

Fonte: IoT Analytics, *State of IoT 2025*.

Un indirizzo IP privato:

- viene utilizzato all'interno di una rete locale;
- può essere riutilizzato in molte reti differenti;
- non viene normalmente instradato direttamente su Internet;
- viene assegnato a computer, smartphone, stampanti e altri dispositivi della LAN.

Le principali gamme private IPv4 sono:

Intervallo iniziale	Intervallo finale	Esempio
10.0.0.0	10.255.255.255	10.20.4.18
172.16.0.0	172.31.255.255	172.20.5.7
192.168.0.0	192.168.255.255	192.168.1.34

Tabella 3.5: Intervalli IPv4 riservati alle reti private.

Un **indirizzo IP pubblico** è invece utilizzato per rendere raggiungibile una comunicazione attraverso Internet. In una situazione domestica semplificata, l'ISP assegna l'indirizzo pubblico al collegamento del router.

3.6.5 Il NAT e la condivisione dell'indirizzo pubblico

Il router può permettere a molti dispositivi con indirizzi privati di condividere lo stesso indirizzo pubblico attraverso il **NAT**, acronimo di *Network Address Translation*.

Quando un dispositivo locale invia dati verso Internet, il router:

1. riceve il pacchetto proveniente dall'indirizzo privato;
2. sostituisce o associa l'indirizzo locale al proprio indirizzo pubblico;
3. registra quale dispositivo ha iniziato la comunicazione;
4. inoltra la risposta al dispositivo interno corretto quando questa ritorna.

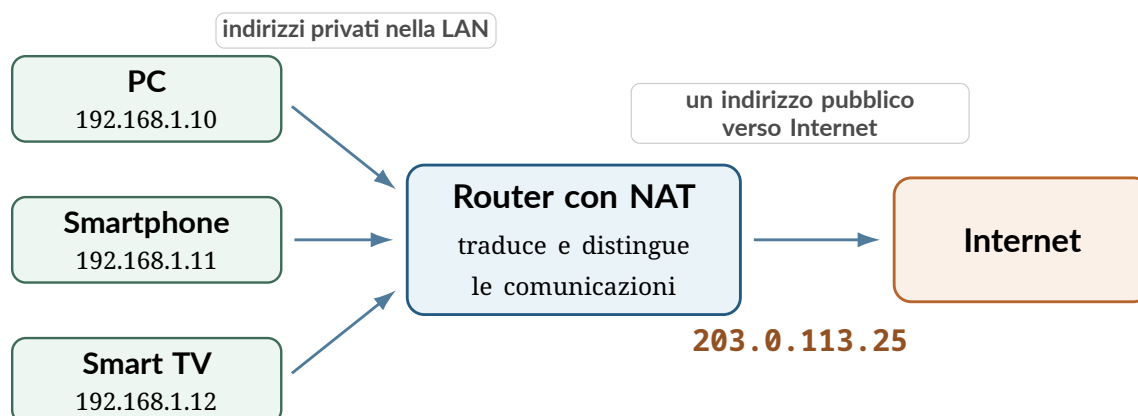


Figura 3.27: Più dispositivi con indirizzi privati possono condividere un indirizzo pubblico attraverso il router e il NAT.

Il NAT ha contribuito a prolungare l'utilizzo di IPv4, ma rende anche meno diretto il collegamento dall'esterno verso un dispositivo interno. Per ricevere connessioni iniziate da Internet possono essere necessarie configurazioni aggiuntive.

Curiosità: indirizzo pubblico condiviso anche tra più utenti

Alcuni operatori utilizzano un ulteriore livello di traduzione, chiamato spesso CGNAT, nel quale più clienti condividono lo stesso indirizzo IPv4 pubblico. In questa situazione l'indirizzo visibile su Internet può non appartenere in modo esclusivo al router dell'utente.

Errore comune: un indirizzo IP non identifica sempre una persona

Un indirizzo IP identifica una connessione o un'interfaccia in un determinato momento, non automaticamente una persona. Può essere condiviso da molti dispositivi, assegnato dinamicamente, tradotto dal NAT o utilizzato da più utenti. Per attribuire un'attività servono informazioni ulteriori e un contesto preciso.

3.6.6 Come viene assegnato un indirizzo

Nelle reti domestiche e scolastiche gli indirizzi vengono spesso assegnati automaticamente. Un servizio chiamato **DHCP**, da *Dynamic Host Configuration Protocol*, comunica al dispositivo i parametri necessari, tra cui:

- indirizzo IP locale;
- informazioni sulla rete alla quale appartiene;
- indirizzo del router da utilizzare per raggiungere altre reti;
- indirizzi dei server DNS.

L'assegnazione automatica evita di configurare manualmente ogni smartphone e computer. In alcune situazioni, come server, stampanti o apparati di rete, può essere utile mantenere un indirizzo prevedibile mediante una configurazione fissa o una prenotazione.

3.6.7 IPv6

Gli indirizzi IPv4 sono 32 bit e quindi, pur essendo numerosi, non sono sufficienti per assegnare facilmente un indirizzo pubblico distinto a ogni dispositivo connesso nel mondo. Per superare questo limite è stato progettato **IPv6**.

Un indirizzo IPv6:

- è formato da 128 bit;
- viene scritto in esadecimale;
- è normalmente suddiviso in otto gruppi separati da due punti;
- offre uno spazio di indirizzamento enormemente più ampio rispetto a IPv4.

Un esempio didattico completo è:

2001:0db8:0000:0000:0000:0000:0000:0034

Gli zeri iniziali di ogni gruppo possono essere omessi e una sequenza continua di gruppi uguali a zero può essere abbreviata una sola volta mediante `::`. Lo stesso indirizzo può quindi essere scritto:

2001:db8::34



IPv4 e IPv6 possono coesistere nella stessa infrastruttura durante la transizione.

Figura 3.28: Confronto essenziale tra la rappresentazione di un indirizzo IPv4 e di un indirizzo IPv6.

IPv6 non consiste soltanto in indirizzi più lunghi. Introduce anche modalità aggiornate di configurazione e gestione della rete. In questa fase è però sufficiente ricordare che il vantaggio principale è la disponibilità di un numero molto maggiore di indirizzi.

Approfondimento: IPv4 e IPv6 possono coesistere

Il passaggio a IPv6 non avviene sostituendo istantaneamente tutta Internet. Molti sistemi utilizzano una configurazione *dual stack*, nella quale IPv4 e IPv6 funzionano contemporaneamente. Il dispositivo sceglie la versione disponibile in base alla rete e al servizio raggiunto.

3.6.8 Seguire il pacchetto di Bob

Riprendendo il viaggio della fotografia:

1. lo smartphone di Bob possiede un indirizzo privato nella LAN domestica;
2. il router usa il NAT e inoltra la comunicazione attraverso l'indirizzo pubblico;
3. i router dell'ISP e di Internet leggono l'indirizzo IP di destinazione e scelgono il passaggio successivo;
4. i pacchetti raggiungono l'infrastruttura del servizio;
5. una nuova comunicazione viene inoltrata verso la rete mobile utilizzata da Alice;
6. lo smartphone di Alice riceve i dati e l'applicazione ricostruisce il messaggio.

L'indirizzo IP permette quindi di individuare la rete e il dispositivo destinatario, mentre router, mezzi trasmissivi e altri protocolli collaborano alla consegna effettiva.

Esempio: lettura ragionata di una rete domestica

Il computer di Bob ha indirizzo 192 . 168 . 1 . 20, lo smartphone 192 . 168 . 1 . 21 e il router 192 . 168 . 1 . 1 nella LAN. Verso Internet il router utilizza l'indirizzo pubblico 203 . 0 . 113 . 25. I tre indirizzi 192 . 168 . 1 . x possono essere usati anche in moltissime altre abitazioni, perché sono privati; l'indirizzo pubblico deve invece essere gestito in modo da essere utilizzabile nella rete esterna.

Esercizi: sezione 3.6

1. Che cosa significa la sigla IP?
2. Quali sono le due funzioni principali del protocollo IP?
3. Completare l'analogia: pacco, indirizzo postale e centro di smistamento corrispondono rispettivamente a quali elementi della rete?
4. Che cosa contiene un pacchetto per poter essere inoltrato verso la destinazione?
5. Perché un router deve conoscere soltanto il passaggio successivo e non l'intero percorso?
6. Da quanti bit è formato un indirizzo IPv4? In quanti ottetti è suddiviso?
7. Perché ciascun ottetto IPv4 può assumere valori da 0 a 255?
8. Indicare quali scritture rispettano la forma di un IPv4 e motivare gli errori:
 - a) 192 . 168 . 1 . 24;
 - b) 10 . 0 . 0;
 - c) 172 . 18 . 256 . 4;
 - d) 8 . 8 . 8 . 8;
 - e) 192 . 168 . 4 . -2.
9. Distinguere indirizzo privato e indirizzo pubblico.
10. Classificare come privati o non appartenenti agli intervalli privati:
 - a) 10 . 4 . 8 . 20;
 - b) 172 . 15 . 4 . 8;
 - c) 172 . 20 . 4 . 8;
 - d) 192 . 168 . 200 . 15;
 - e) 8 . 8 . 8 . 8.
11. Perché gli stessi indirizzi privati possono essere utilizzati in abitazioni differenti?
12. Descrivere il funzionamento generale del NAT.
13. Tre dispositivi domestici condividono lo stesso indirizzo pubblico. Come fa il router a consegnare ogni risposta al dispositivo corretto?
14. Spiegare perché un indirizzo IP non identifica automaticamente una persona.
15. Quale funzione svolge il DHCP?
16. Da quanti bit è formato un indirizzo IPv6 e quale sistema di numerazione utilizza?
17. Riscrivere in forma abbreviata 2001 : 0db8 : 0000 : 0000 : 0000 : 0000 : 0050.
18. Qual è il principale motivo dell'introduzione di IPv6?
19. Che cosa significa configurazione *dual stack*?
20. Disegnare una LAN con tre dispositivi, indirizzi privati, router con NAT e indirizzo pubblico.
21. Ripercorrere il viaggio della fotografia di Bob indicando dove intervengono indirizzo privato, indirizzo pubblico, NAT e router.

3.7 I protocolli di rete

Due persone possono parlare la stessa lingua ma non riuscire comunque a comunicare se parlano contemporaneamente, utilizzano parole con significati differenti o non rispettano un ordine condiviso. Anche i dispositivi collegati in rete hanno bisogno di regole comuni.

Un **protocollo di rete** è un insieme di regole che stabilisce come devono essere formati, inviati, ricevuti e interpretati i messaggi scambiati tra dispositivi e applicazioni. Senza protocolli condivisi, un computer potrebbe ricevere una sequenza di bit ma non sapere come interpretarla.

Definizione: protocollo di rete

Un protocollo di rete definisce le regole utilizzate da due o più sistemi per comunicare. Può stabilire il formato dei messaggi, il loro ordine, il significato dei campi, le risposte attese e il comportamento da adottare quando si verifica un errore.

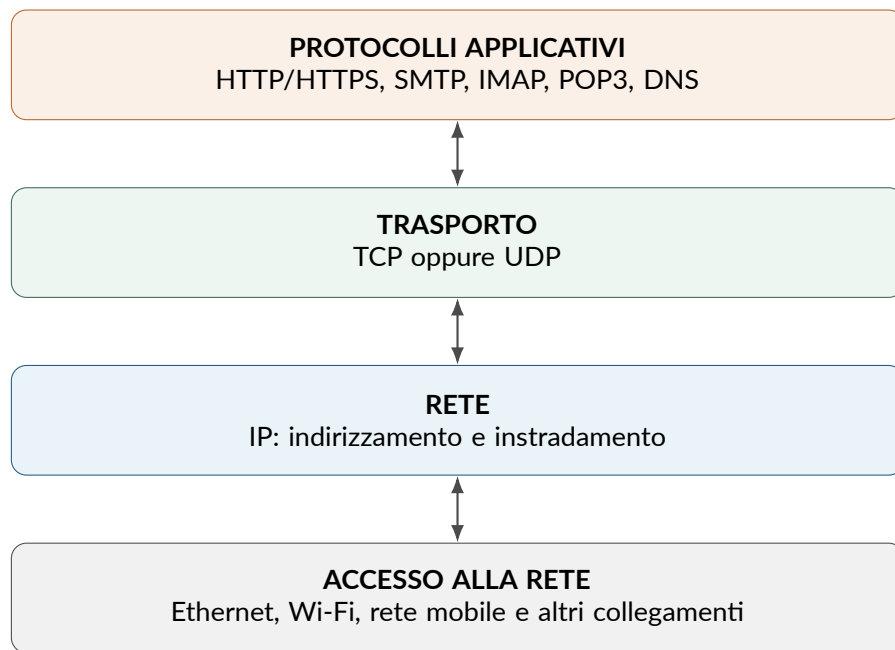
Internet non funziona grazie a un unico protocollo. Durante una sola operazione possono intervenire numerosi protocolli, ciascuno con un compito specifico. Quando Bob invia una fotografia, per esempio, alcuni protocolli gestiscono il collegamento Wi-Fi, altri la consegna dei dati, altri ancora gli indirizzi IP, la sicurezza e il servizio di messaggistica.

3.7.1 Regole diverse per compiti diversi

Per comprendere perché esistono tanti protocolli, si può immaginare una spedizione. Servono regole per preparare il contenuto, scrivere l'indirizzo, trasportare il pacco e confermare la consegna. Allo stesso modo, in una rete sono necessarie regole differenti per:

- collegare fisicamente o tramite onde radio i dispositivi;
- identificare origine e destinazione;
- suddividere e trasportare i dati;
- controllare che i dati arrivino correttamente;
- stabilire il significato della richiesta e della risposta;
- proteggere la comunicazione.

I protocolli vengono spesso organizzati in livelli. Ogni livello utilizza i servizi di quello inferiore e offre un servizio a quello superiore. Questa organizzazione permette di sostituire una tecnologia senza modificare necessariamente tutte le altre: una pagina Web può essere raggiunta sia attraverso Ethernet sia attraverso Wi-Fi o rete mobile.



Ogni livello svolge un compito diverso. La comunicazione funziona perché i protocolli cooperano.

Figura 3.29: Rappresentazione didattica semplificata di alcuni livelli e protocolli che collaborano durante una comunicazione.

Approfondimento: incapsulamento dei dati

Durante l'invio, ogni livello aggiunge ai dati alcune informazioni necessarie al proprio compito. Questo procedimento viene chiamato **incapsulamento**. Il livello IP, per esempio, aggiunge gli indirizzi di origine e destinazione; il collegamento locale aggiunge le informazioni necessarie per il trasferimento sul mezzo utilizzato. Alla ricezione, le informazioni vengono lette e rimosse nell'ordine inverso.

3.7.2 Una panoramica dei protocolli più comuni

La tabella seguente raccoglie alcuni protocolli che verranno incontrati nel capitolo. Non è necessario memorizzarli tutti immediatamente: l'obiettivo è comprendere che ognuno risolve un problema specifico.

Protocollo	Funzione principale	Esempio di utilizzo
Ethernet/Wi-Fi	Trasporto dei dati nella rete locale	Collegamento di un computer allo switch o all'access point
IP	Indirizzamento e instradamento tra reti	Inoltro di un pacchetto verso un server su Internet
TCP	Trasporto affidabile e ordinato	Download di una pagina o di un file
UDP	Trasporto con pochi controlli preliminari	Voce, video in tempo reale o giochi online
DNS	Traduzione dei nomi in indirizzi IP	Ricerca dell'IP associato a un sito
HTTP/HTTPS	Richiesta e trasferimento di risorse Web	Apertura di una pagina nel browser
SMTP	Invio della posta elettronica	Spedizione di un messaggio al server di posta
IMAP/POP3	Consultazione o ricezione dei messaggi	Lettura della casella email da computer e smartphone
DHCP	Assegnazione automatica dei parametri di rete	Configurazione di uno smartphone collegato al Wi-Fi

Tabella 3.6: Alcuni protocolli e il problema principale che risolvono.

Errore comune: Internet non è un unico protocollo

Le espressioni *protocollo Internet* e *protocollo IP* non devono essere confuse con l'intero insieme delle regole utilizzate online. IP è fondamentale, ma da solo non permette di visualizzare una pagina, inviare un'email o garantire una consegna ordinata.

3.7.3 TCP: consegna controllata e ordinata

Il protocollo **TCP**, acronimo di *Transmission Control Protocol*, viene utilizzato quando è importante ricevere tutti i dati correttamente e nello stesso ordine in cui sono stati inviati.

Prima dello scambio dei dati, TCP stabilisce una comunicazione tra le due estremità. Durante il trasferimento:

- identifica la posizione dei byte nel flusso mediante numeri di sequenza;
- usa le conferme di ricezione per controllare l'avanzamento del trasferimento;
- ritrasmette i dati quando rileva una possibile perdita;
- riordina i dati arrivati in una sequenza differente;
- regola la quantità di dati trasmessa per evitare di sovraccaricare la comunicazione.

Questi controlli richiedono tempo e informazioni aggiuntive, ma sono indispensabili quando anche una piccola perdita renderebbe inutilizzabile il contenuto. Il download di un programma o documento deve essere completo, senza parti mancanti.

Esempio: scaricare un documento

L'applicazione invia i byte del documento attraverso TCP, che fornisce un flusso ordinato e affidabile. Il mittente usa le conferme di ricezione e i tempi di attesa per rilevare possibili perdite e ritrasmettere i dati. I byte possono essere consegnati progressivamente all'applicazione ricevente: TCP non aspetta che sia arrivato l'intero file. L'applicazione interpreta il formato del documento e ne gestisce la ricostruzione.

3.7.4 UDP: pochi controlli e minore attesa

Il protocollo **UDP**, acronimo di *User Datagram Protocol*, invia unità di dati senza stabilire una comunicazione controllata come quella di TCP. Non garantisce direttamente che ogni parte arrivi, che arrivi una sola volta o che venga ricevuta nell'ordine corretto.

Questo non significa che UDP sia un protocollo difettoso. È adatto alle situazioni nelle quali:

- è preferibile ricevere subito i dati disponibili;
- una piccola perdita può essere tollerata;
- l'applicazione gestisce autonomamente i controlli necessari;
- la riduzione delle operazioni preliminari è importante.

Durante una conversazione vocale, per esempio, un piccolo frammento arrivato molto in ritardo può essere meno utile di un frammento perso: la conversazione deve continuare in tempo reale.

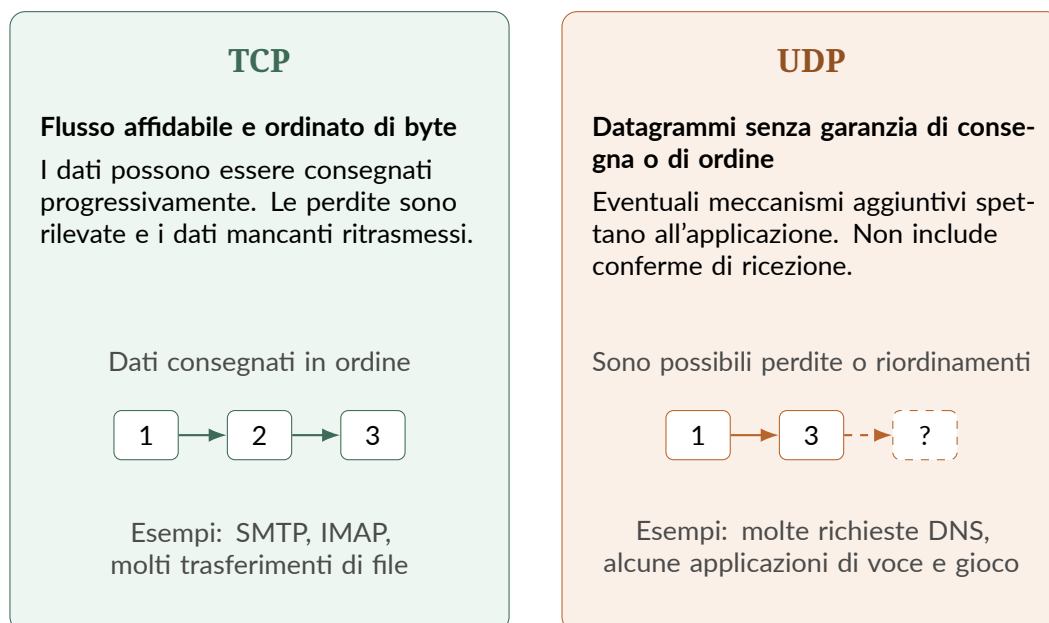


Figura 3.30: Confronto semplificato tra TCP, orientato alla consegna controllata, e UDP, caratterizzato da minori controlli integrati.

Caratteristica	TCP	UDP
Stabilisce una connessione	Sì	No
Controlla consegna e ordine	Sì	Non direttamente
Richiede ritrasmissioni	Sì, quando necessario	Non direttamente
Quantità di controlli	Maggiore	Minore
Utilizzi tipici	Web, email, trasferimento di file	Voce, giochi, streaming dal vivo, DNS

Tabella 3.7: Confronto didattico tra TCP e UDP. L'applicazione può aggiungere ulteriori meccanismi di controllo.

Errore comune: UDP non significa sempre più veloce

UDP introduce meno controlli di TCP, ma le prestazioni reali dipendono dall'applicazione, dalla rete e dal tipo di dati. Un'applicazione che utilizza UDP può comunque aggiungere controlli, conferme e ritrasmissioni. È quindi più corretto dire che UDP offre un servizio più semplice, non che sia automaticamente sempre più veloce.

Curiosità: HTTP/3 utilizza UDP attraverso QUIC

Le prime versioni di HTTP utilizzano normalmente TCP. HTTP/3 utilizza invece QUIC, un protocollo che funziona sopra UDP e integra direttamente meccanismi di affidabilità e sicurezza. Questo mostra che i protocolli possono evolvere e combinare in modo differente le funzioni necessarie.

3.7.5 I protocolli della posta elettronica

La posta elettronica utilizza protocolli differenti per l'invio e per la consultazione dei messaggi. Un'applicazione di posta non consegna normalmente l'email direttamente al dispositivo del destinatario: comunica con uno o più server di posta.

SMTP: inviare i messaggi

SMTP, acronimo di *Simple Mail Transfer Protocol*, viene utilizzato per inviare i messaggi:

- dall'applicazione del mittente al proprio server di posta;
- tra il server del mittente e quello del destinatario.

Il server del destinatario conserva il messaggio nella casella corretta, in attesa che venga consultato.

IMAP: sincronizzare la casella

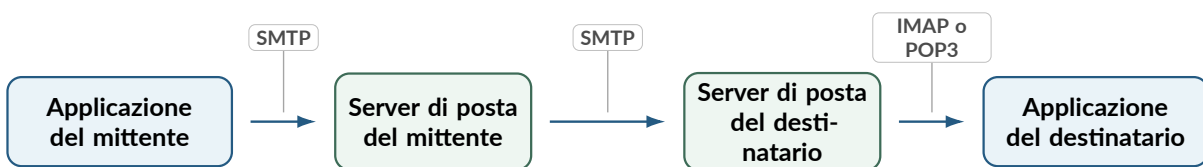
IMAP, acronimo di *Internet Message Access Protocol*, permette di consultare i messaggi mantenendoli sul server. Le modifiche vengono sincronizzate tra dispositivi: se un messaggio viene letto, spostato o eliminato sul telefono, la modifica può comparire anche sul computer.

È particolarmente adatto quando la stessa casella viene utilizzata da più dispositivi.

POP3: scaricare i messaggi

POP3, acronimo di *Post Office Protocol version 3*, è stato progettato principalmente per scaricare i messaggi dal server al dispositivo. Nella configurazione tradizionale i messaggi potevano essere rimossi dal server dopo il download; molti programmi moderni permettono tuttavia di conservarne una copia.

POP3 è semplice, ma offre una sincronizzazione meno completa rispetto a IMAP.



Il messaggio viene inviato tramite SMTP; il destinatario lo consulta normalmente con IMAP o POP3.

Figura 3.31: Percorso semplificato di un'email e ruolo dei protocolli SMTP, IMAP e POP3.

Esempio: la stessa casella su telefono e computer

Alice legge un messaggio sullo smartphone e lo sposta nella cartella *Scuola*. Se la casella è configurata tramite IMAP, il computer può mostrare la stessa organizzazione. Con POP3, invece, la gestione tende a essere più legata alla copia scaricata sul singolo dispositivo.

3.7.6 Un accenno alla posta elettronica certificata

La **PEC**, posta elettronica certificata, è un sistema regolamentato che aggiunge alla posta elettronica ricevute e controlli specifici. Il messaggio passa attraverso gestori del servizio, che producono normalmente una ricevuta di accettazione e una ricevuta di avvenuta consegna.

In modo semplificato:

1. il mittente invia il messaggio dalla propria casella PEC;
2. il gestore del mittente controlla e prende in carico il messaggio;
3. viene prodotta una ricevuta di accettazione;
4. il messaggio viene trasmesso al gestore del destinatario;
5. dopo la consegna viene prodotta una ricevuta.

La PEC non deve essere confusa con una normale email che contiene semplicemente una firma o un allegato. Il valore delle ricevute deriva dal funzionamento del sistema certificato e dalle regole che lo disciplinano.

Approfondimento: contenuto, consegna e lettura

La ricevuta di avvenuta consegna indica che il messaggio è stato depositato nella casella del destinatario. Non dimostra necessariamente che il destinatario lo abbia già aperto e letto. Anche nella comunicazione digitale è quindi importante distinguere tra invio, consegna e lettura.

Errore comune: SMTP, IMAP e POP3 non sono servizi concorrenti

SMTP viene utilizzato per inviare i messaggi; IMAP e POP3 servono invece a consultarli o scaricarli. Una stessa casella può quindi utilizzare SMTP in uscita e IMAP in entrata.

Esercizi: sezione 3.7

1. Che cos'è un protocollo di rete?
2. Perché due dispositivi non possono comunicare soltanto inviando una sequenza di bit priva di regole?
3. Indicare almeno quattro aspetti che possono essere definiti da un protocollo.
4. Perché durante una sola comunicazione vengono utilizzati più protocolli?
5. Associare ciascun compito al protocollo più adatto: indirizzamento, traduzione dei nomi, invio email, consultazione email, richiesta di una pagina Web.
6. Che cosa significa organizzare i protocolli in livelli?
7. Spiegare con parole proprie il concetto di incapsulamento.
8. Quale funzione svolge TCP?
9. Quali problemi può correggere TCP durante il trasferimento?
10. Perché TCP è adatto al download di un programma?
11. Quali garanzie non vengono offerte direttamente da UDP?
12. Indicare due situazioni nelle quali una piccola perdita può essere preferibile a un grande ritardo.
13. Correggere l'affermazione: "UDP è sempre più veloce di TCP".
14. Completare la tabella indicando TCP o UDP:
 - a) trasferimento di un documento;
 - b) conversazione vocale in tempo reale;
 - c) download di un aggiornamento;
 - d) gioco online sensibile alla latenza.
15. Quale protocollo viene usato per inviare un'email?
16. Qual è la differenza principale tra IMAP e POP3?
17. Un utente vuole mantenere sincronizzata la stessa casella su telefono, tablet e computer. Quale protocollo è più adatto e perché?
18. Ricostruire il percorso di un'email dal mittente al destinatario indicando dove intervengono SMTP e IMAP o POP3.
19. Quali ricevute vengono normalmente prodotte durante l'invio di una PEC?
20. Spiegare perché la consegna nella casella non coincide necessariamente con la lettura.

Esercizi: sezione 3.7 (continua)

21. Preparare uno schema che mostri almeno cinque protocolli coinvolti quando si apre una pagina Web attraverso il Wi-Fi domestico.

3.8 Navigare sul Web

Aprire un sito sembra un'operazione immediata: si scrive un indirizzo, si preme Invio e la pagina compare sullo schermo. Dietro questa azione collaborano il browser, il sistema DNS, la rete IP, i protocolli di trasporto, i meccanismi di sicurezza e il server Web.

Il **Web** è uno dei servizi disponibili su Internet. Internet è l'infrastruttura di reti e protocolli che trasporta i dati; il Web utilizza questa infrastruttura per collegare pagine, immagini, video, applicazioni e altre risorse.

Errore comune: Web e Internet non sono sinonimi

Internet comprende molti servizi, tra cui posta elettronica, messaggistica, trasferimento di file, giochi online e Web. Il Web è quindi una parte di Internet, non l'intera rete.

3.8.1 Che cos'è un URL

Per richiedere una risorsa è necessario indicare dove si trova e quale metodo deve essere utilizzato per raggiungerla. Queste informazioni vengono espresse mediante un **URL**, acronimo di *Uniform Resource Locator*.

Un esempio semplice è:

`https://www.scuola.example/reti/pagina.html`

In questo capitolo si considerano tre parti principali:

- **schema o protocollo**: indica il metodo da utilizzare, per esempio `https`;
- **nome dell'host**: identifica il server o il servizio, per esempio `www.scuola.example`;
- **percorso**: indica la risorsa richiesta, per esempio `/reti/pagina.html`.



Figura 3.32: Scomposizione semplificata di un URL in protocollo, host e percorso.

Un URL può essere anche più breve. Nell'indirizzo

`https://www.scuola.example`

sono presenti soltanto il protocollo e il nome dell'host; il server sceglie quindi la pagina iniziale da restituire.

Esempio: leggere un URL

Si consideri:

`https://biblioteca.example/catalogo/libro.html`

Le sue parti sono:

- `https:` protocollo;
- `biblioteca.example:` nome dell'host;
- `/catalogo/libro.html:` percorso della risorsa.

Errore comune: dominio e URL non sono la stessa cosa

Il nome del dominio è soltanto una parte dell'URL. L'URL completo può indicare anche il protocollo e il percorso della risorsa.

3.8.2 HTTP: richieste e risposte

HTTP, acronimo di *Hypertext Transfer Protocol*, definisce il modo in cui un client Web, normalmente il browser, richiede risorse a un server e riceve le risposte.

Il funzionamento essenziale segue il modello richiesta-risposta:

1. il browser invia una **richiesta HTTP**;
2. la richiesta indica la risorsa desiderata e altre informazioni;
3. il server elabora la richiesta;
4. il server restituisce una **risposta HTTP**;
5. la risposta contiene uno stato, alcune informazioni e, quando disponibile, il contenuto richiesto.



Figura 3.33: Il browser richiede una risorsa e il server restituisce una risposta HTTP.

Una pagina non è sempre un unico file. Dopo aver ricevuto il documento principale, il browser può effettuare altre richieste per ottenere:

- immagini;
- fogli di stile;
- caratteri tipografici;
- script;
- video e altri contenuti.

Per questo l'apertura di una pagina può produrre numerose richieste HTTP.

I codici di stato

Ogni risposta HTTP contiene un **codice di stato**, che comunica l'esito della richiesta. Alcuni esempi comuni sono:

Codice	Significato generale
200	La richiesta è stata completata correttamente.
301 / 302	La risorsa si trova temporaneamente o definitivamente a un altro indirizzo.
403	Il server ha compreso la richiesta ma non autorizza l'accesso.
404	La risorsa richiesta non è stata trovata.
500	Il server ha incontrato un errore interno.

Tabella 3.8: Alcuni codici di stato HTTP.

Curiosità: il numero 404

La scritta *404 Not Found* non indica necessariamente che Internet non funziona. Significa che il server è stato raggiunto e ha risposto, ma non ha trovato la risorsa richiesta all'indirizzo indicato.

3.8.3 HTTPS: proteggere la comunicazione

HTTP da solo non protegge il contenuto durante il trasferimento. Su una rete non affidabile, un soggetto intermedio potrebbe tentare di osservare o modificare i dati scambiati.

HTTPS indica l'uso di HTTP all'interno di una comunicazione protetta mediante **TLS**, un insieme di meccanismi crittografici. HTTPS offre tre proprietà fondamentali:

- **riservatezza**: il contenuto viene cifrato e non è leggibile facilmente da chi intercetta il traffico;
- **integrità**: eventuali modifiche durante il trasferimento possono essere rilevate;
- **autenticazione**: il browser verifica, tramite un certificato digitale, l'identità dichiarata dal server.

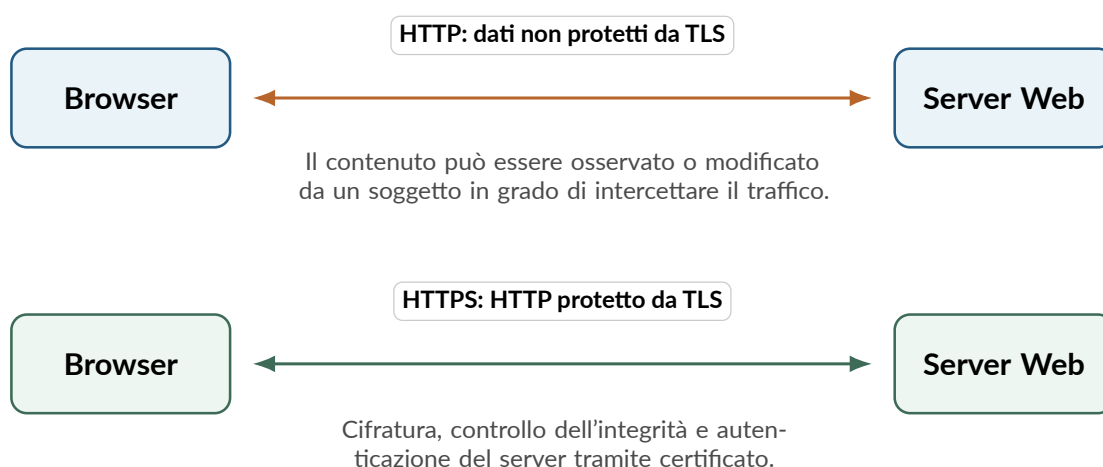


Figura 3.34: HTTP trasferisce richieste e risposte; HTTPS protegge la comunicazione mediante TLS.

Quando la connessione è protetta, il browser mostra normalmente il protocollo `https` e un'indicazione grafica associata alla sicurezza della connessione. È comunque necessario controllare attentamente il nome del sito.

Errore comune: HTTPS non rende automaticamente affidabile un sito

HTTPS protegge il collegamento con il server indicato e permette di verificarne l'identità tecnica. Un sito fraudolento può però utilizzare a sua volta HTTPS. Il lucchetto non garantisce che le informazioni pubblicate siano vere, che il venditore sia onesto o che il contenuto sia privo di rischi.

Esempio: credenziali inviate su una rete Wi-Fi

Quando si inseriscono nome utente e password, HTTPS cifra la comunicazione tra browser e server. Senza questa protezione, i dati potrebbero essere esposti lungo il percorso. La cifratura non protegge però da una password debole, da un dispositivo infetto o dall'inserimento delle credenziali in un sito falso.

3.8.4 DNS: dai nomi agli indirizzi IP

Gli esseri umani ricordano più facilmente nomi come `www.scuola.example`; i router, invece, inoltrano i pacchetti utilizzando indirizzi IP. Il **DNS**, acronimo di *Domain Name System*, collega questi due modi di identificare una risorsa.

In modo semplificato, il DNS funziona come un elenco che associa:

`www.scuola.example` → `203.0.113.40`

Quando il computer deve conoscere l'indirizzo associato a un nome:

1. controlla se il risultato è già memorizzato localmente;
2. se necessario interroga il server DNS configurato;
3. il sistema DNS cerca l'informazione nei propri archivi o presso altri server;
4. restituisce uno o più indirizzi IP;
5. il computer può iniziare la comunicazione con il server indicato.

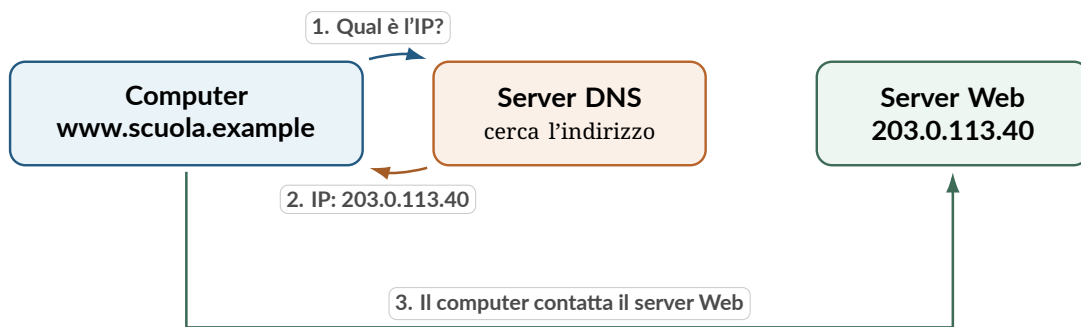


Figura 3.35: Risoluzione DNS semplificata: dal nome del sito all'indirizzo IP del server.

Il DNS è un sistema distribuito: non esiste un unico computer contenente tutti i nomi di Internet. Numerosi server collaborano secondo una struttura gerarchica. I risultati vengono inoltre memorizzati temporaneamente nella **cache**, riducendo il numero di nuove richieste.

Approfondimento: un nome può corrispondere a più indirizzi

Un grande servizio online può associare lo stesso nome a più indirizzi IP. In questo modo le richieste possono essere distribuite tra server differenti o indirizzate verso un centro dati vicino all'utente. Al contrario, più nomi possono anche condurre allo stesso indirizzo IP.

Errore comune: il DNS non trasporta la pagina

Il DNS fornisce l'indirizzo IP associato al nome. La pagina viene poi richiesta al server Web attraverso altri protocolli. Ottenere l'indirizzo è simile a cercare un numero telefonico: la ricerca del numero non coincide con la conversazione successiva.

3.8.5 Che cosa accade quando si preme Invio

È ora possibile ricostruire il percorso completo. La sequenza seguente è semplificata: molte operazioni possono essere già state svolte, memorizzate nella cache o eseguite contemporaneamente.

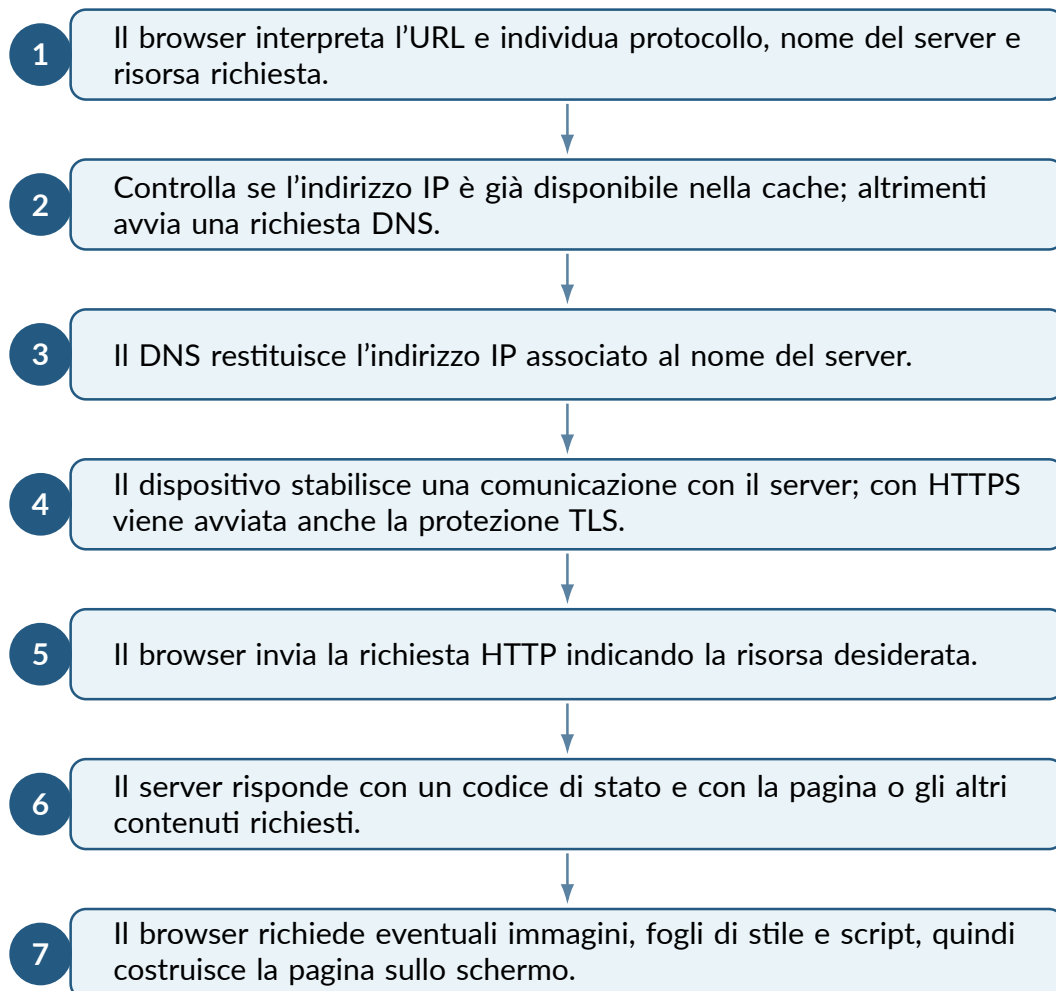


Figura 3.36: Principali passaggi che avvengono dopo aver inserito un URL e premuto Invio.

Le fasi principali sono:

1. **interpretazione dell'URL:** il browser riconosce protocollo, host e percorso;
2. **ricerca dell'indirizzo:** viene controllata la cache e, se necessario, interrogato il DNS;
3. **individuazione del server:** il DNS restituisce l'indirizzo IP;
4. **apertura della comunicazione:** il dispositivo contatta il server attraverso la rete IP;
5. **protezione TLS:** con HTTPS viene verificato il certificato e vengono stabilite le chiavi di cifratura;
6. **richiesta HTTP:** il browser richiede la risorsa indicata;
7. **risposta del server:** il server restituisce uno stato e i dati disponibili;

8. **richieste aggiuntive:** il browser scarica immagini, stili, script e altre risorse;
9. **visualizzazione:** i contenuti vengono interpretati e composti sullo schermo.

Esempio: perché il nome si apre ma un'immagine manca

Il documento principale può essere ricevuto correttamente, mentre una delle immagini può trovarsi a un URL errato o su un altro server momentaneamente irraggiungibile. Una pagina può quindi comparire solo in parte, perché è composta da più risorse e più richieste.

3.8.6 Interpretare alcuni problemi di navigazione

Le fasi studiate aiutano a interpretare alcuni problemi comuni:

Situazione osservata	Possibile interpretazione iniziale
Nessun sito è raggiungibile	Collegamento locale, router, ISP o configurazione di rete
Un sito non si apre, gli altri sì	Problema del sito, del DNS o del percorso verso quel servizio
Compare un errore 404	Server raggiunto, ma risorsa non trovata
Compare un avviso sul certificato	Problema nell'identità o nella validità della protezione HTTPS
Il nome non funziona ma l'IP sì	Possibile problema di risoluzione DNS
La pagina compare senza immagini	Alcune risorse secondarie non sono state raggiunte o caricate

Tabella 3.9: Esempi di problemi di navigazione e prime ipotesi da verificare.

Queste indicazioni non costituiscono una diagnosi certa. Lo stesso sintomo può avere cause differenti, ma la scomposizione in fasi permette di eseguire controlli con un ordine logico.

Approfondimento: cache del browser e del DNS

Per ridurre tempi e traffico, il browser può conservare copie di alcune risorse e il sistema può memorizzare temporaneamente i risultati DNS. La cache accelera le richieste successive, ma può talvolta mostrare una versione non aggiornata o conservare per breve tempo un'informazione ormai cambiata.

Esercizi: sezione 3.8

1. Qual è la differenza tra Internet e Web?
2. Che cosa significa la sigla URL?
3. Quali sono le tre parti principali di un URL considerate nel capitolo?
4. Scomporre il seguente URL:
<https://laboratorio.example/materiali/reti.pdf>.

Esercizi: sezione 3.8 (continua)

5. Qual è la differenza tra nome di dominio e URL completo?
6. Quale funzione svolge HTTP?
7. Descrivere il modello richiesta-risposta.
8. Perché l'apertura di una pagina può richiedere numerose richieste HTTP?
9. Associare i codici 200, 404 e 500 al significato corretto.
10. Perché un errore 404 dimostra che il server è stato almeno raggiunto?
11. Quale protezione aggiunge HTTPS a HTTP?
12. Spiegare riservatezza, integrità e autenticazione.
13. Perché il simbolo del lucchetto non garantisce che il contenuto del sito sia affidabile?
14. Che cosa significa la sigla DNS?
15. Quale problema risolve il DNS?
16. Correggere l'affermazione: "Il DNS scarica la pagina dal server".
17. Perché i risultati DNS vengono memorizzati temporaneamente nella cache?
18. Ordinare le seguenti fasi: risposta HTTP, ricerca DNS, interpretazione dell'URL, visualizzazione, connessione al server.
19. Descrivere ciò che avviene dopo aver digitato un URL HTTPS e premuto Invio.
20. Un sito restituisce il codice 404. Quali parti del percorso hanno probabilmente funzionato?
21. Nessun sito è raggiungibile da un computer, ma gli altri dispositivi della stessa LAN navigano. Quali controlli iniziali si potrebbero eseguire?
22. Il nome di un sito non funziona, ma il collegamento diretto al suo indirizzo IP sì. Quale servizio potrebbe avere un problema?
23. Disegnare uno schema che colleghi browser, DNS e server Web, numerando le richieste.
24. Individuare almeno cinque protocolli che possono intervenire durante l'apertura di una pagina HTTPS attraverso Wi-Fi.

3.9 Client-server e reti distribuite

Nelle sezioni precedenti sono stati studiati i mezzi attraverso cui viaggiano i dati, gli apparati che li inoltrano, gli indirizzi e i protocolli utilizzati nella comunicazione. Resta ora da osservare *come vengono organizzati i dispositivi che partecipano a un servizio di rete*.

Quando si apre un sito, si consulta il registro elettronico o si invia un messaggio, non tutti i computer svolgono lo stesso compito. Alcuni richiedono una risorsa, altri la mettono a disposizione. In altri sistemi, invece, i dispositivi possono assumere entrambi i ruoli e scambiarsi direttamente parti di un contenuto.

I due modelli fondamentali presentati in questa sezione sono:

- il modello **client-server**, basato su servizi forniti da uno o più server;
- il modello **peer-to-peer**, nel quale i nodi collaborano tra loro con ruoli più simili.

3.9.1 Il modello client-server

Nel modello **client-server** i dispositivi assumono due ruoli distinti:

- il **client** avvia la comunicazione e richiede una risorsa o un servizio;
- il **server** riceve la richiesta, la elabora e restituisce una risposta.

Il termine inglese *client* significa letteralmente “cliente”. Come un cliente entra in un negozio e richiede un prodotto, un programma client contatta un server per ottenere una pagina, un file, un dato o l’esecuzione di un’operazione.

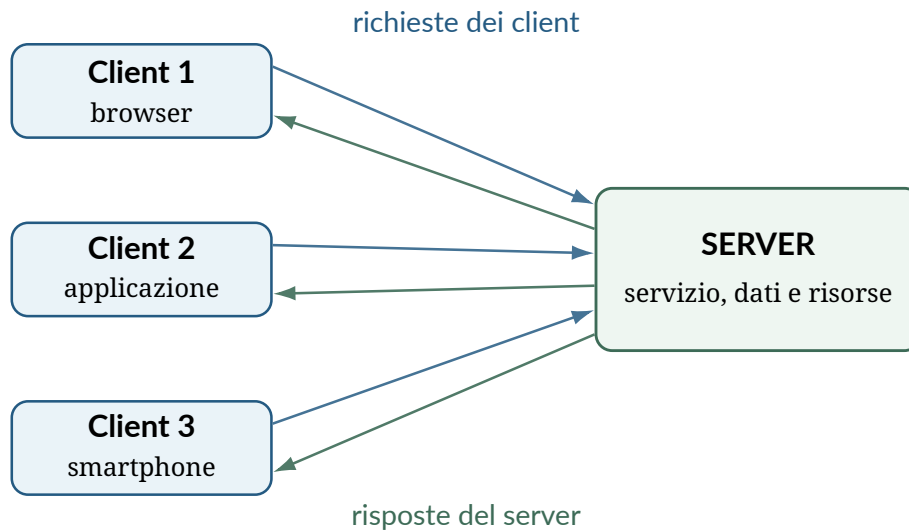


Figura 3.37: Nel modello client-server più client inviano richieste al server, che elabora le operazioni e restituisce le risposte.

Il client e il server comunicano mediante protocolli condivisi. Per esempio:

- un browser è client di un server Web e utilizza HTTP o HTTPS;
- un’applicazione di posta è client di server SMTP e IMAP;
- un’applicazione per il cloud è client di un servizio che conserva e sincronizza i file;
- un programma gestionale può interrogare un server che contiene un database;
- un’applicazione di messaggistica comunica con l’infrastruttura del proprio fornitore.

Definizione: client

Un client è un programma o un dispositivo che avvia una comunicazione per richiedere un servizio, una risorsa o un’operazione a un server.

Definizione: server

Un server è un programma o un sistema che rimane disponibile per ricevere richieste dai client, elaborarle e fornire le risposte previste dal servizio.

Client e server sono ruoli

Le parole *client* e *server* non indicano necessariamente due particolari tipi di computer. Indicano soprattutto il **ruolo svolto durante una comunicazione**.

Un notebook può funzionare come client quando apre una pagina Web, ma può anche eseguire un piccolo programma server per condividere una risorsa nella rete locale. Allo stesso modo, un server professionale è normalmente progettato per fornire servizi a molti utenti, ma può comportarsi da client quando richiede aggiornamenti o interroga un altro server.

Errore comune: il server non è soltanto un grande computer

Un server può essere un computer fisico, una macchina virtuale, un insieme di programmi o un gruppo di sistemi distribuiti. Ciò che lo definisce è la funzione svolta: ricevere richieste e fornire un servizio.

Il ciclo richiesta-risposta

La comunicazione client-server segue spesso una sequenza di questo tipo:

1. il client individua il server mediante un nome o un indirizzo IP;
2. apre una comunicazione utilizzando i protocolli richiesti;
3. invia una richiesta;
4. il server controlla la richiesta e, quando necessario, identifica l'utente;
5. il server legge dati, esegue calcoli o interroga altri sistemi;
6. prepara una risposta;
7. restituisce il risultato al client;
8. il client presenta il contenuto all'utente.

Il server può rispondere immediatamente oppure impiegare più tempo, in base al tipo di operazione. Una richiesta per una semplice immagine è diversa dalla ricerca di tutti i voti di uno studente, dall'elaborazione di un pagamento o dalla generazione di un grafico partendo da migliaia di misurazioni.

Esempio: consultare il registro elettronico

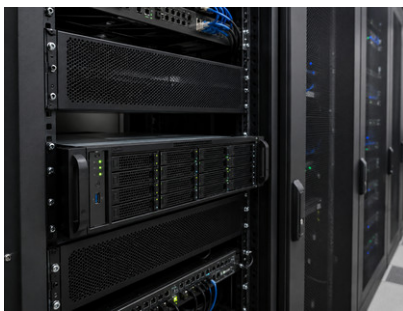
Quando si apre il registro elettronico, il browser o l'applicazione svolge il ruolo di client. Dopo l'accesso, il client richiede al server i dati autorizzati per quell'utente. Il server controlla le credenziali, consulta gli archivi e restituisce le informazioni. Il client organizza infine i dati sullo schermo.

3.9.2 Un server può servire molti client

Un singolo servizio può essere utilizzato contemporaneamente da numerosi client. Il server deve quindi:

- distinguere le diverse richieste;
- gestire più comunicazioni nello stesso periodo;
- controllare quali utenti sono autorizzati;
- proteggere i dati;
- evitare che un errore di un client interrompa il servizio per tutti;
- distribuire correttamente le risorse disponibili.

Per servizi di piccole dimensioni può essere sufficiente un solo computer. I grandi servizi online, invece, utilizzano normalmente molti sistemi coordinati tra loro. La parola *server* può quindi indicare l'intera infrastruttura logica, anche quando dietro il servizio sono presenti numerosi computer.



Server in un centro dati



Dispositivi client

Figura 3.38: Server e client possono avere forme molto diverse: il modello descrive i ruoli svolti nella comunicazione.

Centri dati e servizi distribuiti

Un **centro dati**, o *data center*, è un ambiente progettato per ospitare server, apparati di rete, sistemi di alimentazione, raffreddamento e protezione. I grandi fornitori possono utilizzare centri dati in più città o Paesi.

Distribuire un servizio tra più server permette di:

- gestire un numero maggiore di richieste;
- ridurre i tempi di risposta scegliendo una sede vicina all'utente;
- continuare a funzionare quando un singolo server si guasta;
- eseguire manutenzioni senza interrompere completamente il servizio;
- conservare copie ridondanti dei dati.

Un apparato chiamato **bilanciatore di carico** può distribuire le richieste tra più server disponibili. Per una trattazione introduttiva è sufficiente ricordare che il client vede un unico servizio, mentre dietro possono collaborare molti sistemi.

Approfondimento: server fisico, macchina virtuale e cloud

Un server fisico è un computer reale. Su uno stesso computer fisico possono però essere eseguite più *macchine virtuali*, ciascuna con un proprio sistema operativo e propri servizi. Nel cloud le risorse vengono fornite e gestite attraverso grandi infrastrutture, permettendo di aumentare o ridurre capacità, memoria e potenza in base alle necessità.

3.9.3 Vantaggi e limiti del modello client-server

Il modello client-server è molto diffuso perché permette una gestione centralizzata dei servizi.

Vantaggi principali

- **gestione centralizzata:** dati, permessi e aggiornamenti possono essere amministrati in un punto controllato;
- **controllo degli accessi:** il server può verificare identità e autorizzazioni;
- **coerenza dei dati:** gli utenti consultano archivi condivisi e aggiornati;
- **backup:** le copie di sicurezza possono essere organizzate sull'infrastruttura centrale;
- **servizi disponibili da più dispositivi:** lo stesso account può essere usato da computer, tablet e smartphone;
- **manutenzione più semplice per l'utente:** molte modifiche vengono svolte sul server senza intervenire su ogni client.

Limiti principali

- **dipendenza dal server:** se il servizio centrale non è raggiungibile, i client possono non funzionare;
- **punto di guasto:** un'infrastruttura non ridondata può interrompersi per un singolo problema;
- **concentrazione dei dati:** un attacco o un errore sul server può coinvolgere molti utenti;
- **costi di gestione:** server, connettività, sicurezza e manutenzione richiedono risorse;
- **fiducia nel fornitore:** l'utente affida al gestore disponibilità, protezione e trattamento dei dati.

Esempio: file conservato soltanto nel cloud

Un documento presente esclusivamente su un servizio cloud può essere consultato da più dispositivi, ma diventa temporaneamente irraggiungibile se il servizio o la connessione non funzionano. Una copia locale può ridurre questa dipendenza, purché venga mantenuta aggiornata e protetta.

Errore comune: centralizzato non significa necessariamente fragile

Un servizio client-server può utilizzare molti server, collegamenti ridondanti e copie dei dati. Il modello è centralizzato dal punto di vista del controllo del servizio, ma l'infrastruttura fisica può essere distribuita e molto resistente ai guasti.

3.9.4 Il modello peer-to-peer

Nel modello **peer-to-peer**, abbreviato in **P2P**, i dispositivi partecipanti sono chiamati **peer**, cioè “pari”. Ogni peer può richiedere risorse agli altri e, nello stesso tempo, mettere a disposizione le proprie.

Non esiste necessariamente un unico server centrale che contiene tutto il servizio. Le risorse e il lavoro possono essere distribuiti tra i nodi partecipanti.

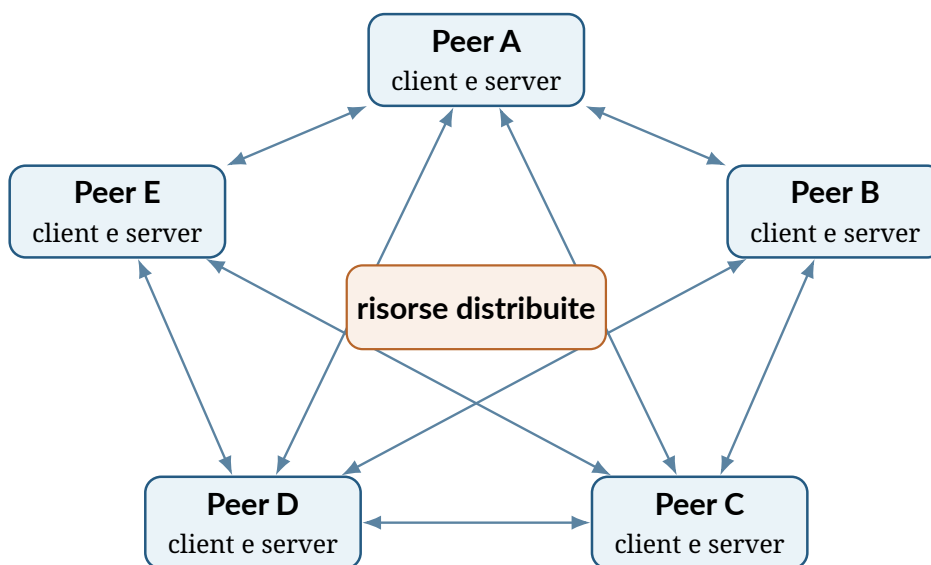


Figura 3.39: In una rete peer-to-peer ogni nodo può ricevere e offrire risorse agli altri peer.

Definizione: peer-to-peer

Una rete peer-to-peer è un sistema nel quale i nodi partecipanti possono comunicare e condividere risorse assumendo ruoli simili, senza dipendere per ogni operazione da un unico server centrale.

In una rete P2P un nodo può essere contemporaneamente:

- client, perché richiede dati;
- server, perché invia dati ad altri peer.

Questo non significa che ogni sistema P2P sia completamente privo di componenti centrali. Alcune reti utilizzano un server iniziale per registrare gli utenti, cercare i peer o facilitare il primo collegamento. Una volta individuati, tuttavia, i nodi possono scambiarsi direttamente le risorse.

Vantaggi del peer-to-peer

- il carico può essere distribuito tra molti dispositivi;
- aumentando i peer può aumentare anche la quantità di risorse offerte;
- non è sempre necessario conservare tutti i dati in un solo punto;
- più nodi possono fornire contemporaneamente parti della stessa risorsa;
- il sistema può continuare a funzionare anche se alcuni peer si disconnettono.

Limiti del peer-to-peer

- i peer possono collegarsi e scollegarsi in qualsiasi momento;
- la disponibilità di una risorsa dipende dai nodi che la possiedono;
- qualità e velocità possono variare;
- il controllo centralizzato è più difficile;
- ogni nodo deve proteggersi dai dati e dai peer con cui comunica;
- gli indirizzi IP dei partecipanti possono essere visibili agli altri nodi connessi.

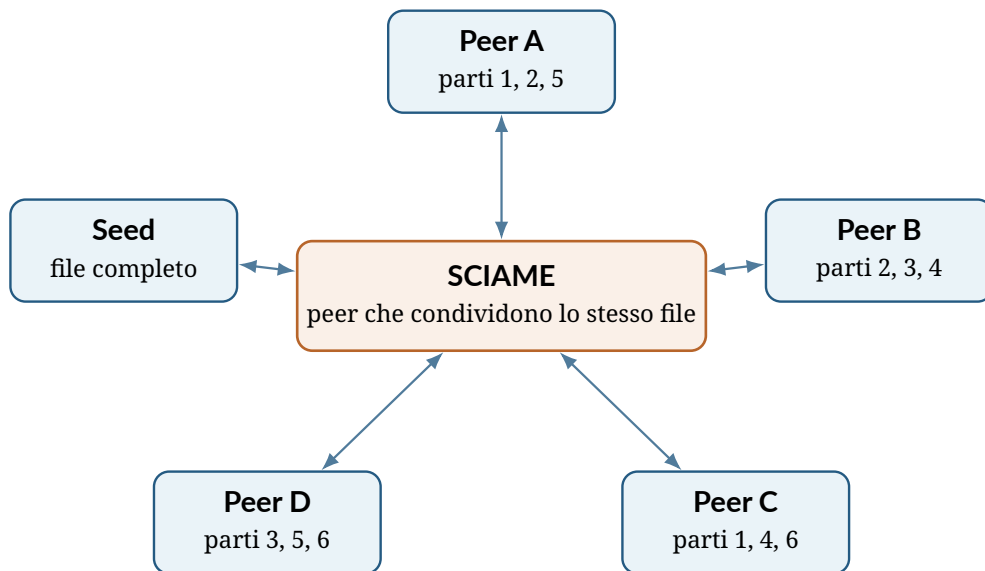
Errore comune: P2P non significa anonimato

Il fatto che un sistema sia distribuito non nasconde automaticamente l'identità o l'indirizzo IP dei partecipanti. In molte reti P2P i peer devono conoscere gli indirizzi degli altri nodi per potersi collegare direttamente.

3.9.5 BitTorrent: distribuire un file tra molti peer

BitTorrent è un protocollo peer-to-peer progettato per distribuire file. Invece di scaricare necessariamente l'intero contenuto da un solo server, il file viene suddiviso in molte **parti**. Un peer può ricevere parti differenti da più dispositivi e, nello stesso tempo, condividere quelle già ottenute.

L'insieme dei peer che partecipano alla distribuzione dello stesso contenuto viene chiamato **sciame**, dall'inglese *swarm*.



Ogni parte ricevuta può essere verificata e poi condivisa con altri peer.

Figura 3.40: In uno sciame BitTorrent i peer possiedono e scambiano parti differenti dello stesso file.

Il funzionamento generale può essere riassunto così:

1. il file viene diviso in parti;
2. un'informazione descrive il contenuto e permette di cercare i peer che lo condividono;
3. il programma individua altri partecipanti allo sciame;
4. scarica parti diverse da più peer;
5. controlla che le parti ricevute siano corrette;
6. rende disponibili agli altri le parti già possedute;
7. ricompone il file completo quando tutte le parti sono state ottenute.

Un peer che possiede il file completo e continua a condividerlo viene spesso chiamato **seed**. Un contenuto con molti seed è normalmente più disponibile di un contenuto posseduto da pochi utenti.

Come vengono trovati gli altri peer

Per entrare nello sciame, il programma deve individuare altri partecipanti. Può utilizzare:

- un **tracker**, cioè un servizio che facilita l'incontro tra i peer;
- una **DHT**, cioè una tabella distribuita mantenuta dai nodi;
- informazioni contenute in un file `.torrent`;
- un **magnet link**, che identifica il contenuto senza dover scaricare prima un file descrittivo separato.

Il tracker non deve necessariamente possedere il file condiviso: può limitarsi a indicare quali peer stanno partecipando allo sciame.

Approfondimento: controllare l'integrità delle parti

Le parti del file vengono associate a impronte digitali calcolate mediante funzioni matematiche. Dopo la ricezione, il programma confronta l'impronta della parte ottenuta con quella prevista. Se non coincidono, la parte è considerata danneggiata o non valida e deve essere richiesta nuovamente.

Utilizzi e responsabilità

BitTorrent può essere utilizzato per distribuire contenuti di grandi dimensioni, per esempio:

- immagini di installazione di sistemi operativi liberi;
- software distribuito legalmente;
- archivi pubblici;
- grandi raccolte di dati;
- contenuti messi a disposizione direttamente dagli autori.

Il protocollo è uno strumento tecnico e non determina da solo se la condivisione sia lecita. Distribuire opere protette senza autorizzazione può violare il diritto d'autore. Prima di scaricare o condividere un contenuto è necessario verificare di avere il permesso per farlo.

Errore comune: il file non proviene dal sito che mostra il collegamento

Una pagina Web può pubblicare un file .torrent o un magnet link, ma i dati principali vengono poi scambiati tra i peer. Il sito, il tracker e i dispositivi che forniscono le parti svolgono ruoli differenti.

Esercizi: sezione 3.9

1. Che cosa si intende per modello client-server?
2. Quale ruolo svolge il client e quale ruolo svolge il server?
3. Perché client e server devono essere considerati ruoli e non soltanto tipi di computer?
4. Descrivere il ciclo richiesta-risposta con un esempio.
5. Individuare client, server, protocollo e risorsa quando si apre una pagina Web.
6. Individuare client e server quando si consulta il registro elettronico.
7. Perché un grande servizio utilizza normalmente più server?
8. Che cos'è un centro dati?
9. A che cosa serve un bilanciatore di carico?
10. Indicare tre vantaggi e tre limiti del modello client-server.
11. Spiegare perché un servizio centralizzato può essere fisicamente distribuito.
12. Che cosa significa la sigla P2P?
13. Quale doppio ruolo può svolgere un peer?

Esercizi: sezione 3.9 (continua)

14. Perché una rete P2P non garantisce automaticamente l'anonimato?
15. Indicare due vantaggi e due limiti di una rete peer-to-peer.
16. Come viene distribuito un file attraverso BitTorrent?
17. Che cos'è uno sciame?
18. Che cosa si intende per seed?
19. Quale funzione può svolgere un tracker?
20. Qual è la differenza tra un file .torrent e le parti del contenuto scaricate dai peer?
21. Perché BitTorrent può ridurre il carico su un unico server?
22. Correggere l'affermazione: "Usare BitTorrent rende anonimo l'indirizzo IP".
23. Fare un esempio di utilizzo legittimo di BitTorrent.
24. Confrontare client-server e peer-to-peer indicando organizzazione, vantaggi e limiti.
25. Progettare un semplice servizio per condividere i materiali di una classe e scegliere il modello più adatto.

Esercizi riepilogativi

Reti e Internet

Capitolo 3

1. Ricostruisci il viaggio di una fotografia

Numera da 1 a 7 le fasi, dalla partenza dallo smartphone di Bob fino all'arrivo ad Alice.

- ☐ Il server del servizio riceve e memorizza temporaneamente i dati.
- ☐ Lo smartphone di Alice riceve i pacchetti attraverso la rete mobile.
- ☐ Il router domestico inoltra i dati verso la rete dell'operatore.
- ☐ I dati vengono suddivisi in pacchetti.
- ☐ Il DNS permette di ottenere l'indirizzo IP associato al nome del servizio.
- ☐ I pacchetti attraversano più router di Internet.
- ☐ Lo smartphone di Bob trasmette i dati al punto di accesso Wi-Fi.

2. Scegli il mezzo o il dispositivo più adatto

Scrivi: **rame**, **fibra**, **wireless**, **hub**, **switch** oppure **router**.

1. Collegare due reti differenti:
2. Collegare stabilmente due edifici con elevata velocità e lunga distanza:
3. Inoltrare i dati soltanto verso la porta corretta in una LAN:
4. Collegare uno smartphone senza cavo:
5. Ripetere il segnale ricevuto verso tutte le porte:
6. Realizzare un breve collegamento Ethernet economico:

3. Classifica reti e topologie

Situazione	Tipo o topologia
Auricolari collegati allo smartphone	
Rete interna di un laboratorio scolastico	
Collegamento tra sedi in città diverse	
Tutti i dispositivi dipendono da un apparato centrale	
Ogni nodo dispone di più percorsi alternativi	

4. Diagnosi rapida

Il browser raggiunge un sito inserendo l'indirizzo IP, ma non usando il nome di dominio. Quale servizio potrebbe non funzionare? Spiega perché HTTPS non risolverebbe questo problema.

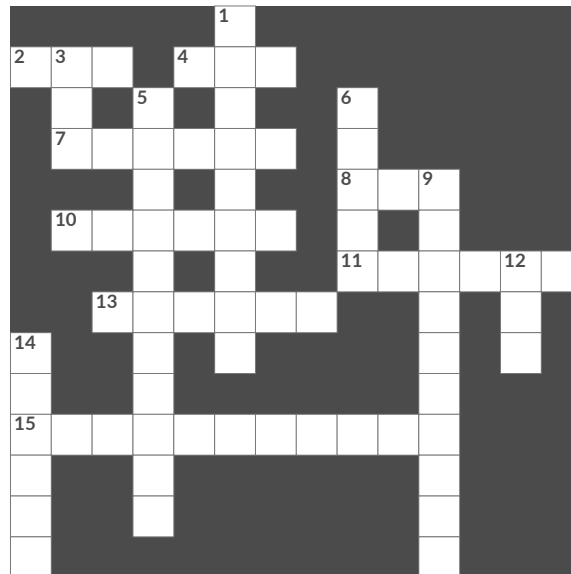
Esercizi riepilogativi

Cruciverba delle reti

Capitolo 3

Completa lo schema

Inserisci le parole senza spazi. Le sigle vanno scritte in lettere maiuscole.



Orizzontali

2. Protocollo con pochi controlli e minore attesa. (3)
4. Rete geografica. (3)
7. Inoltra i dati alla porta corretta in una LAN. (6)
8. Protocollo con consegna controllata e ordinata. (3)
10. Fornisce servizi o risorse. (6)
11. Topologia con un nodo centrale. (6)
13. Collega reti differenti. (6)
15. Identifica un'interfaccia o un dispositivo in rete. (11)

Verticali

1. Unità in cui vengono suddivisi i dati trasmessi. (9)
3. Traduce i nomi di dominio in indirizzi IP. (3)
5. Mezzo che usa impulsi luminosi. (11)
6. Protocollo Web con comunicazione protetta. (5)
9. Modello distribuito in cui i nodi possono offrire e richiedere risorse. (10)
12. Rete locale. (3)
14. Richiede un servizio a un server. (6)

Progetta una piccola rete

Un laboratorio ha 12 computer, una stampante di rete, un server e l'accesso a Internet. Indica gli apparati necessari, la topologia più adatta e il tipo di rete per estensione. Motiva in poche righe.

Parte IV

Algoritmi e programmazione

4. Algoritmi e programmazione

Un computer esegue con grande rapidità le istruzioni ricevute, ma non è in grado di inventare autonomamente la soluzione di un problema. Prima di scrivere un programma è necessario stabilire quali dati utilizzare, quali operazioni eseguire e in quale ordine. La programmazione comincia quindi dalla progettazione di un algoritmo chiaro e verificabile.

In questo capitolo i problemi verranno rappresentati principalmente attraverso i **diagrammi di flusso**. Saranno utilizzati pochi simboli, sempre con lo stesso significato: cerchi per inizio e fine, rettangoli per le operazioni, parallelogrammi per l'acquisizione e la comunicazione dei dati, riconoscibili dalle etichette IN e OUT collocate sopra l'angolo superiore destro del simbolo, e rombi per le condizioni, dai quali partono i rami V e F. Nella prima parte tutte le variabili saranno numeriche e non sarà necessario dichiararne il tipo, così da concentrare l'attenzione sulla logica della soluzione. Le frecce di ritorno permetteranno inoltre di rappresentare le strutture iterative e di osservare come uno stesso gruppo di istruzioni possa essere ripetuto fino al verificarsi di una condizione oppure per un numero stabilito di volte.

Conoscenze	Abilità	Competenze
• algoritmo e programma; • diagrammi di flusso; • variabili numeriche, input e output; • operatori aritmetici e di confronto; • condizioni semplici e composte; • operatore modulo; • iterazioni, contatori, accumulatori e sentinelle; • schemi di ripetizione nei diagrammi di flusso; • numeri pseudocasuali, seed e generazione di valori in un intervallo.	• individuare input, elaborazione e output; • interpretare e costruire diagrammi di flusso; • usare variabili numeriche e assegnamenti; • formulare condizioni con confronti e operatori logici; • progettare e tracciare iterazioni nei diagrammi di flusso; • usare contatori, accumulatori e sentinelle; • generare valori pseudocasuali in un intervallo; • combinare numeri pseudocasuali e decisioni.	• analizzare un problema prima di programmare; • progettare soluzioni ordinate e non ambigue; • verificare gli algoritmi con casi di prova; • riconoscere e correggere errori logici; • organizzare una ripetizione e verificarne la terminazione; • applicare la programmazione a semplici situazioni reali.

4.1 Dal problema all'algoritmo

Programmare non significa iniziare immediatamente a scrivere istruzioni in un linguaggio. Il primo passo consiste nel comprendere il problema e nel progettare una sequenza di azioni capace di produrre il risultato richiesto.

Per esempio, se si deve calcolare l'area di un rettangolo, è necessario sapere quali dati servono, quale formula applicare e quale risultato comunicare. Solo dopo aver chiarito questi elementi sarà possibile rappresentare la soluzione con un diagramma di flusso e, successivamente, tradurla in un programma.

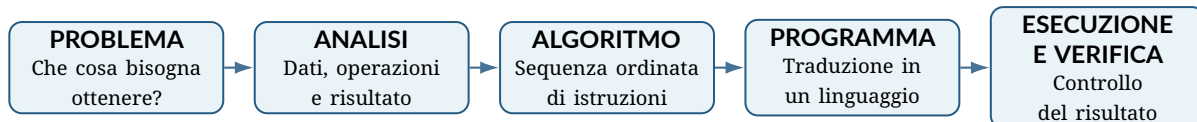


Figura 4.1: Dalla comprensione del problema all'esecuzione del programma. La scrittura del codice è soltanto una delle fasi del lavoro.

4.1.1 Che cos'è un algoritmo

Il termine **algoritmo** indica una procedura formata da istruzioni ordinate. Gli algoritmi non appartengono esclusivamente all'informatica: una ricetta, le indicazioni per raggiungere un luogo e il procedimento per svolgere un calcolo sono esempi di sequenze operative.

Definizione: algoritmo

Un **algoritmo** è una sequenza ordinata e finita di istruzioni non ambigue che, a partire da determinati dati, permette di ottenere un risultato o di risolvere un problema.

Un semplice algoritmo espresso in linguaggio naturale potrebbe essere:

1. acquisire la base del rettangolo;
2. acquisire l'altezza del rettangolo;
3. moltiplicare la base per l'altezza;
4. comunicare il valore dell'area.

L'ordine è essenziale. Non sarebbe possibile comunicare l'area prima di aver ricevuto i dati e svolto il calcolo.

Curiosità: origine del termine

La parola *algoritmo* deriva dalla latinizzazione del nome del matematico persiano al-Khwarizmi, vissuto tra l'VIII e il IX secolo. Le sue opere contribuirono alla diffusione di metodi sistematici di calcolo e dell'algebra.

4.1.2 Le caratteristiche di un buon algoritmo

Un algoritmo deve possedere alcune caratteristiche fondamentali.

Ordine

Le istruzioni devono essere disposte nella sequenza corretta. Spostare un'operazione può modificare il risultato o rendere impossibile l'esecuzione.

Chiarezza

Ogni istruzione deve avere un significato preciso. Espressioni come “fare il calcolo necessario” oppure “scegliere un valore adatto” sono troppo vaghe per essere eseguite da un computer.

Finitudine

La procedura deve terminare dopo un numero finito di passaggi. Un algoritmo che continua per sempre senza produrre il risultato non risolve il problema assegnato.

Eseguibilità

Le operazioni devono essere concretamente realizzabili con i dati e gli strumenti disponibili.

Generalità

Quando possibile, la soluzione deve funzionare per tutti i dati che rispettano le condizioni del problema, non soltanto per un singolo esempio.

Esempio: procedura troppo specifica

L'istruzione “calcolare 5×8 ” produce correttamente l'area di un rettangolo con base 5 e altezza 8, ma non risolve il problema in generale. Una soluzione più generale è “moltiplicare il valore della base per il valore dell'altezza”.

Errore comune: confondere dettaglio e precisione

Un algoritmo preciso non deve necessariamente essere molto lungo. Deve contenere tutte le informazioni necessarie, evitando sia i passaggi mancanti sia i dettagli che non contribuiscono alla soluzione.

4.1.3 Algoritmo e programma

Algoritmo e programma sono concetti collegati, ma non coincidono.

- L'**algoritmo** descrive la soluzione logica del problema.
- Il **programma** traduce tale soluzione in un linguaggio che può essere elaborato da un computer o da un microcontrollore.

Lo stesso algoritmo può essere tradotto in linguaggi differenti. Il calcolo dell'area di un rettangolo può essere realizzato in C++, Python, JavaScript oppure nel linguaggio usato

da Arduino. La formula e l'ordine logico delle operazioni rimangono gli stessi, mentre cambia la sintassi.

Aspetto	Algoritmo	Programma
Scopo	Descrivere la soluzione	Far eseguire la soluzione a un dispositivo
Forma	Linguaggio naturale, pseudo-codice o diagramma	Codice scritto in un linguaggio di programmazione
Dipendenza dal linguaggio	Generalmente indipendente	Dipende dalla sintassi scelta
Verifica	Simulazione dei passaggi	Esecuzione e test sul dispositivo

Tabella 4.1: Confronto tra algoritmo e programma.

4.1.4 Input, elaborazione e output

Per analizzare un problema è utile dividerlo in tre parti: **input**, **elaborazione** e **output**.

Definizione: input

Gli **input** sono i dati che devono essere forniti all'algoritmo. Possono provenire da una persona, da un file, da un sensore o da un altro sistema.

Definizione: elaborazione

L'**elaborazione** comprende i calcoli, i confronti e le operazioni svolti sui dati di ingresso.

Definizione: output

Gli **output** sono i risultati prodotti dall'algoritmo e comunicati all'esterno, per esempio attraverso uno schermo, un file, un messaggio o un dispositivo elettronico.

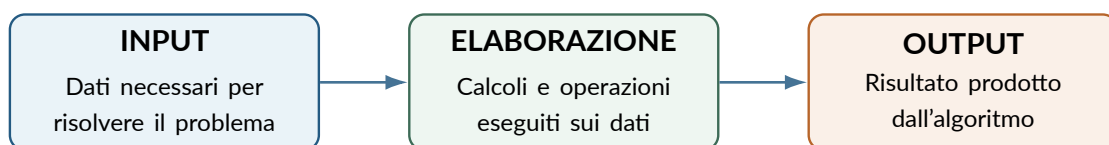


Figura 4.2: Il modello input-elaborazione-output permette di individuare gli elementi principali di molti problemi.

Nel problema dell'area del rettangolo:

- gli input sono la base e l'altezza;
- l'elaborazione è la moltiplicazione $\text{base} \times \text{altezza}$;
- l'output è l'area calcolata.

Esempio: costo totale di un acquisto

Si vuole calcolare il costo totale conoscendo la quantità acquistata e il prezzo di un singolo prodotto.

- **Input:** quantità e prezzo unitario.
- **Elaborazione:** $\text{totale} = \text{quantità} \times \text{prezzo unitario}$.
- **Output:** costo totale.

Esercizi: comprendere algoritmi e problemi

1. Scrivere con parole proprie la differenza tra algoritmo e programma.
2. Indicare quali tra le seguenti procedure possono essere considerate algoritmi e motivare la risposta:
 - a) sommare due numeri e comunicare il risultato;
 - b) scegliere una risposta bella;
 - c) leggere tre valori, calcolarne la media e visualizzarla;
 - d) continuare a eseguire calcoli senza mai terminare.
3. Riordinare le seguenti istruzioni per calcolare il perimetro di un rettangolo:
 - a) comunicare il perimetro;
 - b) acquisire l'altezza;
 - c) calcolare $2 \times (\text{base} + \text{altezza})$;
 - d) acquisire la base.
4. Individuare input, elaborazione e output nel calcolo della media di tre voti.
5. Individuare input, elaborazione e output nel calcolo del costo di una recinzione, conoscendo la lunghezza totale e il costo per metro.
6. Spiegare perché l'istruzione "calcolare il risultato corretto" non è adatta a un algoritmo.
7. Trasformare in una sequenza ordinata di istruzioni il problema: calcolare il volume di un parallelepipedo conoscendo lunghezza, larghezza e altezza.
8. Un algoritmo riceve in input un numero e produce in output il suo quadrato. Indicare l'elaborazione necessaria.
9. Proporre due diversi insiemi di dati con cui verificare l'algoritmo dell'area del rettangolo.
10. Descrivere un semplice algoritmo della vita quotidiana formato da almeno cinque passaggi, facendo attenzione a non utilizzare istruzioni ambigue.

4.2 Variabili, assegnamento, input e output

Dopo aver individuato i dati necessari e il risultato da ottenere, bisogna stabilire dove conservare i valori durante l'esecuzione dell'algoritmo. A questo scopo si utilizzano le **variabili**. Prima di rappresentare le istruzioni con i simboli dei diagrammi di flusso è quindi necessario comprendere che cosa sono le variabili e come funzionano le operazioni di input, assegnamento e output.

4.2.1 La variabile come contenitore

Una variabile può essere immaginata come un **contenitore dotato di un nome**. Nel contenitore viene memorizzato un valore che può essere letto, utilizzato nei calcoli oppure sostituito durante l'esecuzione.

Definizione: variabile

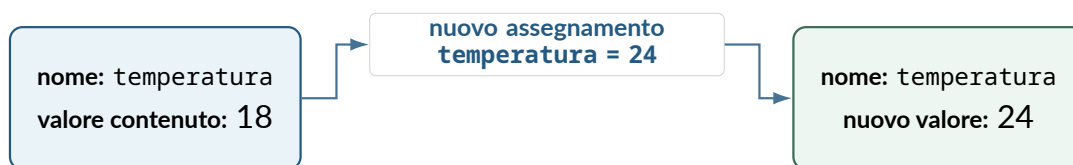
Una **variabile** è uno spazio della memoria identificato da un nome e utilizzato per conservare un valore. Il valore può cambiare mentre l'algoritmo viene eseguito; il nome permette di ritrovarlo e di utilizzarlo nelle istruzioni successive.

Ogni variabile presenta quindi due elementi fondamentali:

- un **nome**, che identifica il contenitore, per esempio base, altezza o temperatura;
- un **valore**, cioè il dato che in quel momento è conservato nel contenitore.

Le variabili servono a:

- memorizzare i dati ricevuti in input;
- conservare i risultati intermedi prodotti dai calcoli;
- conservare il risultato finale prima di comunicarlo in output;
- aggiornare un dato quando il suo valore cambia.



Il nome rimane invariato; il nuovo valore sostituisce quello precedente.

Figura 4.3: Una variabile può essere immaginata come un contenitore con un nome e un valore modificabile.

4.2.2 Scegliere il nome delle variabili

Il nome dovrebbe indicare con chiarezza ciò che la variabile contiene. Nomi come base, prezzo e temperatura sono più comprensibili di nomi generici come x1 o dato2.

In questo manuale si adotteranno alcune semplici regole:

- il nome inizia con una lettera;
- non contiene spazi;
- non contiene accenti o caratteri speciali;
- quando è composto da più parole, si può usare una lettera maiuscola per separarle, per esempio prezzoFinale;
- il nome deve essere coerente con il valore conservato.

Nei problemi matematici molto brevi potranno essere usati anche nomi come *a*, *b* e *c*. Nei problemi applicativi sono invece preferibili nomi descrittivi.

Approfondimento: in questa prima parte si usano soltanto numeri

Nei primi diagrammi tutte le variabili conterranno valori numerici e non sarà necessario dichiararne il tipo. Questa semplificazione permette di concentrarsi sulla logica dell'algoritmo. La dichiarazione formale delle variabili verrà introdotta nella parte dedicata ad Arduino.

4.2.3 L'assegnamento

L'**assegnamento** è l'operazione con cui un valore viene memorizzato in una variabile. Si utilizzerà la scrittura:

`variabile = valore oppure espressione`

Per esempio:

`temperatura = 18`

memorizza il valore 18 nella variabile *temperatura*. Se successivamente viene eseguita l'istruzione

`temperatura = 24`

il valore 24 sostituisce il valore precedente. Il nome del contenitore rimane invariato, mentre cambia ciò che contiene.

Quando a destra del simbolo `=` compare un'espressione, l'espressione viene prima calcolata e il risultato viene poi conservato nella variabile posta a sinistra:

`area = base * altezza`

Se *base* contiene 5 e *altezza* contiene 3, l'espressione produce 15 e il valore 15 viene memorizzato in *area*.

Errore comune: l'assegnamento non è un'uguaglianza matematica

L'istruzione `x = x + 1` significa: leggere il valore attuale di *x*, aggiungere 1 e memorizzare il risultato nuovamente in *x*. Se *x* contiene 7, dopo l'istruzione conterrà 8.

Errore comune: una variabile conserva un solo valore alla volta

Quando a una variabile viene assegnato un nuovo valore, quello precedente viene sostituito, a meno che non sia stato prima copiato in un'altra variabile.

4.2.4 Acquisire un dato: l'input

Un'istruzione di **input** acquisisce un valore dall'esterno e lo memorizza in una variabile. In forma testuale verrà scritta così:

```
IN base
```

L'istruzione non significa semplicemente "leggere la parola base": significa ricevere un valore e conservarlo nella variabile chiamata base. Il dato può provenire da una persona, da un sensore, da un file oppure da un altro dispositivo.

Una variabile deve ricevere un valore prima di essere utilizzata in un calcolo. Per esempio, prima di eseguire

```
area = base * altezza
```

è necessario che base e altezza abbiano già ricevuto un valore.

Esempio: input per l'area del rettangolo

Le istruzioni

```
IN base    IN altezza
```

acquisiscono due valori e li conservano nei rispettivi contenitori. Se vengono forniti 5 e 3, dopo gli input base contiene 5 e altezza contiene 3.

4.2.5 Comunicare un risultato: l'output

Un'istruzione di **output** comunica all'esterno un valore già disponibile. In forma testuale verrà scritta così:

```
OUT area
```

L'istruzione comunica il valore contenuto nella variabile area. L'output può essere visualizzato sullo schermo, salvato in un file, inviato a un altro sistema oppure utilizzato per comandare un dispositivo.

Un output può anche contenere una parola o una frase. Il testo che deve essere mostrato esattamente com'è scritto viene racchiuso tra virgolette:

```
OUT "Calcolo completato"
```

È inoltre possibile affiancare, cioè **concatenare**, una frase e il valore di una variabile. In questo manuale la concatenazione verrà indicata con il simbolo +:

OUT "Ecco il risultato: " + area

Le virgolette delimitano il testo letterale, mentre area viene sostituita dal valore contenuto nella variabile. Per esempio, se area contiene 15, l'output visualizzato sarà Ecco il risultato: 15. In questo contesto il simbolo + non indica una somma aritmetica, ma l'unione di più elementi nell'output.

Input e output hanno quindi funzioni opposte:

- con IN un valore entra nell'algoritmo e viene memorizzato in una variabile;
- con OUT un valore già calcolato viene comunicato all'esterno.

4.2.6 Seguire i valori passo dopo passo

Per controllare un algoritmo è utile annotare come cambia il contenuto delle variabili dopo ogni istruzione. Si consideri la sequenza:

IN base IN altezza area = base * altezza OUT area

Supponendo che gli input siano 5 e 3, si ottiene la seguente tabella di esecuzione.

Istruzione eseguita	base	altezza	area	Output
IN base	5	—	—	—
IN altezza	5	3	—	—
area = base * altezza	5	3	15	—
OUT area	5	3	15	15

Tabella 4.2: Esecuzione passo dopo passo del calcolo dell'area.

La tabella permette di verificare che nessuna variabile venga utilizzata prima di aver ricevuto un valore e che l'output sia prodotto soltanto dopo il calcolo.

Esercizi: variabili, assegnamento, input e output

1. Spiegare con parole proprie perché una variabile può essere paragonata a un contenitore.
2. Indicare il nome e il valore nelle seguenti variabili: altezza = 12, prezzo = 4.5, temperatura = 21.
3. Scegliere nomi significativi per conservare: la quantità acquistata, il prezzo unitario, il costo totale e l'umidità del terreno.
4. Dopo le istruzioni $x = 4$, $y = x + 3$, $x = 10$, indicare il valore finale contenuto in x e in y.
5. Simulare le istruzioni $a = 6$, $b = 2$, $somma = a + b$, $a = 10$. Indicare i valori finali di tutte le variabili.

Esercizi: variabili, assegnamento, input e output (continua)

6. Scrivere le istruzioni testuali necessarie per acquisire due numeri, calcolarne il prodotto e comunicarlo.
7. Individuare l'errore nella sequenza: `totale = quantita * prezzo`, IN `quantita`, IN `prezzo`, OUT `totale`.
8. Completare una tabella di esecuzione per: IN `lato`, `perimetro = lato * 4`, OUT `perimetro`, usando come input 7.
9. Scrivere le variabili e gli assegnamenti necessari per calcolare il costo totale di una recinzione conoscendo la lunghezza e il costo per metro.
10. Spiegare la differenza tra IN `temperatura`, `temperatura = 25` e OUT `temperatura`.

4.3 I simboli dei diagrammi di flusso

4.3.1 Dalla descrizione al diagramma di flusso

Il linguaggio naturale è utile per spiegare una soluzione, ma può contenere formulazioni ambigue. Per rappresentare con maggiore precisione l'ordine delle operazioni si possono utilizzare i **diagrammi di flusso**, chiamati anche *flow chart*.

Un diagramma di flusso utilizza simboli geometrici collegati da frecce. La forma di ogni simbolo indica il tipo di istruzione, mentre le frecce mostrano il percorso dell'esecuzione.

Nei primi esercizi del capitolo verrà adottata una convenzione semplificata:

- tutte le variabili conterranno valori numerici;
- le variabili saranno utilizzate direttamente, senza dichiararne il tipo;
- le operazioni saranno scritte in una forma vicina alla programmazione;
- la dichiarazione formale delle variabili verrà introdotta soltanto nella parte dedicata ad Arduino.

Questa scelta permette di concentrarsi sul ragionamento e sulla corretta sequenza delle istruzioni.

Approfondimento: diagramma di flusso e pseudocodice

Un algoritmo può essere rappresentato anche mediante lo **pseudocodice**, cioè un testo strutturato simile a un linguaggio di programmazione ma non legato a una sintassi precisa. In questo manuale si partirà dai diagrammi di flusso perché rendono visibile il percorso dell'esecuzione; lo pseudocodice potrà essere introdotto successivamente come forma più compatta.

Dopo aver introdotto variabili, assegnamento, input e output, gli stessi concetti possono essere rappresentati graficamente. Un diagramma di flusso è leggibile soltanto se ogni simbolo viene utilizzato in modo coerente. In questo manuale verranno adottati cinque

elementi fondamentali: cerchio, rettangolo, parallelogramma di input, parallelogramma di output e rombo. Le frecce collegano i simboli e determinano l'ordine di esecuzione.

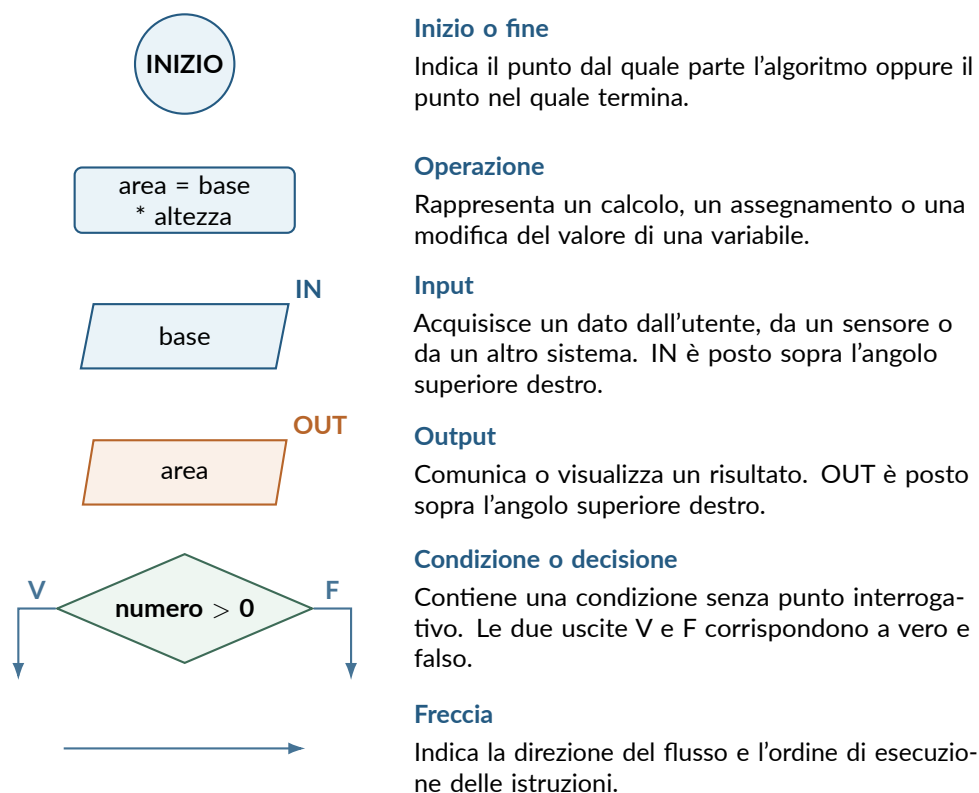


Figura 4.4: Simboli adottati nel manuale per costruire i diagrammi di flusso.

4.3.2 Il cerchio: inizio e fine

Il cerchio indica il punto iniziale o finale dell'algoritmo. Al suo interno viene scritta una delle due parole:

INIZIO oppure FINE.

Ogni diagramma deve avere un punto di inizio chiaramente riconoscibile. Tutti i percorsi devono inoltre condurre a una conclusione, salvo i casi in cui un programma sia progettato intenzionalmente per ripetere continuamente alcune istruzioni, come accade spesso nei sistemi Arduino.

Definizione: simbolo di inizio e fine

Il cerchio con la scritta INIZIO rappresenta la prima posizione del flusso. Il cerchio con la scritta FINE rappresenta la conclusione dell'algoritmo.

4.3.3 Il rettangolo: operazioni e assegnamenti

Il rettangolo contiene un'operazione da eseguire. Può rappresentare:

- un calcolo aritmetico;

- l'assegnamento di un valore a una variabile;
- la modifica di un valore già presente;
- un'altra istruzione che non richiede una scelta.

Esempi:

somma = a + b

area = base * altezza

prezzoFinale = prezzo - sconto

Il simbolo = indica l'**assegnamento**: il valore ottenuto dall'espressione a destra viene memorizzato nella variabile indicata a sinistra, come spiegato nella [sezione 4.2](#).

Errore comune: scrivere una domanda nel rettangolo

Il rettangolo descrive un'operazione. Una condizione come temperatura > 30 deve essere inserita in un rombo, perché può produrre due percorsi differenti.

4.3.4 Il parallelogramma IN: acquisire un dato

Il parallelogramma di input indica che un dato deve entrare nell'algoritmo. La variabile che riceverà il valore viene scritta all'interno del simbolo; l'etichetta IN viene collocata appena sopra il suo angolo superiore destro.

Per esempio:

IN base

significa che l'algoritmo acquisisce un valore e lo associa alla variabile base.

L'input può provenire da fonti differenti:

- una persona che digita un valore;
- un sensore che misura una grandezza;
- un file contenente dati;
- un altro programma o dispositivo.

Nel diagramma non è necessario specificare subito come il dato verrà acquisito tecnicamente. È sufficiente indicare che quel valore deve essere disponibile prima delle operazioni che lo utilizzano.

4.3.5 Il parallelogramma OUT: comunicare un risultato

Il parallelogramma di output rappresenta la comunicazione di un risultato. Il valore o la variabile da mostrare viene scritto all'interno del simbolo; l'etichetta OUT viene collocata appena sopra il suo angolo superiore destro.

Per esempio:

OUT area

significa che il valore contenuto nella variabile area deve essere reso disponibile all'esterno dell'algoritmo.

Un output può essere:

- un numero visualizzato sullo schermo;
- una parola o una frase racchiusa tra virgolette;
- una frase concatenata a una variabile, per esempio "Ecco il risultato: " + a;
- un valore registrato in un file;
- un comando inviato a un dispositivo, come l'accensione di un LED.

Nel parallelogramma si può quindi scrivere a, per mostrare soltanto il valore, oppure "Ecco il risultato: " + a, per produrre un messaggio più comprensibile. Le virgolette delimitano la parte testuale e non vengono visualizzate nel risultato finale.

Definizione: distinzione tra IN e OUT

Input e output utilizzano la stessa forma geometrica, ma vengono distinti dall'etichetta collocata all'esterno, sopra l'angolo superiore destro: IN per i dati acquisiti, OUT per i risultati comunicati.

4.3.6 Il rombo: condizioni e decisioni

Il rombo contiene una condizione che può risultare vera oppure falsa. La condizione viene scritta senza punto interrogativo. Dai due lati del rombo partono i rami:

- **V**, quando la condizione è vera;
- **F**, quando la condizione è falsa.

Esempi di condizioni:

numero > 0

prezzo >= 100

temperatura == 25

Il rombo permette quindi di modificare il percorso dell'algoritmo in base ai dati. Gli operatori di confronto e la costruzione delle decisioni saranno approfonditi nelle sezioni successive.

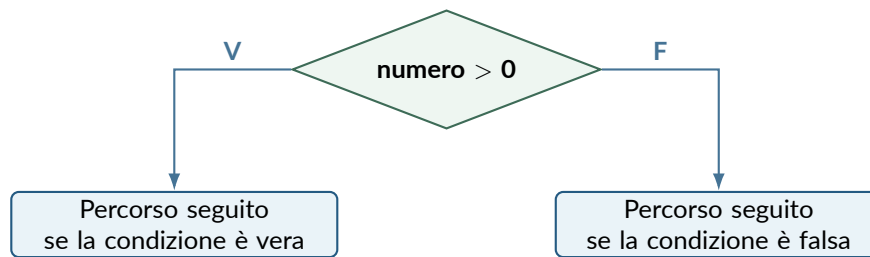


Figura 4.5: Dal rombo partono due rami: uno viene seguito quando la condizione è vera, l'altro quando è falsa.

Errore comune: dimenticare un ramo

Una condizione deve prevedere entrambe le possibilità. Anche quando in uno dei due casi non è necessario svolgere alcuna operazione, il percorso deve essere rappresentato chiaramente e deve ricongiungersi al flusso oppure raggiungere la fine.

4.3.7 Le frecce: la direzione dell'esecuzione

Le frecce collegano i simboli e indicano l'ordine con cui vengono eseguite le istruzioni. In assenza di decisioni, il diagramma procede normalmente dall'alto verso il basso.

Una freccia deve:

- partire da un simbolo e raggiungere quello successivo;
- terminare con una punta chiaramente visibile;
- evitare di attraversare testi e simboli;
- rendere immediatamente comprensibile il percorso.

Quando il flusso contiene un rombo, le etichette V e F devono essere collocate vicino ai rispettivi rami senza sovrapporsi alle linee o ai simboli.

4.3.8 Un primo diagramma completo

La figura seguente anticipa un semplice algoritmo: acquisisce un numero, ne calcola il doppio e comunica il risultato.

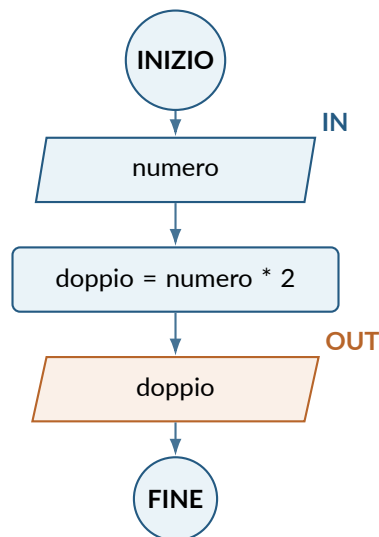


Figura 4.6: Primo esempio di diagramma di flusso sequenziale.

Il diagramma viene letto seguendo le frecce:

1. l'esecuzione parte dal cerchio INIZIO;
2. il parallelogramma IN acquisisce il valore di numero;
3. il rettangolo calcola il doppio e lo memorizza in doppio;
4. il parallelogramma OUT comunica il risultato;
5. il cerchio FINE conclude l'algoritmo.

Non è necessario dichiarare la variabile numero né specificare che contiene un valore numerico: in questa prima parte del manuale tale informazione è assunta come convenzione.

4.3.9 Regole grafiche per diagrammi leggibili

Un diagramma deve essere corretto anche dal punto di vista grafico. Una rappresentazione disordinata rende difficile interpretare persino un algoritmo semplice.

1. Collocare l'INIZIO nella parte superiore del diagramma.
2. Disporre il flusso principale dall'alto verso il basso.
3. Utilizzare sempre la stessa forma per lo stesso tipo di istruzione.
4. Scrivere testi brevi e leggibili all'interno dei simboli.
5. Evitare l'incrocio delle frecce quando è possibile.
6. Etichettare entrambi i rami dei rombi.
7. Verificare che ogni simbolo sia raggiungibile.
8. Controllare che ogni percorso termini o prosegua verso un'istruzione definita.

Approfondimento: convenzioni grafiche differenti

Esistono standard e programmi che utilizzano simboli leggermente differenti. In molti diagrammi, per esempio, inizio e fine sono rappresentati da ovali anziché da cerchi. Nel manuale si adotterà una convenzione unica e semplificata, così che la forma dei simboli rimanga costante in tutti gli esercizi.

Simbolo	Scritta tipica	Funzione
Cerchio	INIZIO oppure FINE	Avvia o conclude il flusso
Rettangolo	area = base * altezza	Esegue un'operazione
Parallelogramma IN	IN base	Acquisisce un dato
Parallelogramma OUT	OUT area	Comunica un risultato
Rombo	numero > 0	Sceglie tra i rami V e F
Freccia	—	Indica l'ordine di esecuzione

Tabella 4.3: Riepilogo dei simboli usati nei diagrammi di flusso.

Esercizi: riconoscere e usare i simboli

1. Associare a ciascuna istruzione il simbolo corretto:
 - a) IN altezza;
 - b) somma = a + b;
 - c) OUT somma;
 - d) a > b;
 - e) FINE.
2. Spiegare perché input e output usano la stessa forma ma devono riportare scritte differenti.
3. Individuare l'errore: un diagramma utilizza un rettangolo per la condizione voto >= 6.
4. Individuare l'errore: dal rombo temperatura > 30 parte soltanto il ramo V.
5. Disegnare la sequenza di simboli necessaria per acquisire due numeri, sommarli e comunicare il risultato.
6. Disegnare un diagramma che acquisisca la base e l'altezza di un rettangolo, calcoli l'area e la comunichi.
7. Disegnare un diagramma che acquisisca un numero, ne calcoli il quadrato e mostri il risultato.
8. Completare il diagramma della [figura 4.6](#) sostituendo il calcolo del doppio con il calcolo del triplo.

Esercizi: riconoscere e usare i simboli (continua)

9. Proporre una condizione adatta a essere inserita in un rombo e indicare i significati dei rami V e F.
10. Riscrivere in forma più chiara l'istruzione troppo lunga: “prendere il primo valore e poi prendere il secondo valore e fare il calcolo della loro somma e mostrare il risultato”.
11. Controllare il diagramma realizzato nell'esercizio 6 usando i valori base 5 e altezza 3. Scrivere il risultato atteso.
12. Elencare almeno quattro regole grafiche che rendono leggibile un diagramma di flusso.

4.4 Gli algoritmi sequenziali

Un **algoritmo sequenziale** è formato da istruzioni eseguite una dopo l'altra, nell'ordine indicato dalle frecce. Non sono presenti domande né percorsi alternativi: ogni volta che l'algoritmo viene avviato, tutte le istruzioni vengono attraversate nello stesso ordine.

Questa struttura è adatta ai problemi nei quali è sufficiente:

1. acquisire uno o più dati;
2. eseguire calcoli;
3. comunicare uno o più risultati.

Definizione: sequenza

Una **sequenza** è una struttura algoritmica nella quale ogni istruzione viene eseguita una sola volta, procedendo dalla prima all'ultima senza effettuare scelte.

4.4.1 Gli operatori aritmetici

Nei rettangoli dei diagrammi verranno utilizzati gli operatori aritmetici riportati nella tabella seguente.

Operatore	Significato	Esempio
+	Addizione	somma = a + b
-	Sottrazione	differenza = a - b
*	Moltiplicazione	area = base * altezza
/	Divisione	media = somma / 3

Tabella 4.4: Operatori aritmetici utilizzati nei primi algoritmi.

Il simbolo * viene usato al posto del segno di moltiplicazione, mentre il simbolo / indica

la divisione. Le parentesi permettono di stabilire con chiarezza l'ordine dei calcoli.

Esempio: l'importanza delle parentesi

L'istruzione

$$\text{media} = (a + b + c) / 3$$

somma prima i tre valori e divide poi il risultato per 3. Senza parentesi, la divisione verrebbe applicata soltanto a c prima delle addizioni.

Errore comune: confondere il valore con il nome della variabile

Nel rettangolo si scrive $\text{area} = \text{base} * \text{altezza}$, non $\text{area} = 5 * 3$, se si vuole ottenere un algoritmo generale. I valori 5 e 3 possono essere usati per verificare l'algoritmo, ma non devono sostituire le variabili nel progetto della soluzione.

4.4.2 Esempio guidato: sommare due numeri

Il primo problema consiste nell'acquisire due numeri, calcolarne la somma e comunicare il risultato. Quando più dati vengono acquisiti nello stesso passaggio, possono essere indicati nello stesso parallelogramma IN, separandone i nomi con una virgola.

Input	I valori delle variabili a e b.
Elaborazione	$\text{somma} = a + b$.
Output	Il valore contenuto in somma.

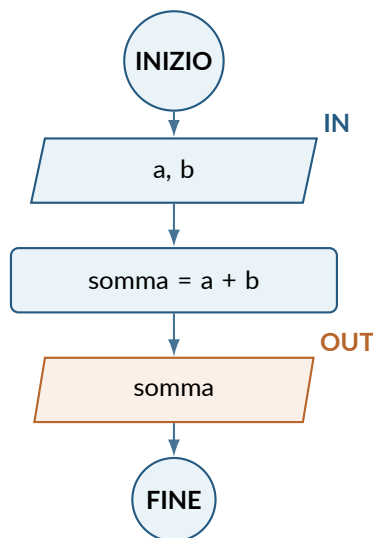


Figura 4.7: Diagramma di flusso per la somma di due numeri.

Il diagramma deve essere letto dall'alto verso il basso:

1. vengono acquisiti i valori di a e b;
2. la loro somma viene calcolata e memorizzata in somma;

3. il valore di somma viene comunicato in output.

Supponendo che gli input siano 7 e 5, la tabella di esecuzione diventa:

Istruzione	a	b	somma	Output
IN a	7	—	—	—
IN b	7	5	—	—
somma = a + b	7	5	12	—
OUT somma	7	5	12	12

Tabella 4.5: Esecuzione dell'algoritmo della somma con gli input 7 e 5.

4.4.3 Esempio guidato: area del rettangolo

Per calcolare l'area di un rettangolo servono due dati: la base e l'altezza. La formula è:

$$\text{area} = \text{base} \cdot \text{altezza}$$

Nel rettangolo del diagramma la moltiplicazione viene scritta con l'operatore *.

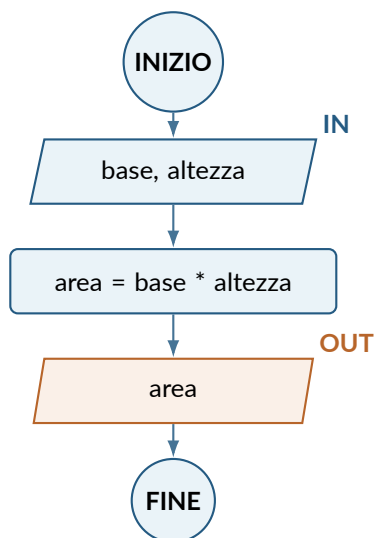


Figura 4.8: Diagramma di flusso per il calcolo dell'area di un rettangolo.

Esempio: verifica con un caso di prova

Se base = 8 e altezza = 5, il rettangolo di elaborazione calcola:

$$\text{area} = 8 * 5 = 40$$

L'output atteso è quindi 40.

4.4.4 Produrre più risultati

Un algoritmo può produrre più output. Per esempio, conoscendo base e altezza di un rettangolo, è possibile calcolare sia l'area sia il perimetro:

$$\text{area} = \text{base} * \text{altezza}$$

$$\text{perimetro} = 2 * (\text{base} + \text{altezza})$$

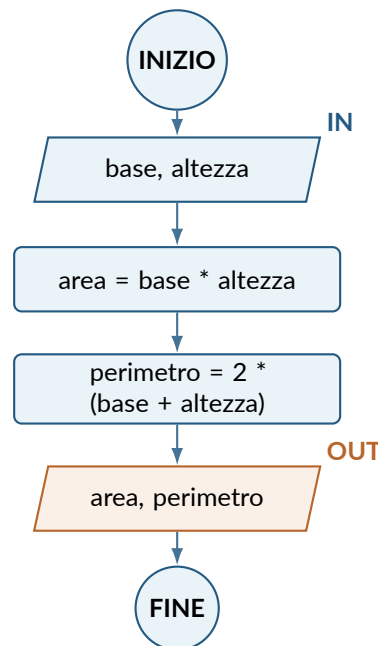


Figura 4.9: Un algoritmo sequenziale può eseguire più calcoli e produrre più output.

I due risultati devono essere calcolati prima di essere comunicati. Quando vengono prodotti nello stesso momento, possono essere indicati nello stesso parallelogramma OUT, separandone i nomi con una virgola. L'ordine dei due rettangoli può essere invertito senza modificare il risultato, perché il calcolo dell'area non dipende dal perimetro e il calcolo del perimetro non dipende dall'area.

Approfondimento: istruzioni indipendenti e dipendenti

Due istruzioni sono **indipendenti** se possono essere scambiate senza modificare il risultato dell'algoritmo. Bisogna considerare sia le variabili lette sia quelle modificate. Per esempio, $x = 1$ e $x = 2$ non leggono il risultato reciproco, ma il loro ordine cambia il valore finale di x : non sono indipendenti. Sono invece **dipendenti** le istruzioni per le quali l'ordine deve essere rispettato. Per esempio, $\text{prezzoFinale} = \text{prezzo} - \text{sconto}$ dipende dal precedente calcolo di sconto.

4.4.5 Esempio guidato: media di tre numeri

Per calcolare la media di tre numeri si può procedere in due modi.

Metodo diretto

$$\text{media} = (a + b + c) / 3$$

Metodo con un risultato intermedio

$$\text{somma} = a + b + c$$

$$\text{media} = \text{somma} / 3$$

Il secondo metodo richiede una variabile in più, ma rende visibili i due passaggi logici e può risultare più semplice da controllare.

Esempio: tabella di esecuzione della media

Con gli input 6, 8 e 10:

$$\text{somma} = 6 + 8 + 10 = 24$$

$$\text{media} = 24 / 3 = 8$$

4.4.6 Applicazioni semplici

La struttura sequenziale può essere utilizzata in numerosi problemi quotidiani e tecnici.

Esempio: costo totale

Conoscendo la quantità acquistata e il prezzo unitario:

$$\text{totale} = \text{quantita} * \text{prezzoUnitario}$$

Esempio: conversione da chilogrammi a grammi

Poiché un chilogrammo corrisponde a 1000 grammi:

$$\text{grammi} = \text{chilogrammi} * 1000$$

Esempio: applicazione facoltativa: volume di una vasca

Conoscendo lunghezza, larghezza e altezza:

$$\text{volume} = \text{lunghezza} * \text{larghezza} * \text{altezza}$$

Il diagramma è identico a quello dell'area del rettangolo, ma contiene tre input e una moltiplicazione tra tre valori.

Esercizi: algoritmi sequenziali

1. Disegnare il diagramma di flusso che acquisisce due numeri, ne calcola la differenza e comunica il risultato.
2. Disegnare il diagramma che acquisisce un numero e ne comunica il quadrato.
3. Acquisire base e altezza di un triangolo e calcolare $\text{area} = \text{base} * \text{altezza} / 2$.
4. Calcolare il perimetro di un quadrato conoscendo il lato.
5. Acquisire tre numeri, calcolarne la somma e la media e comunicare entrambi i risultati.
6. Calcolare il costo totale conoscendo quantità e prezzo unitario.
7. Calcolare il resto da consegnare conoscendo l'importo pagato e il costo dell'acquisto.
8. Convertire una lunghezza espressa in metri in centimetri e millimetri, producendo due output.
9. Calcolare la superficie totale di un terreno rettangolare e il costo della semina conoscendo il costo per metro quadrato.
10. Calcolare la quantità di soluto, espressa in grammi, conoscendo il volume della soluzione e la quantità di soluto per litro.
11. Costruire una tabella di esecuzione per l'esercizio 3 usando base 10 e altezza 6.
12. Individuare e correggere l'errore nell'istruzione $\text{media} = a + b + c / 3$.
13. Spiegare perché l'algoritmo OUT area, IN base, IN altezza, $\text{area} = \text{base} * \text{altezza}$ non può funzionare correttamente.
14. Proporre almeno tre casi di prova per verificare un algoritmo che calcola il costo totale.
15. Modificare il diagramma della somma affinché produca l'output "La somma è: " + somma.

4.5 Le decisioni

Gli algoritmi sequenziali eseguono sempre le stesse istruzioni. Molti problemi, però, richiedono di scegliere cosa fare in base ai dati disponibili. Per esempio, uno sconto deve essere applicato soltanto quando il prezzo supera una determinata soglia, mentre un allarme deve essere attivato soltanto quando una misura non rientra nei limiti stabiliti.

La struttura che permette di scegliere tra percorsi differenti prende il nome di **selezione** o **decisione**. Nei diagrammi di flusso viene rappresentata con un rombo.

4.5.1 Dalla sequenza alla scelta

Nel rombo viene scritta una **condizione**, cioè un'espressione che può assumere soltanto uno dei due risultati logici:

- **V**, se la condizione è vera;
- **F**, se la condizione è falsa.

La condizione viene scritta senza punto interrogativo. L'algoritmo valuta la condizione e segue esclusivamente uno dei due rami.

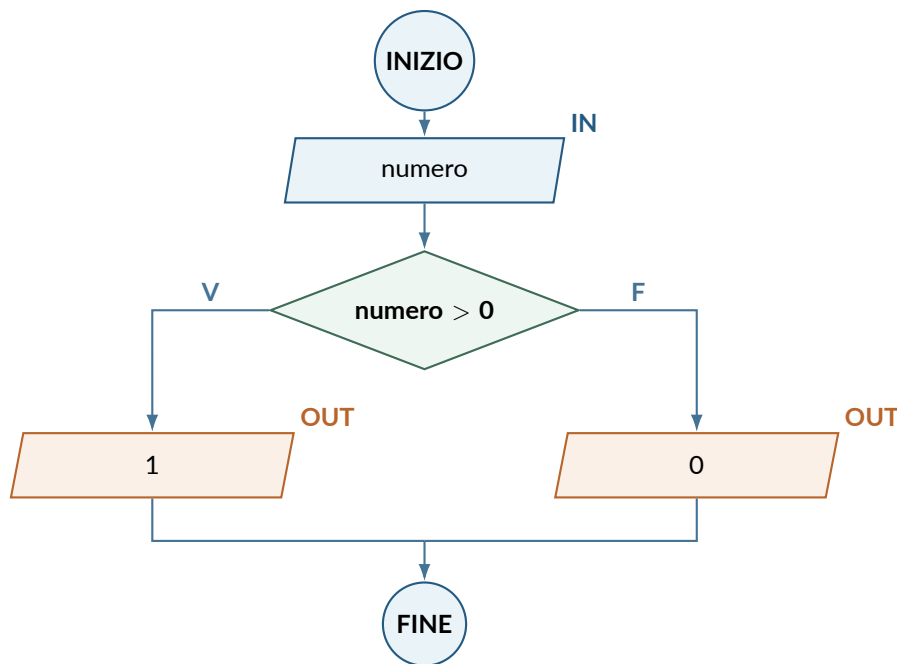


Figura 4.10: Una decisione sceglie uno dei due percorsi in base al risultato della condizione.

Nell'esempio, se `numero` contiene 8, la condizione `numero > 0` è vera e viene seguito il ramo V. Se contiene `-3` oppure `0`, la condizione è falsa e viene seguito il ramo F.

Errore comune: il rombo valuta una condizione

Il rombo valuta una condizione e decide quale percorso seguire. La condizione può contenere confronti, operatori logici e i calcoli necessari a ottenere vero o falso, come `numero % 2 == 0`. Gli assegnamenti, che memorizzano un risultato in una variabile, si rappresentano invece nei rettangoli.

4.5.2 Gli operatori di confronto

Le condizioni vengono costruite utilizzando gli operatori di confronto.

Operatore	Significato	Esempio
>	Maggiore di	temperatura > 30
<	Minore di	umidità < 40
>=	Maggiore o uguale	voto >= 6
<=	Minore o uguale	pressione <= 100
==	Uguale a	numero == 0
!=	Diverso da	numero != 0

Tabella 4.6: Operatori di confronto usati nelle condizioni.

Errore comune: assegnamento e confronto

Il simbolo = assegna un valore, mentre == confronta due valori. L'istruzione `x = 5` memorizza 5 in x; la condizione `x == 5` controlla se il valore di x è uguale a 5.

Anche gli operatori inclusivi richiedono attenzione:

- `prezzo > 100` è falso quando il prezzo è esattamente 100;
- `prezzo >= 100` è vero anche quando il prezzo è esattamente 100.

4.5.3 Decisione semplice: il maggiore tra due numeri

Si vuole acquisire due numeri e comunicare il maggiore. Dopo gli input, il rombo confronta i due valori.

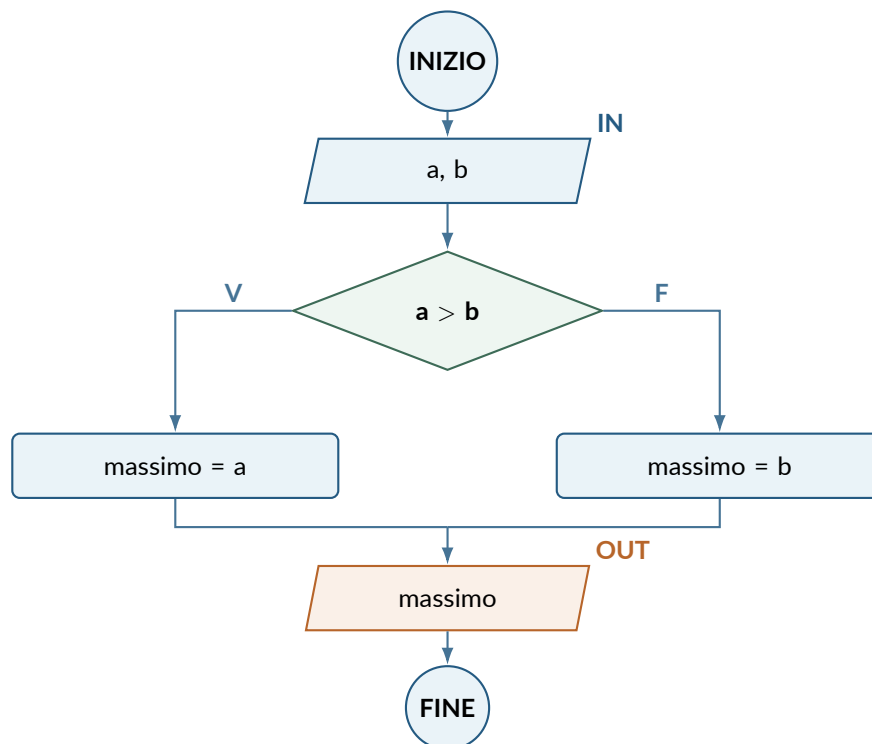


Figura 4.11: Diagramma di flusso per individuare il maggiore tra due numeri.

Se la condizione $a > b$ è vera, viene eseguita l'assegnazione $\text{massimo} = a$; altrimenti viene eseguita $\text{massimo} = b$. I due rami si ricongiungono prima dell'output.

Approfondimento: che cosa accade quando i numeri sono uguali

Se a e b sono uguali, la condizione $a > b$ è falsa. Il ramo F assegna b a massimo ; il risultato è comunque corretto perché i due valori coincidono. Per distinguere esplicitamente il caso di uguaglianza servirebbe una seconda decisione.

4.5.4 Decisioni annidate: positivo, negativo o zero

Una sola condizione permette di distinguere due casi. Per separare tre possibilità sono necessarie due decisioni.

Il problema può essere affrontato così:

1. controllare se il numero è maggiore di zero;
2. se la prima condizione è falsa, controllare se il numero è uguale a zero;
3. il caso rimanente corrisponde necessariamente a un numero negativo.

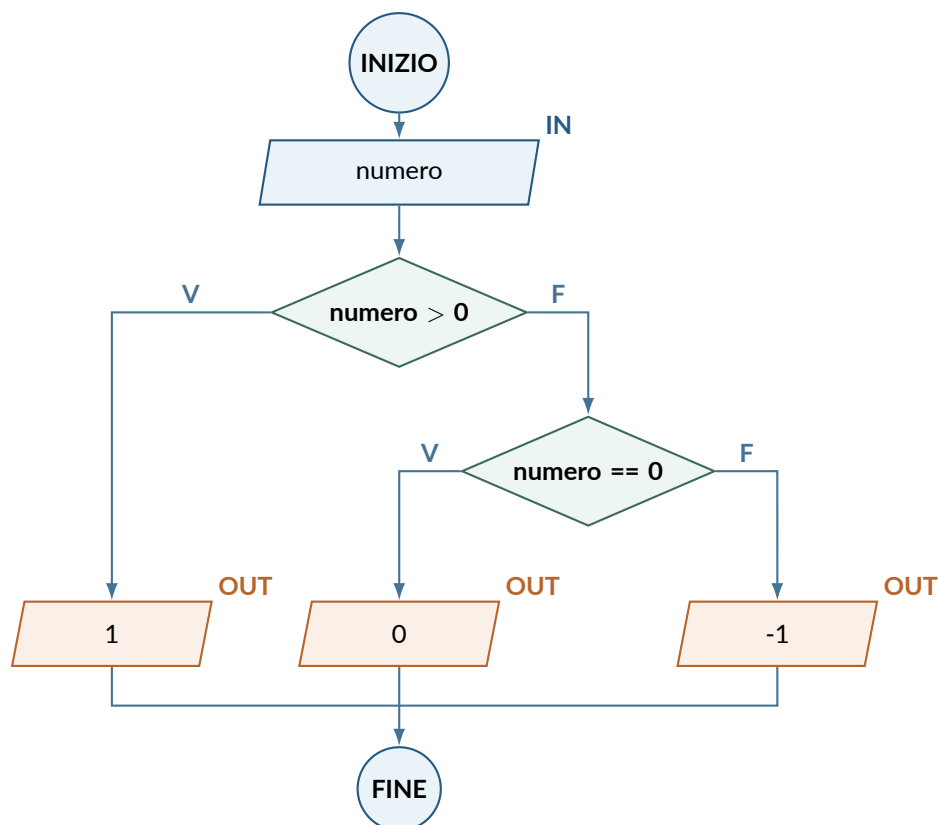


Figura 4.12: Due decisioni annidate distinguono un numero positivo, nullo o negativo.

Una decisione è detta **annidata** quando si trova all'interno di uno dei rami di un'altra decisione.

Esempio: verifica dei tre percorsi

Per controllare il diagramma devono essere provati almeno tre valori:

- 5, che segue il ramo positivo;
- 0, che segue il ramo nullo;
- -4, che segue il ramo negativo.

4.5.5 Calcolare uno sconto

Si consideri la seguente regola:

Se il prezzo supera 100 euro, applicare uno sconto del 10%; altrimenti mantenere il prezzo iniziale.

Gli input e i risultati sono tutti numerici. La condizione determina se eseguire o meno il calcolo dello sconto.

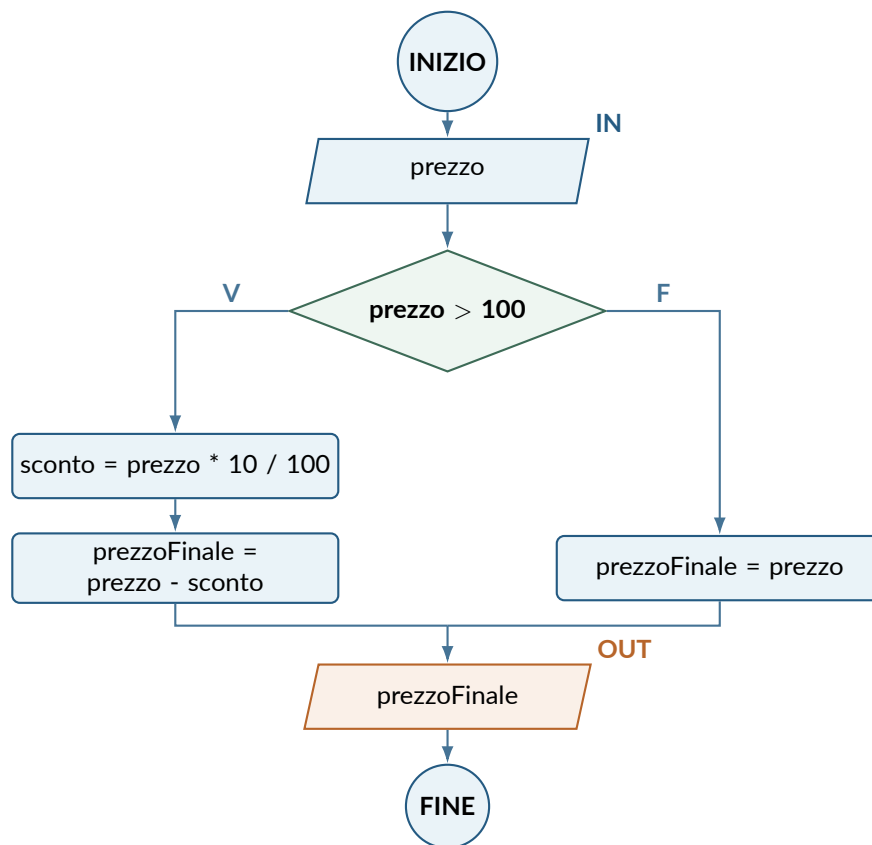


Figura 4.13: Calcolo del prezzo finale con uno sconto applicato oltre la soglia di 100 euro.

Nel ramo V vengono eseguiti due calcoli:

$$\text{sconto} = \text{prezzo} * 10 / 100$$

$$\text{prezzoFinale} = \text{prezzo} - \text{sconto}$$

Nel ramo F il prezzo finale coincide con il prezzo iniziale:

$$\text{prezzoFinale} = \text{prezzo}$$

Esempio: controllo della soglia

Con $\text{prezzo} = 150$, la condizione è vera:

$$\text{sconto} = 150 \cdot \frac{10}{100} = 15$$

$$\text{prezzo finale} = 150 - 15 = 135$$

Con $\text{prezzo} = 100$, la condizione $\text{prezzo} > 100$ è falsa e il prezzo finale rimane 100.

4.5.6 Sconti su più fasce

Una politica commerciale può prevedere più soglie:

- oltre 250 euro: sconto del 20%;
- oltre 100 euro e fino a 250 euro: sconto del 10%;
- fino a 100 euro: nessuno sconto.

È importante controllare prima la soglia più alta. Se si controllasse per prima $\text{prezzo} > 100$, anche un prezzo di 300 euro entrerebbe immediatamente nel ramo dello sconto del 10%, senza raggiungere il controllo successivo.

Esempio: ordine corretto delle condizioni

Una possibile struttura è:

1. controllare $\text{prezzo} > 250$;
2. nel ramo F controllare $\text{prezzo} > 100$;
3. assegnare rispettivamente le percentuali 20, 10 oppure 0;
4. calcolare lo sconto e il prezzo finale.

Errore comune: dimenticare il caso limite

Quando una regola utilizza parole come “oltre”, “almeno”, “fino a” oppure “non più di”, bisogna scegliere con attenzione tra $>$, $\geq$, $<$ e $\leq$. Il valore esattamente uguale alla soglia deve essere sempre incluso in uno dei casi.

4.5.7 Verificare entrambi i rami

Un algoritmo con decisioni deve essere provato con dati che attraversino tutti i percorsi. Per la condizione $\text{prezzo} > 100$ sono utili almeno tre casi:

Prezzo	Risultato della condizione	Motivo del test
80	F	Valore inferiore alla soglia
100	F	Valore esattamente uguale alla soglia
150	V	Valore superiore alla soglia

Tabella 4.7: Casi di prova per una condizione con soglia.

Il valore uguale alla soglia è chiamato spesso **caso limite**: è particolarmente utile per individuare un operatore di confronto scelto in modo errato.

Esercizi: decisioni e operatori di confronto

1. Indicare se le seguenti condizioni sono vere o false quando $a = 8$ e $b = 5$: $a > b$, $a < b$, $a == 8$, $b != 5$, $a >= 8$.
2. Spiegare la differenza tra $x = 10$ e $x == 10$.
3. Disegnare un diagramma che acquisisca un numero e comunichi 1 se è maggiore di zero, 0 altrimenti.
4. Disegnare un diagramma che acquisisca due numeri e comunichi il minore.
5. Modificare la [figura 4.11](#) per distinguere anche il caso in cui i due numeri siano uguali.
6. Disegnare un algoritmo che acquisisca un voto e comunichi 1 se il voto è almeno 6, 0 altrimenti.
7. Acquisire una temperatura e comunicare un valore di allarme uguale a 1 quando la temperatura supera 35, altrimenti 0.
8. Acquisire un prezzo e applicare uno sconto del 5% quando il prezzo è almeno 50 euro.
9. Acquisire quantità e prezzo unitario, calcolare il totale e applicare uno sconto del 10% quando il totale supera 200 euro.
10. Costruire il diagramma per le tre fasce di sconto descritte nel testo.
11. Acquisire un numero e stabilire se è positivo, negativo o nullo utilizzando due decisioni.
12. Individuare l'errore logico: per applicare uno sconto agli importi di almeno 100 euro viene usata la condizione `prezzo > 100`.
13. Proporre tre casi di prova per un algoritmo che controlla se una temperatura è inferiore a 5 gradi.
14. Applicazione facoltativa: acquisire l'umidità del terreno e assegnare `irrigazione = 1` se il valore è inferiore a 30, altrimenti assegnare 0.
15. Applicazione facoltativa: acquisire il valore di pH e assegnare `acido = 1` quando il valore è minore di 7, altrimenti assegnare 0.

4.6 Le condizioni composte

Una condizione semplice confronta due valori, per esempio `temperatura > 30`. In molti problemi, però, una decisione dipende da **più condizioni contemporaneamente**. Per rappresentare questi casi si utilizzano gli **operatori logici**, che permettono di combinare più confronti in un'unica espressione.

In questo capitolo verranno utilizzati due operatori:

- `&&`, letto *AND*, che richiede che entrambe le condizioni siano vere;
- `||`, letto *OR*, che richiede che almeno una delle condizioni sia vera.

Ogni confronto deve essere completo. Per verificare che `x` sia compreso tra 10 e 20 non si scrive `10 <= x <= 20`, ma si combinano due condizioni:

`(x >= 10) && (x <= 20)`

4.6.1 L'operatore AND: entrambe le condizioni devono essere vere

L'operatore `&&` produce un risultato vero soltanto quando sono vere sia la condizione a sinistra sia quella a destra.

Condizione A	Condizione B	A && B
F	F	F
F	V	F
V	F	F
V	V	V

Tabella 4.8: Tabella di verità dell'operatore AND.

Si vuole controllare se una temperatura è compresa tra 18 e 25 gradi, estremi inclusi. Devono essere vere contemporaneamente le condizioni:

`temperatura >= 18      e      temperatura <= 25`

La condizione composta è quindi:

`(temperatura >= 18) && (temperatura <= 25)`

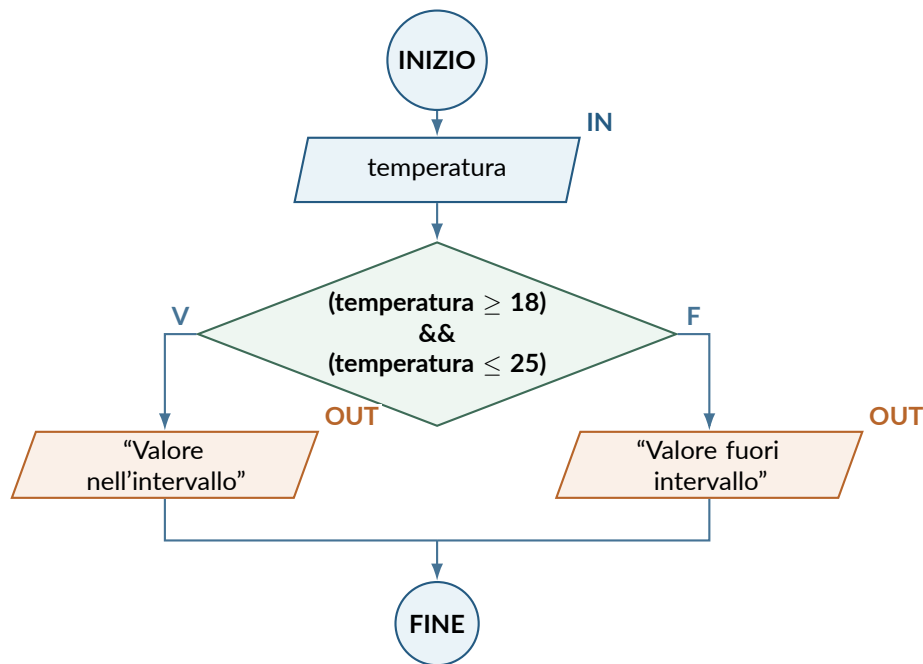


Figura 4.14: Uso dell'operatore AND per verificare che un valore appartenga a un intervallo.

Esempio: controllo della condizione AND

Con temperatura = 22, entrambi i confronti sono veri e viene seguito il ramo V.
 Con temperatura = 28, il primo confronto è vero ma il secondo è falso: l'intera condizione composta è quindi falsa.

Errore comune: AND non significa "una delle due"

La condizione (voto ≥ 6) && (presenze ≥ 75) è vera soltanto se entrambi i requisiti sono soddisfatti. Se ne è soddisfatto soltanto uno, il risultato è falso.

4.6.2 L'operatore OR: almeno una condizione deve essere vera

L'operatore `||` produce un risultato vero quando almeno una delle condizioni è vera. Il risultato è falso soltanto quando entrambe sono false.

Condizione A	Condizione B	A B
F	F	F
F	V	V
V	F	V
V	V	V

Tabella 4.9: Tabella di verità dell'operatore OR.

Si vuole attivare un allarme quando la temperatura è inferiore a 5 gradi oppure superiore a 35 gradi. È sufficiente che si verifichi una delle due situazioni:

`(temperatura < 5) || (temperatura > 35)`

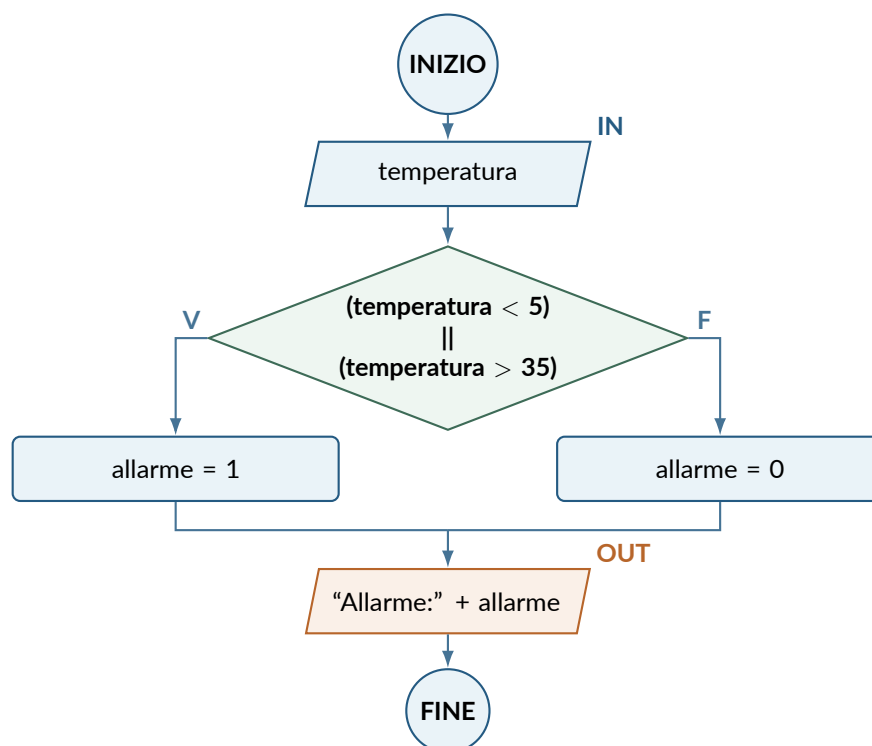


Figura 4.15: Uso dell'operatore OR per riconoscere valori esterni a un intervallo.

Esempio: controllo della condizione OR

Con temperatura = 2, la prima condizione è vera e l'allarme viene attivato. Con temperatura = 40, è vera la seconda condizione e l'allarme viene attivato ugualmente. Con temperatura = 20, entrambe sono false e l'allarme rimane disattivato.

4.6.3 Intervalli interni ed esterni

Gli operatori AND e OR vengono spesso utilizzati per distinguere due situazioni opposte:

- per controllare che un valore sia **dentro** un intervallo si usa normalmente AND;
- per controllare che un valore sia **fuori** da un intervallo si usa normalmente OR.

Situazione	Condizione	Significato
Dentro l'intervallo	$(x \geq 10) \ \&\& \ (x \leq 20)$	x è compreso tra 10 e 20
Fuori dall'intervallo	$(x < 10) \ \ (x > 20)$	x è minore di 10 oppure maggiore di 20

Tabella 4.10: Confronto tra il controllo interno ed esterno a un intervallo.

Approfondimento: perché si usano le parentesi

Le parentesi rendono evidente quali confronti vengono combinati. La scrittura $(x \geq 10) \ \&\& \ (x \leq 20)$ è più leggibile e riduce la possibilità di commettere errori. Nei primi algoritmi verranno quindi sempre utilizzate, anche quando il linguaggio potrebbe accettare la condizione senza parentesi.

4.6.4 Applicare uno sconto con due requisiti

Un negozio applica uno sconto del 10% soltanto quando il cliente acquista almeno 5 prodotti e il totale supera 50 euro. I due requisiti devono essere entrambi soddisfatti:

$(\text{quantita} \geq 5) \ \&\& \ (\text{totale} > 50)$

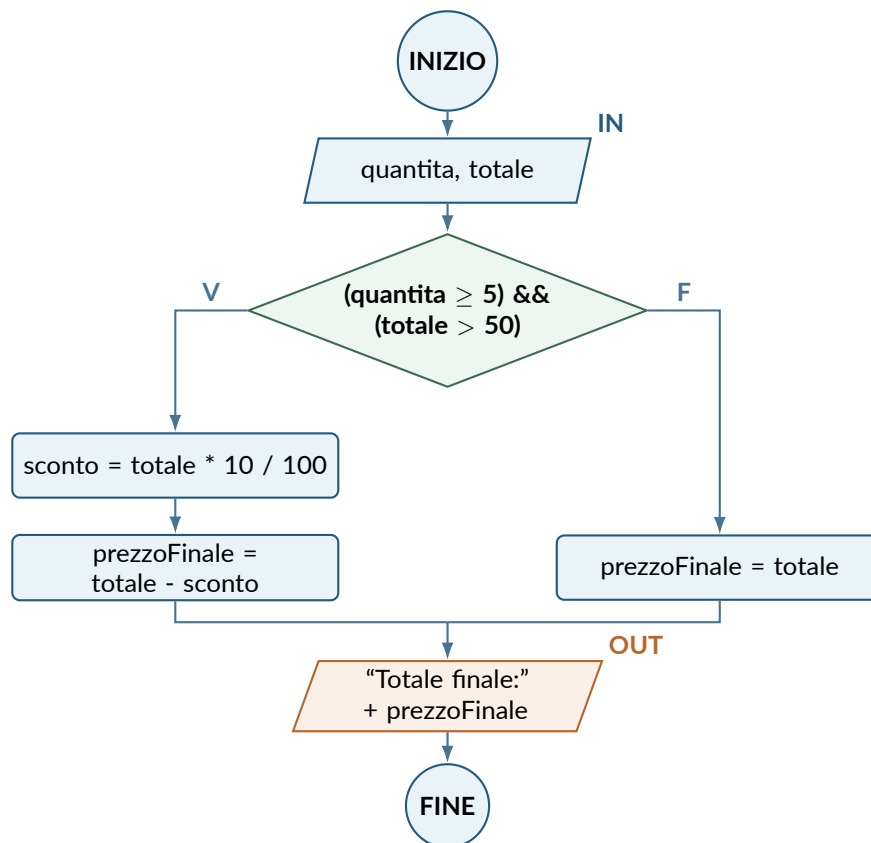


Figura 4.16: Uno sconto viene applicato soltanto quando sono veri entrambi i requisiti.

Esempio: tre casi di prova

- $\text{quantita} = 6, \text{totale} = 80$: entrambi i requisiti sono veri, quindi si applica lo sconto;
- $\text{quantita} = 3, \text{totale} = 80$: il primo requisito è falso, quindi lo sconto non si applica;
- $\text{quantita} = 6, \text{totale} = 40$: il secondo requisito è falso, quindi lo sconto non si applica.

4.6.5 Scegliere tra AND e OR

Per scegliere l'operatore corretto è utile osservare le parole del problema:

- espressioni come “entrambi”, “contemporaneamente”, “e” oppure “tutti i requisiti” suggeriscono `&&`;
- espressioni come “almeno uno”, “oppure”, “uno dei due casi” suggeriscono `||`.

La scelta deve comunque essere verificata attraverso esempi concreti. La sola presenza della parola “e” o “oppure” nel testo non sostituisce l'analisi logica del problema.

Esercizi: condizioni composte

1. Stabilire il risultato delle condizioni quando $x = 12$: $(x > 5) \ \&\& \ (x < 20)$, $(x < 5) \ || \ (x == 12)$, $(x > 20) \ || \ (x < 0)$.
2. Scrivere la condizione che verifica se un numero è compreso tra 1 e 10, estremi inclusi.
3. Scrivere la condizione che verifica se un numero è esterno all'intervallo da 1 a 10.
4. Disegnare un diagramma che acquisisca un voto e comunichi "Valore valido" quando è compreso tra 1 e 10, altrimenti "Valore non valido".
5. Acquisire una temperatura e assegnare $allarme = 1$ quando è inferiore a 0 oppure superiore a 40; assegnare 0 negli altri casi.
6. Acquisire età e altezza e assegnare $ammesso = 1$ soltanto quando l'età è almeno 12 e l'altezza è almeno 140.
7. Applicare uno sconto del 15% quando il totale è almeno 100 euro e la quantità acquistata è almeno 10.
8. Individuare l'errore nella condizione usata per verificare che x sia compreso tra 5 e 15: $(x >= 5) \ || \ (x <= 15)$.
9. Spiegare perché $(temperatura < 5) \ \&\& \ (temperatura > 35)$ non può essere vera per un singolo valore.
10. Proporre almeno quattro casi di prova per verificare la condizione $(pH >= 6) \ \&\& \ (pH <= 8)$.
11. Applicazione facoltativa: attivare l'irrigazione quando l'umidità è inferiore a 30 e la temperatura è superiore a 20.
12. Applicazione facoltativa: attivare un allarme quando il pH è inferiore a 5 oppure superiore a 9.

4.7 Il resto della divisione: l'operatore modulo

La divisione tra numeri interi produce un quoziente e, in alcuni casi, un resto. L'**operatore modulo**, rappresentato dal simbolo %, permette di ottenere direttamente tale resto.

Definizione: operatore modulo

L'espressione $a \% b$ restituisce il resto della divisione intera di a per b .

Per esempio:

$$10 \% 3 = 1 \quad 12 \% 4 = 0 \quad 17 \% 5 = 2$$

Nel primo caso, 10 diviso 3 produce quoziente 3 e resto 1. Nel secondo caso la divisione è esatta, quindi il resto è zero.

Errore comune: il simbolo % non indica una percentuale

Nelle espressioni degli algoritmi, % indica il resto della divisione intera. Il calcolo di una percentuale continua a essere scritto, per esempio, come $\text{sconto} = \text{prezzo} * 10 / 100$.

4.7.1 Verificare se un numero è pari o dispari

Ogni numero pari è divisibile per 2 senza resto. Di conseguenza:

$$\text{numero} \% 2 == 0$$

è vera quando numero è pari. Se il resto è diverso da zero, il numero è dispari.

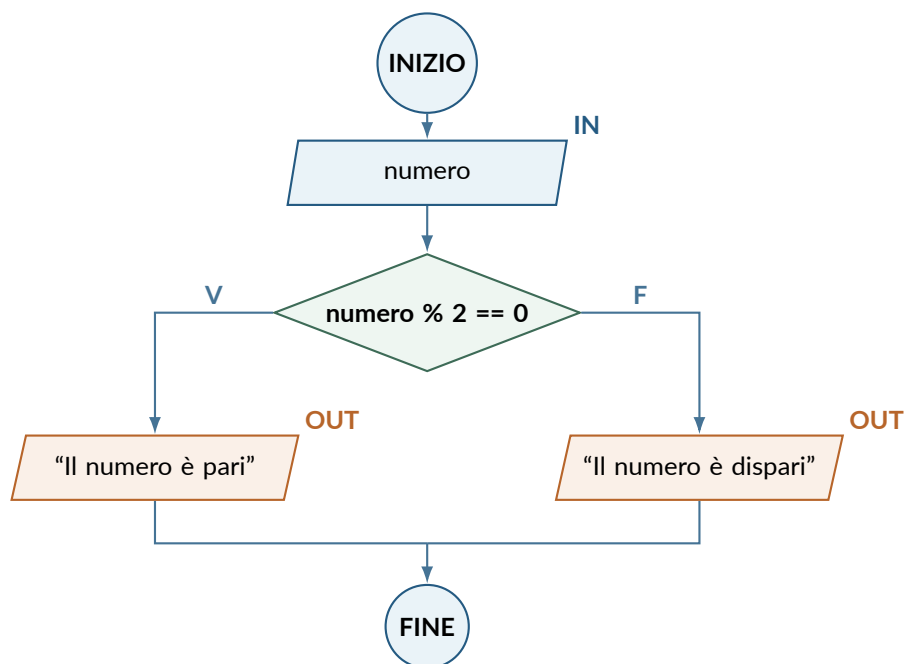


Figura 4.17: Uso dell'operatore modulo per distinguere numeri pari e dispari.

Esempio: simulazione

Con `numero = 18`, l'espressione `18 % 2` produce 0 e viene seguito il ramo V. Con `numero = 7`, il resto è 1 e viene seguito il ramo F.

4.7.2 Verificare la divisibilità e i multipli

Un numero a è divisibile per un numero b quando il resto della divisione è zero:

$$a \% b == 0$$

Per esempio:

- `24 % 6 == 0`, quindi 24 è multiplo di 6;
- `25 % 6 == 1`, quindi 25 non è multiplo di 6;
- `45 % 5 == 0`, quindi 45 è divisibile per 5.

Esempio: controllare un multiplo

Per verificare se un valore è multiplo di 3, nel rombo si scrive:

$$\text{numero} \% 3 == 0$$

Il ramo V corrisponde ai multipli di 3, mentre il ramo F corrisponde a tutti gli altri numeri.

4.7.3 Estrarre l'ultima cifra

Il resto della divisione per 10 coincide con l'ultima cifra di un numero intero non negativo, compreso lo zero.

$$\text{ultimaCifra} = \text{numero} \% 10$$

Per esempio:

$$347 \% 10 = 7 \quad 120 \% 10 = 0$$

Questo procedimento sarà utile anche nella generazione di numeri pseudocasuali, perché il modulo permette di limitare i risultati a un determinato intervallo.

Approfondimento: quali resti sono possibili

Quando si calcola `numero % n` con `numero` non negativo e `n` positivo, i possibili resti sono compresi tra 0 e $n - 1$. Per esempio, con `% 6` si possono ottenere soltanto 0, 1, 2, 3, 4 oppure 5.

4.7.4 Costruire condizioni con il modulo

Il modulo può essere combinato con gli operatori di confronto e con gli operatori logici.

- `numero % 5 == 0`: il numero è multiplo di 5;
- `numero % 2 != 0`: il numero è dispari;
- `(numero % 3 == 0) && (numero % 5 == 0)`: il numero è multiplo sia di 3 sia di 5;
- `(numero % 2 == 0) || (numero % 3 == 0)`: il numero è divisibile per 2 oppure per 3.

Errore comune: dividere per zero

L'espressione `a % 0` non è definita, così come non è possibile eseguire una divisione per zero. Il secondo operando del modulo deve quindi essere diverso da zero.

Esercizi: operatore modulo

1. Calcolare: `14 % 3`, `25 % 5`, `29 % 4`, `100 % 9`.
2. Indicare quali numeri tra 12, 17, 24, 35 e 41 sono pari.
3. Scrivere la condizione che verifica se un numero è dispari.
4. Disegnare un diagramma che acquisisca un numero e comunichi "Multiplo di 5" oppure "Non multiplo di 5".
5. Disegnare un diagramma che acquisisca due numeri `a` e `b` e comunichi se `a` è divisibile per `b`. Considerare soltanto valori di `b` diversi da zero.
6. Calcolare l'ultima cifra dei numeri 128, 905 e 3472 utilizzando il modulo.
7. Acquisire un numero intero non negativo e assegnare alla variabile `ultimaCifra` la sua ultima cifra, quindi produrre l'output "L'ultima cifra è: " + `ultimaCifra`.
8. Scrivere la condizione che verifica se un numero è multiplo sia di 4 sia di 6.
9. Scrivere la condizione che verifica se un numero è divisibile per 3 oppure per 7.
10. Spiegare perché i possibili risultati di `numero % 10` sono soltanto i valori da 0 a 9.
11. Individuare l'errore nell'affermazione: "numero % 6 genera un valore compreso tra 1 e 6".
12. Applicazione facoltativa: i campioni di laboratorio sono numerati progressivamente. Stabilire se il numero di un campione è multiplo di 4 e deve quindi essere sottoposto a un controllo aggiuntivo.

4.8 Le iterazioni nei diagrammi di flusso

Gli algoritmi studiati finora eseguono ogni istruzione una sola volta, scegliendo eventualmente tra percorsi differenti. Molti problemi richiedono invece di ripetere la stessa operazione: acquisire più misure, contare gli elementi di un insieme, sommare una serie di valori oppure chiedere nuovamente un dato non valido.

Riscrivere molte volte le stesse istruzioni renderebbe l'algoritmo lungo, rigido e difficile da correggere. Una **struttura iterativa**, chiamata anche **ciclo**, permette di rappresentare una sola volta il gruppo di istruzioni da ripetere e di stabilire quando la ripetizione deve terminare.

Definizione: iterazione

Un'**iterazione** è una singola esecuzione del gruppo di istruzioni che deve essere ripetuto. Questo gruppo prende il nome di **corpo del ciclo**.

Se un diagramma comunica i numeri da 1 a 5, il corpo viene attraversato cinque volte: ciascun passaggio costituisce un'iterazione.

4.8.1 La freccia che torna indietro

In un algoritmo sequenziale le frecce procedono sempre verso il basso. In una struttura iterativa, invece, una freccia riporta l'esecuzione verso un punto già attraversato. Questo ritorno indica che alcune istruzioni devono essere eseguite nuovamente.

La freccia di ritorno non è una scorciatoia grafica: deve condurre con precisione al simbolo dal quale ricomincia il controllo o l'azione. Per rendere il diagramma leggibile, nel manuale verrà disegnata all'esterno del flusso principale, con segmenti orizzontali e verticali e senza attraversare altri simboli.

Approfondimento: leggere una ripetizione

Per comprendere una struttura iterativa bisogna seguire le frecce nell'ordine e domandarsi:

1. quali istruzioni vengono ripetute;
2. quale condizione decide se tornare indietro;
3. quali valori cambiano durante ogni passaggio;
4. in quale situazione la freccia non torna indietro e l'algoritmo prosegue verso la fine.

4.8.2 Gli elementi fondamentali

Una ripetizione contiene normalmente quattro elementi.

■ Inizializzazione

Prima di iniziare si assegnano i valori iniziali alle variabili utilizzate. Per esempio, un contatore può partire da 1 e una somma può partire da 0.

Condizione

Una domanda stabilisce se eseguire o ripetere il corpo oppure se proseguire oltre il ciclo. La condizione deve essere formulata in modo preciso.

Corpo della ripetizione

È l'insieme delle istruzioni attraversate più volte. Può contenere input, output, operazioni e altre decisioni.

Aggiornamento

Una o più variabili vengono modificate. L'aggiornamento deve avvicinare l'algoritmo alla condizione che farà terminare la ripetizione.

Errore comune: una condizione che non cambia mai

Se nessuna istruzione modifica i valori utilizzati nella condizione, la freccia può continuare a tornare indietro per sempre. Si ottiene così un **ciclo infinito**, cioè un algoritmo che non raggiunge mai la fine.

4.8.3 Contatori, accumulatori e sentinelle

Nelle iterazioni si incontrano frequentemente tre tipi di variabili.

Definizione: contatore

Un **contatore** registra quante volte è avvenuta un'operazione oppure quale ripetizione è in corso. Viene spesso aggiornato con un assegnamento come $i = i + 1$.

Definizione: accumulatore

Un **accumulatore** raccoglie progressivamente un risultato, per esempio una somma. Prima della ripetizione viene normalmente inizializzato a 0 e poi aggiornato con un assegnamento come $\text{totale} = \text{totale} + \text{valore}$.

Definizione: sentinella

Una **sentinella** è un valore speciale che non deve essere elaborato come dato, ma segnala che l'inserimento deve terminare. Per esempio, il valore 0 può indicare che non devono essere acquisiti altri numeri.

Variabile	Esempio di aggiornamento	Possibile utilizzo
Contatore	$i = i + 1$	contare misure, ripetizioni o elementi elaborati
Accumulatore	$\text{somma} = \text{somma} + \text{valore}$	calcolare progressivamente un totale
Sentinella	$\text{numero} == 0$	indicare che l'inserimento dei dati è terminato

Tabella 4.11: Ruolo di alcune variabili utilizzate nelle iterazioni.

Esercizi: riconoscere la ripetizione

1. Spiegare con parole proprie la differenza tra ciclo e iterazione.
2. Individuare corpo, condizione e aggiornamento nell'attività: "finché il serbatoio non è pieno, aggiungere un litro d'acqua e aggiornare la quantità contenuta".
3. Stabilire quale variabile può essere usata come contatore per elaborare 12 campioni numerati da 1 a 12.
4. Scrivere l'assegnamento necessario per incrementare di uno la variabile campione.
5. Scrivere l'assegnamento necessario per aggiungere misura all'accumulatore totale.
6. Indicare quale valore iniziale deve avere un accumulatore usato per calcolare una somma.
7. Proporre una sentinella adatta a terminare l'inserimento di temperature comprese tra -20 e 50 gradi e motivare la scelta.
8. Spiegare perché una ripetizione che contiene la condizione $i \leq 10$ ma non modifica mai i può non terminare.

4.9 Esempio 1: sommare valori fino a una sentinella

Si vogliono acquisire e sommare numeri interi. Il valore 0 segnala la fine dell'inserimento e non deve essere aggiunto al totale.

Prima di iniziare bisogna:

- inizializzare l'accumulatore con $\text{totale} = 0$;
- acquisire il primo valore nella variabile `numero`.

Il rombo controlla se `numero != 0`. Se la risposta è vera, il valore viene aggiunto al totale e viene acquisito un nuovo numero. La freccia esterna riporta quindi al rombo. Se la

risposta è falsa, l'algoritmo comunica il totale e termina.

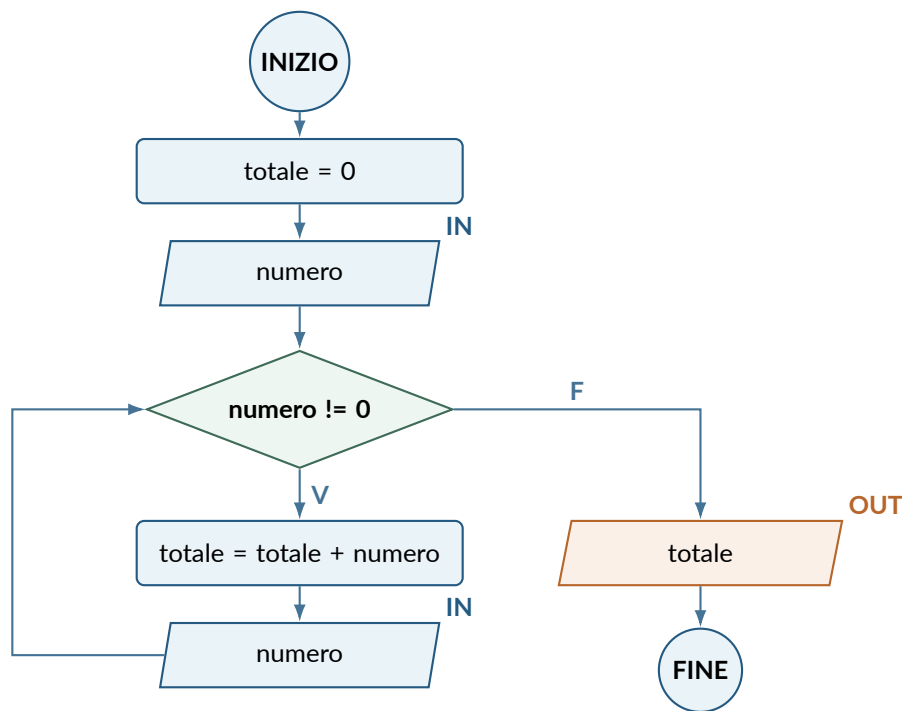


Figura 4.18: Somma di numeri fino all'inserimento della sentinella 0. La freccia di ritorno conduce al controllo prima di una nuova possibile elaborazione.

4.9.1 Perché il primo input precede il rombo

La condizione deve controllare un valore già presente nella variabile `numero`. Per questo motivo il primo input viene eseguito prima del rombo. L'input collocato nel corpo prepara invece il valore che verrà controllato al passaggio successivo.

Se il primo numero inserito è 0, la condizione è immediatamente falsa: il corpo non viene attraversato neppure una volta e il totale resta 0. Questa possibilità è corretta, perché potrebbe non esserci alcun dato da elaborare.

4.9.2 Seguire il diagramma passo dopo passo

Supponiamo che vengano inseriti, nell'ordine, i valori 4, 3 e 0.

Controllo	numero	totale dopo il corpo	Risultato
1	4	4	risposta vera, prima iterazione
2	3	7	risposta vera, seconda iterazione
3	0	7	risposta falsa, uscita dalla ripetizione

Tabella 4.12: Traccia dell'esecuzione con sentinella.

La sentinella 0 non viene sommata perché il percorso esce dal rombo prima di raggiungere l'operazione `totale = totale + numero`.

Errore comune: dimenticare il nuovo input

Se nel corpo manca il secondo IN numero, il valore della variabile non cambia. Quando il primo numero è diverso da zero, la condizione rimane sempre vera e la freccia continua a tornare indietro.

Errore comune: controllare la condizione opposta

La condizione del rombo deve indicare quando entrare nel corpo. Con sentinella 0 si controlla numero $\neq 0$, perché il dato deve essere elaborato soltanto quando è diverso dalla sentinella.

Esercizi: ripetere fino a una sentinella

1. Stabilire che cosa accade nel diagramma se il primo valore inserito è 0.
2. Costruire la tabella di esecuzione per i valori 5, 2, 8 e 0.
3. Modificare il diagramma affinché comunichi anche il numero di valori inseriti.
4. Disegnare un diagramma che acquisisca numeri fino alla sentinella -1 e conti quanti dati sono stati inseriti. La sentinella non deve essere conteggiata.
5. Acquisire temperature finché non viene inserito 999 e calcolarne la somma.
6. Acquisire valori di umidità fino all'inserimento di -1 e contare quanti sono inferiori a 30.
7. Spiegare perché la sentinella deve essere scelta fuori dall'insieme dei valori validi, quando ciò è possibile.
8. Individuare l'errore in un diagramma nel quale il nuovo input è collocato dopo la freccia di ritorno.

4.10 Esempio 2: chiedere nuovamente un dato non valido

Un algoritmo deve acquisire un voto compreso tra 1 e 10. Poiché almeno un valore deve essere inserito, l'input compare prima della condizione. Il rombo domanda se il voto è fuori dall'intervallo consentito:

$$(\text{voto} < 1) \mid \mid (\text{voto} > 10)$$

Se la risposta è vera, la freccia esterna torna all'input e viene chiesto un nuovo valore. Se la risposta è falsa, il voto è valido e l'algoritmo prosegue.

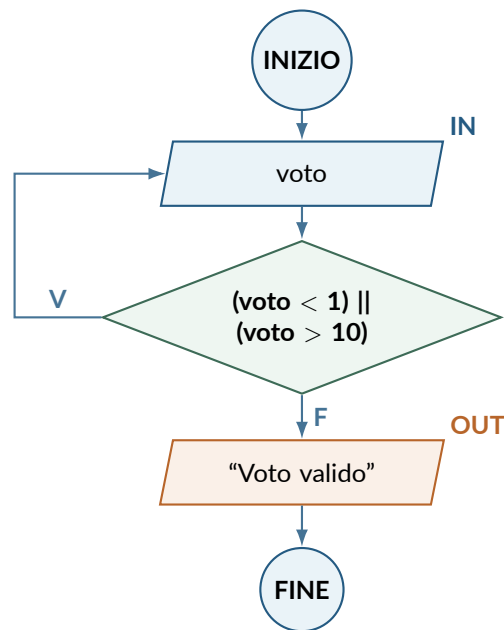


Figura 4.19: Acquisizione ripetuta di un voto. L'input viene eseguito almeno una volta e il controllo decide se deve essere ripetuto.

4.10.1 Il controllo avviene dopo l'azione

In questo esempio non serve un input iniziale separato, perché il corpo da ripetere consiste proprio nell'acquisizione del voto. La prima esecuzione è inevitabile: senza un valore non sarebbe possibile controllare la validità.

Se il primo voto è 7, l'input viene attraversato una sola volta. Se vengono inseriti -2, 15 e 8, l'input viene attraversato tre volte; soltanto il terzo valore permette di proseguire verso l'output.

4.10.2 Condizione di errore e condizione di validità

La freccia di ritorno parte dal ramo che rappresenta un dato errato. La condizione di errore è:

$$(\text{voto} < 1) \ || \ (\text{voto} > 10)$$

La condizione opposta, che descrive un voto valido, è:

$$(\text{voto} \geq 1) \ \&\& \ (\text{voto} \leq 10)$$

Entrambe sono corrette, ma conducono a rami differenti. Prima di scrivere la condizione bisogna quindi stabilire quale risposta deve riportare all'input e quale deve far proseguire l'algoritmo.

Approfondimento: la domanda deve corrispondere ai rami

Una condizione non va valutata isolatamente. È necessario leggere insieme il testo nel rombo, le etichette V e F e la destinazione delle due frecce.

Errore comune: usare AND per riconoscere un valore esterno

La condizione `(voto < 1) && (voto > 10)` non può mai essere vera: nessun numero può essere contemporaneamente minore di 1 e maggiore di 10. Per riconoscere i valori esterni all'intervallo bisogna utilizzare OR.

Esercizi: ripetere un input non valido

1. Seguire il diagramma con gli input 12, 0, 6 e indicare quante volte viene attraversato il parallelogramma.
2. Modificare la condizione per accettare un valore compreso tra 0 e 100, estremi inclusi.
3. Disegnare un diagramma che ripeta la richiesta di una percentuale fino all'inserimento di un valore valido.
4. Disegnare un diagramma che acquisisca una temperatura compresa tra -20 e 50 gradi.
5. Costruire una condizione che ripeta l'input quando il numero è uguale a 0.
6. Riscrivere il diagramma usando nel rombo la condizione di validità. Stabilire quale ramo deve tornare all'input.
7. Spiegare perché questo schema esegue l'input almeno una volta.
8. Individuare l'errore nella condizione `(x < 10) || (x > 20)` se il ramo vero deve accettare valori compresi tra 10 e 20. Scrivere la condizione corretta.

4.11 Esempio 3: elaborare un numero stabilito di misure

Un laboratorio deve acquisire esattamente cinque misure e calcolarne la somma. Il numero di ripetizioni è noto prima di iniziare, quindi si usa un contatore.

Prima del controllo vengono inizializzate due variabili:

- `totale = 0`, perché nessuna misura è stata ancora sommata;
- `i = 1`, perché la prima misura sarà identificata dal valore 1.

Il rombo controlla `i <= 5`. Se la risposta è vera, il diagramma acquisisce una misura, la aggiunge al totale e incrementa il contatore. La freccia esterna torna quindi al rombo. Quando `i` diventa 6, la risposta è falsa e viene comunicato il totale.

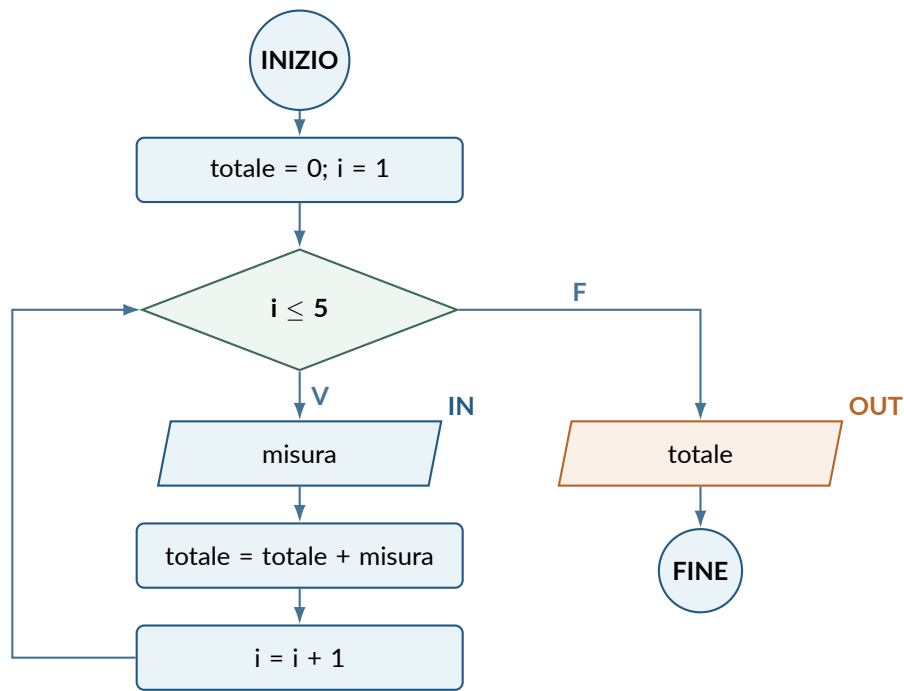


Figura 4.20: Acquisizione e somma di cinque misure mediante un contatore. La freccia di ritorno parte dall'aggiornamento e conduce nuovamente al controllo.

4.11.1 Seguire il valore del contatore

Il contatore assume i valori 1, 2, 3, 4 e 5 durante le cinque elaborazioni. Dopo la quinta misura viene eseguito ancora l'aggiornamento $i = i + 1$, quindi i diventa 6. Il nuovo controllo risulta falso e l'algoritmo esce dalla ripetizione.

Iterazione	i al controllo	Misura elaborata	i dopo l'aggiornamento
1	1	prima	2
2	2	seconda	3
3	3	terza	4
4	4	quarta	5
5	5	quinta	6

Tabella 4.13: Evoluzione del contatore durante l'acquisizione di cinque misure.

4.11.2 Controllare gli estremi

Una piccola modifica alla condizione può cambiare il numero di iterazioni.

Inizializzazione e condizione	Valori elaborati
$i = 1$, poi $i \leq 5$	1, 2, 3, 4, 5: cinque iterazioni
$i = 1$, poi $i < 5$	1, 2, 3, 4: quattro iterazioni
$i = 0$, poi $i < 5$	0, 1, 2, 3, 4: cinque iterazioni
$i = 5$, poi $i \geq 1$, con decremento	5, 4, 3, 2, 1: cinque iterazioni

Tabella 4.14: Effetto dei valori iniziali e degli estremi della condizione.

Un errore di una unità nel valore iniziale o nella condizione viene chiamato spesso **errore off-by-one**: il corpo viene eseguito una volta in più oppure una volta in meno rispetto al necessario.

4.11.3 Calcolare anche la media

Dopo aver acquisito e sommato cinque misure, è possibile calcolare:

$$\text{media} = \text{totale} / 5$$

Questo assegnamento deve essere collocato dopo la ripetizione, quando tutte le misure sono state sommate. Se il numero di misure fosse contenuto in una variabile n , la formula diventerebbe $\text{media} = \text{totale} / n$.

Errore comune: azzerare l'accumulatore nel corpo

L'assegnamento $\text{totale} = 0$ deve precedere la ripetizione. Se venisse eseguito prima di ogni nuovo input, il totale precedente andrebbe perso e alla fine resterebbe soltanto l'ultima misura.

Errore comune: aggiornare nella direzione sbagliata

Se il contatore parte da 1 e la condizione è $i \leq 5$, l'assegnamento $i = i - 1$ allontana il valore dalla conclusione. La condizione rimane vera e la ripetizione non termina.

Esercizi: ripetere un numero stabilito di volte

1. Elencare i valori assunti da i se parte da 2, continua finché $i \leq 8$ e aumenta di 2.
2. Indicare quante iterazioni vengono eseguite se i parte da 0, continua finché $i < 10$ e aumenta di 1.
3. Disegnare un diagramma che comunichi i numeri da 1 a 10.
4. Disegnare un diagramma che comunichi i numeri pari da 2 a 20.
5. Disegnare un diagramma che comunichi i numeri da 10 a 1 usando un decremento.

Esercizi: ripetere un numero stabilito di volte (continua)

6. Acquisire esattamente sei misure e calcolarne la media.
7. Acquisire otto temperature e contare quante sono inferiori a 5 gradi.
8. Correggere un diagramma che dovrebbe elaborare cinque dati ma usa $i = 1$ e la condizione $i < 5$.
9. Spiegare che cosa accade se `totale = 0` viene collocato all'interno del corpo.
10. Progettare un diagramma che simuli dieci lanci di un dado e conti quante volte esce 6.

4.12 Progettare e verificare una ripetizione

I tre esempi precedenti mostrano che la ripetizione può essere organizzata in modi differenti senza introdurre nuovi simboli. La scelta dipende soprattutto dal problema.

Domanda da porsi	Organizzazione del diagramma	Esempio
Potrebbe non esserci alcun dato da elaborare?	acquisire un primo valore, controllarlo e tornare al controllo dopo ogni nuovo input	sommare valori fino a una sentinella
Un'azione deve essere eseguita almeno una volta?	eseguire l'azione, controllarne il risultato e tornare direttamente all'azione se necessario	chiedere nuovamente un dato non valido
Il numero di elaborazioni è noto?	inizializzare un contatore, controllarlo, eseguire il corpo, aggiornarlo e tornare al controllo	acquisire cinque misure

Tabella 4.15: Tre modi di organizzare una ripetizione nei diagrammi di flusso.

4.12.1 Progettare prima di disegnare

Prima di tracciare le forme geometriche è utile compilare una piccola scheda.

1. **Che cosa deve essere ripetuto?** Individuare le istruzioni che formano il corpo.
2. **Quando deve terminare?** Formulare la condizione di uscita o di continuazione.
3. **Quali variabili servono?** Stabilire valori iniziali, contatori, accumulatori o sentinelle.
4. **Che cosa cambia a ogni passaggio?** Scrivere gli aggiornamenti.
5. **Dove deve tornare la freccia?** Individuare il simbolo esatto dal quale ricomincia il

percorso.

Solo dopo queste decisioni conviene disegnare il diagramma.

4.12.2 Verificare la terminazione

Prima di considerare corretta una ripetizione bisogna rispondere a tre domande:

1. quali variabili determinano la risposta del rombo;
2. quali istruzioni modificano tali variabili;
3. gli aggiornamenti avvicinano realmente i valori alla situazione che farà uscire dal ciclo?

Se la condizione dipende da $i \leq 5$, il contatore deve prima o poi diventare maggiore di 5. Se dipende dall'inserimento della sentinella, deve esistere un nuovo input che possa modificare il dato controllato.

4.12.3 Usare casi di prova

Una verifica efficace deve comprendere situazioni differenti:

- nessuna iterazione, quando il diagramma lo permette;
- una sola iterazione;
- più iterazioni;
- valori posti esattamente ai confini della condizione;
- dati non validi o sentinelle inserite immediatamente.

Per il diagramma con sentinella è utile provare 0 come primo input, poi una sequenza con un solo valore e infine una sequenza con più valori. Per il diagramma con contatore è utile verificare il primo valore, l'ultimo valore ammesso e il valore che provoca l'uscita.

Approfondimento: tracciare prima di correggere

Quando un diagramma non produce il risultato atteso, conviene costruire una tabella con i valori delle variabili a ogni passaggio. La traccia permette di individuare il punto preciso nel quale il percorso o un aggiornamento diventano errati.

Esercizi: progettare e verificare iterazioni

1. Per ciascuna situazione, indicare dove dovrebbe trovarsi il controllo e motivare la risposta:
 - a) acquisire valori fino all'inserimento di 0;
 - b) ripetere la richiesta di una percentuale fino all'inserimento di un valore valido;
 - c) elaborare le misure di 12 campioni;

Esercizi: progettare e verificare iterazioni (continua)

- d) estrarre numeri casuali finché compare il valore 10.
- 2. Individuare corpo, condizione, inizializzazione e aggiornamento in un diagramma che comunica i numeri da 1 a 20.
- 3. Spiegare perché una freccia di ritorno che conduce direttamente all'aggiornamento, saltando la condizione, può produrre un errore.
- 4. Preparare casi di prova per un diagramma che acquisisce voti fino all'inserimento di 0 e calcola la somma.
- 5. Disegnare un diagramma che acquisisca dati fino alla sentinella 999, esclusa dai dati validi, e comunichi il valore massimo. Inizializzare il massimo con il primo dato valido; se 999 è il primo input, comunicare "Nessun dato".
- 6. Disegnare un diagramma che chieda un numero positivo finché l'utente non inserisce un valore corretto.
- 7. Disegnare un diagramma che acquisisca dieci valori e conti quanti sono pari e quanti dispari.
- 8. Individuare e correggere un ciclo infinito nel quale i parte da 1, la condizione è $i \leq 10$ e l'aggiornamento è $i = i - 1$.

4.13 I numeri pseudocasuali

In alcuni algoritmi non è possibile, o non è utile, conoscere in anticipo il valore che verrà utilizzato. Un videogioco può scegliere casualmente la posizione di un oggetto, una simulazione può produrre un valore diverso a ogni esecuzione e un programma può estrarre uno dei campioni disponibili.

I computer, tuttavia, eseguono istruzioni precise e deterministiche. Per questo motivo, nella maggior parte dei casi, non generano valori realmente casuali: utilizzano un procedimento matematico che produce una sequenza di valori dall'aspetto casuale.

Definizione: numero pseudocasuale

Un **numero pseudocasuale** è un valore prodotto da un algoritmo. La sequenza sembra casuale, ma dipende dal procedimento utilizzato e da un valore iniziale chiamato **seed**.

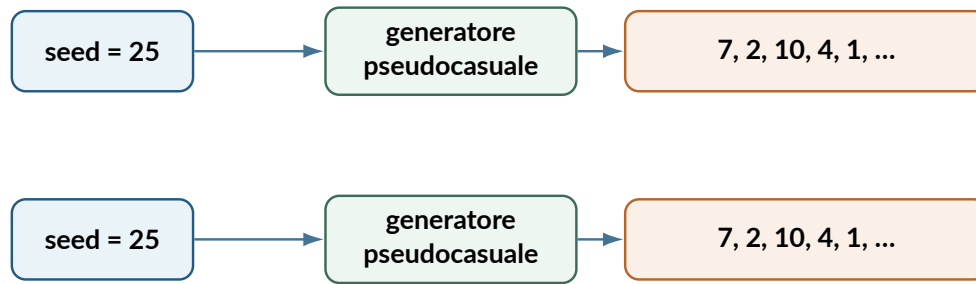
Il prefisso *pseudo* indica che la casualità è simulata. I valori sono sufficientemente imprevedibili per molti esercizi, giochi e simulazioni, ma non devono essere considerati automaticamente adatti ad applicazioni di sicurezza o crittografia.

4.13.1 Il seed: il punto di partenza della sequenza

Il termine inglese **seed** significa *seme*. È il valore iniziale dal quale il generatore costruisce la sequenza pseudocasuale.

Se si utilizza lo stesso generatore con lo stesso seed, si ottiene nuovamente la stessa

successione di valori. Questa caratteristica è utile durante le prove di un programma, perché permette di ripetere esattamente una simulazione e controllare eventuali errori.



A parità di generatore, lo stesso seed produce la stessa sequenza.

Figura 4.21: Lo stesso seed produce la stessa sequenza pseudocasuale. Cambiando il seed si ottiene normalmente una sequenza differente.

Nei programmi reali il seed può essere ricavato, per esempio, dall'orario corrente o da un valore che cambia tra un'esecuzione e l'altra. In questa parte del capitolo è sufficiente comprendere il suo ruolo; le istruzioni specifiche per impostarlo verranno riprese nell'introduzione ad Arduino.

Approfondimento: perché una sequenza ripetibile può essere utile

Durante lo sviluppo di una simulazione, una sequenza sempre diversa potrebbe rendere difficile riprodurre un errore. Impostando un seed noto, lo stesso esperimento può essere eseguito più volte con gli stessi valori. Dopo aver verificato il programma, il seed potrà essere reso variabile.

4.13.2 La funzione `rand()`

Negli algoritmi di questo capitolo la funzione:

`rand()`

rappresenta un generatore che restituisce un numero intero pseudocasuale non negativo. Il valore prodotto può essere molto grande e non appartiene necessariamente all'intervallo richiesto dal problema.

Per limitare i possibili risultati si utilizza l'operatore modulo studiato nella sezione precedente. L'espressione:

`rand() % 10`

può produrre soltanto i resti della divisione per 10, cioè i valori da 0 a 9.

Errore comune: il valore massimo non è 10

L'espressione `rand() % 10` non genera valori da 1 a 10. I possibili resti sono dieci, ma partono da 0: 0, 1, 2, ..., 9.

4.13.3 Generare un numero da 1 a 10

Per spostare l'intervallo da 0–9 a 1–10 si aggiunge 1:

$$a = \text{rand}() \% 10 + 1$$

Il procedimento può essere letto in tre passaggi:

1. `rand()` produce un numero intero pseudocasuale;
2. `% 10` limita il risultato a uno dei dieci valori da 0 a 9;
3. `+ 1` sposta tutti i valori di una posizione, ottenendo l'intervallo da 1 a 10.

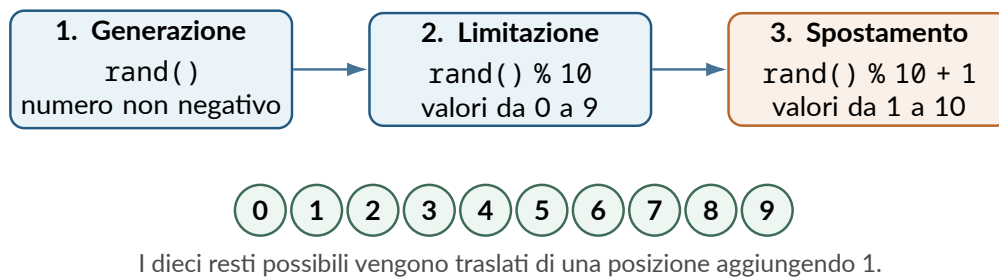


Figura 4.22: Costruzione di un numero pseudocasuale compreso tra 1 e 10.

Il diagramma completo è sequenziale: genera il valore, lo memorizza nella variabile `a` e lo comunica.

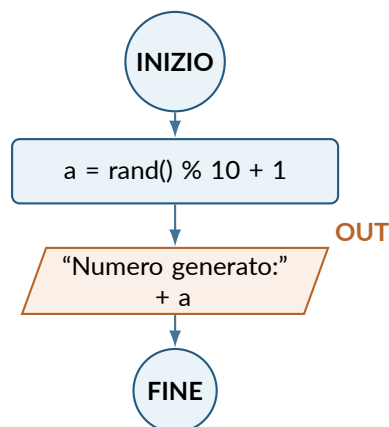


Figura 4.23: Generazione e output di un valore pseudocasuale compreso tra 1 e 10.

4.13.4 La formula generale per un intervallo

Per generare un numero intero compreso tra un valore minimo e un valore massimo, entrambi inclusi, si può utilizzare la formula:

$$\text{numero} = \text{rand}() \% (\text{max} - \text{min} + 1) + \text{min}$$

L'espressione `max - min + 1` calcola quanti valori interi sono presenti nell'intervallo. Il termine `+ min` sposta poi i resti fino al limite inferiore desiderato.

Esempio: generare un numero da 5 a 12

Nell'intervallo da 5 a 12 sono presenti otto valori: 5, 6, 7, 8, 9, 10, 11 e 12. Si scrive quindi:

$$\text{numero} = \text{rand}() \% 8 + 5$$

Il modulo produce valori da 0 a 7; aggiungendo 5 si ottengono valori da 5 a 12.

Intervallo	Espressione	Controllo dei valori possibili
da 0 a 5	$\text{rand}() \% 6$	sei valori: da 0 a 5
da 1 a 6	$\text{rand}() \% 6 + 1$	sei valori spostati di una posizione
da 10 a 20	$\text{rand}() \% 11 + 10$	undici valori: da 10 a 20
da 20 a 50	$\text{rand}() \% 31 + 20$	trentuno valori: da 20 a 50
da 0 a 100	$\text{rand}() \% 101$	centouno valori: da 0 a 100

Tabella 4.16: Esempi di intervalli pseudocasuali con estremi inclusi.

Errore comune: contare soltanto la differenza

Per includere entrambi gli estremi bisogna aggiungere 1. Nell'intervallo da 20 a 50 i valori possibili non sono 30, ma 31. Per questo si utilizza $\% 31$.

4.13.5 Verificare l'intervallo prima di disegnare

Prima di costruire un diagramma è utile controllare sempre il valore minimo e il valore massimo prodotti dall'espressione.

Per esempio, con:

$$x = \text{rand}() \% 7 + 3$$

si ha:

- resto minimo: 0, quindi valore minimo $0 + 3 = 3$;
- resto massimo: 6, quindi valore massimo $6 + 3 = 9$.

L'espressione genera dunque numeri compresi tra 3 e 9.

Esercizi: numeri pseudocasuali

1. Indicare i valori minimo e massimo prodotti da: $\text{rand}() \% 8$, $\text{rand}() \% 8 + 1$, $\text{rand}() \% 20 + 5$.
2. Scrivere l'espressione che genera un numero intero da 1 a 100.
3. Scrivere l'espressione che genera un numero intero da 0 a 20.
4. Scrivere l'espressione che genera un numero intero da 15 a 30.

Esercizi: numeri pseudocasuali (continua)

5. Scrivere l'espressione che genera un numero intero da 50 a 75.
6. Spiegare perché `rand() % 6` non simula correttamente un dado numerato da 1 a 6.
7. Correggere l'espressione `x = rand() % 10` affinché produca valori da 1 a 10.
8. Individuare l'errore nell'espressione proposta per generare valori da 20 a 50: `x = rand() % 30 + 20`.

4.14 Numeri casuali e decisioni

La generazione pseudocasuale diventa particolarmente utile quando viene combinata con le decisioni. L'algoritmo può produrre un valore non noto in anticipo, verificarne le caratteristiche e scegliere un percorso differente in base al risultato.

La struttura generale è:

generazione del valore → condizione → scelta del ramo

Gli esempi seguenti riutilizzano operatori di confronto, operatori logici e modulo. I valori generati simulano eventi o misurazioni: non provengono ancora da un dispositivo reale.

4.14.1 Simulare il lancio di un dado

Un dado tradizionale può produrre i sei valori da 1 a 6. L'istruzione necessaria è:

`dado = rand() % 6 + 1`

Dopo aver comunicato il valore ottenuto, l'algoritmo può verificare se è stato generato il valore massimo.

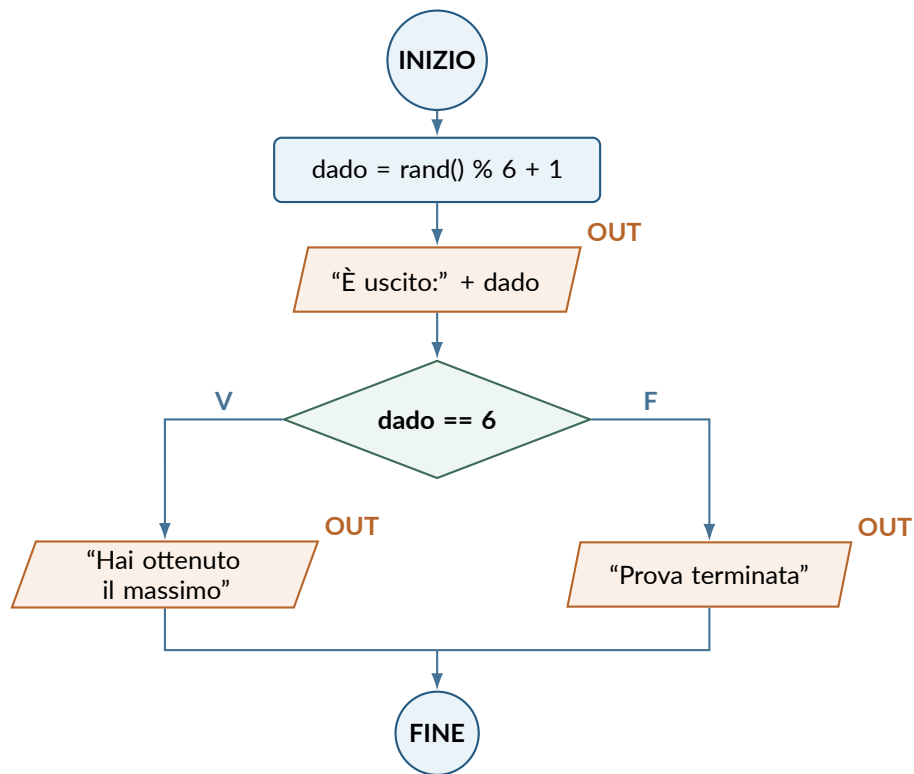


Figura 4.24: Simulazione del lancio di un dado e controllo dell'uscita 6.

Esempio: due esecuzioni possibili

Se viene generato `dado = 6`, la condizione `dado == 6` è vera e viene seguito il ramo V. Se viene generato `dado = 3`, la condizione è falsa e viene seguito il ramo F.

4.14.2 Simulare il lancio di una moneta

Per rappresentare due soli risultati si può generare un valore 0 oppure 1:

```
moneta = rand() % 2
```

Una possibile convenzione è:

- 0 rappresenta testa;
- 1 rappresenta croce.

Il rombo potrà contenere la condizione `moneta == 0`; il ramo V comunicherà "Testa", mentre il ramo F comunicherà "Croce".

Errore comune: la convenzione deve essere dichiarata

I valori 0 e 1 non significano automaticamente testa e croce. È l'algoritmo a stabilire la corrispondenza. La convenzione scelta deve essere dichiarata e poi utilizzata in modo coerente.

4.14.3 Generare un numero e verificarne la parità

Si genera un numero compreso tra 1 e 100:

```
numero = rand() % 100 + 1
```

Successivamente si utilizza il modulo per controllare se il numero è pari:

```
numero % 2 == 0
```

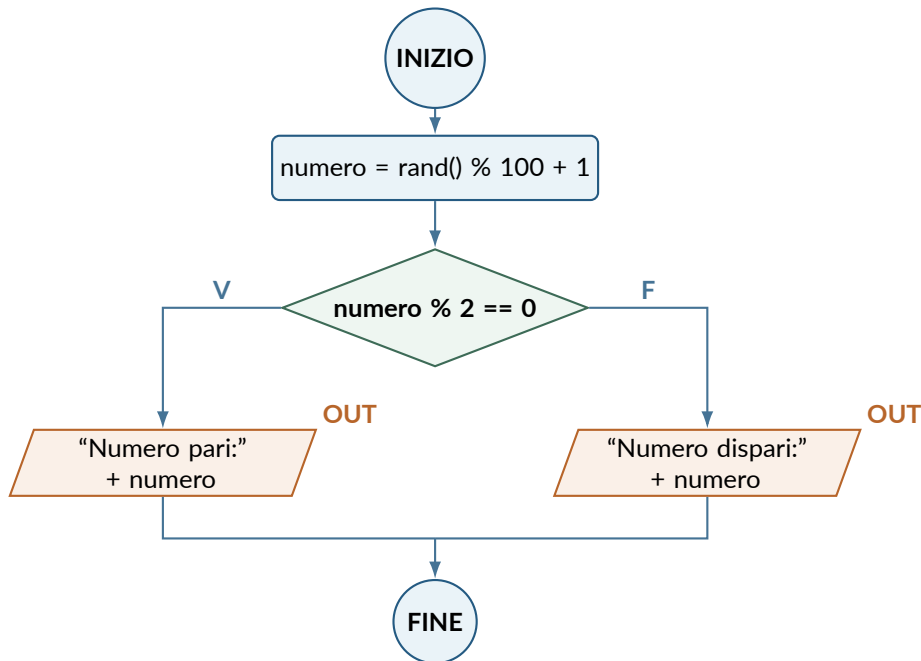


Figura 4.25: Generazione di un numero da 1 a 100 e classificazione come pari o dispari.

Questo esempio mostra che una variabile generata casualmente viene utilizzata esattamente come una variabile acquisita tramite input: può partecipare a calcoli, confronti e output.

4.14.4 Simulare una temperatura

Per simulare una temperatura intera compresa tra 10 e 40 gradi si utilizza:

```
temperatura = rand() % 31 + 10
```

I valori possibili sono 31, perché si contano anche gli estremi. L'algoritmo può attivare un allarme quando il valore supera 30.

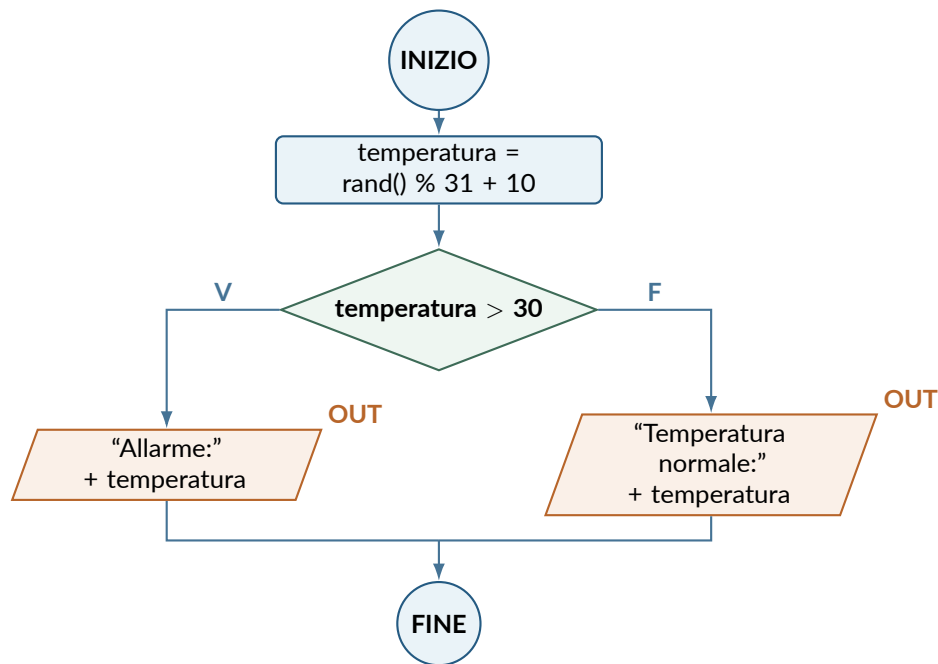


Figura 4.26: Simulazione di una temperatura e decisione basata su una soglia.

Approfondimento: simulazione e misurazione reale

In questo diagramma la temperatura è inventata dal generatore pseudocasuale. In un sistema reale il valore sarebbe acquisito da un sensore attraverso un'operazione di input. La parte decisionale dell'algoritmo, invece, potrebbe rimanere quasi identica.

4.14.5 Simulare l'umidità del terreno

Una percentuale di umidità compresa tra 0 e 100 può essere simulata con:

```
umidita = rand() % 101
```

Il valore 101 dopo il modulo è necessario perché i resti possibili devono andare da 0 a 100. Si può poi stabilire che l'irrigazione sia necessaria quando l'umidità è inferiore a 30.

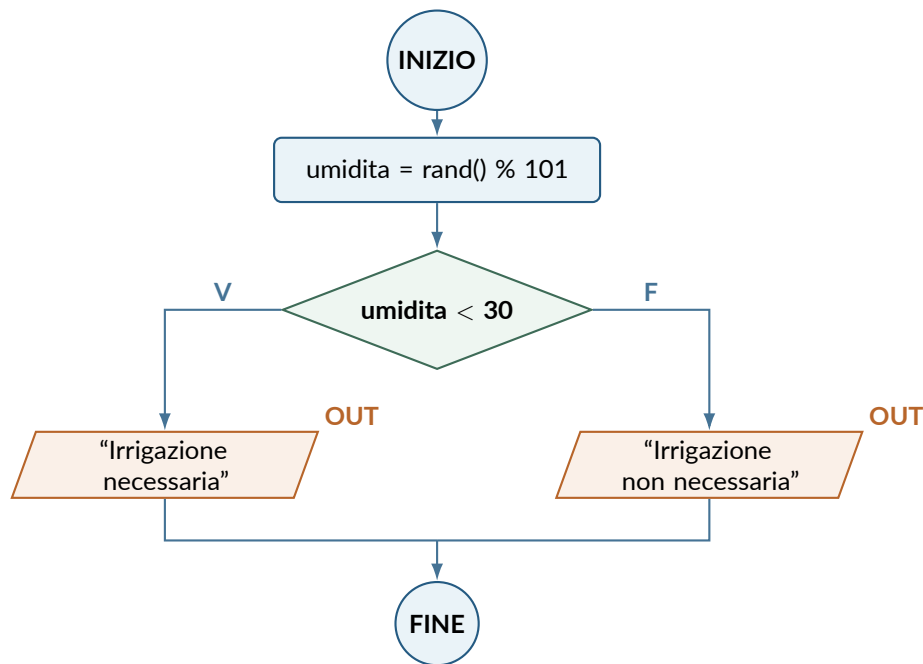


Figura 4.27: Simulazione dell'umidità del terreno e decisione sull'irrigazione.

L'esempio è una semplificazione didattica. In un impianto reale sarebbero necessari un sensore correttamente calibrato, controlli di sicurezza e un dispositivo capace di comandare l'irrigazione.

4.14.6 Simulare un valore di pH

Per un primo esercizio con sole variabili intere si può generare un valore compreso tra 0 e 14:

```
pH = rand() % 15
```

Si può poi verificare se il valore appartiene a un intervallo considerato accettabile, per esempio:

```
(pH >= 6) && (pH <= 8)
```

Il ramo V comunica che il valore appartiene all'intervallo, mentre il ramo F segnala che si trova al di fuori. Nelle misure reali il pH può avere una parte decimale; l'esempio utilizza numeri interi soltanto per esercitare la logica delle condizioni composte.

4.14.7 Scegliere correttamente l'intervallo e la condizione

Negli algoritmi che combinano casualità e decisioni si devono controllare due aspetti distinti:

1. l'espressione con `rand()` deve produrre esattamente l'intervallo desiderato;
2. la condizione nel rombo deve classificare correttamente tutti i valori possibili, compresi quelli uguali alle soglie.

Per esempio, se una temperatura pari a 30 deve essere considerata normale, la condizione di allarme sarà `temperatura > 30`. Se anche 30 deve attivare l'allarme, sarà invece necessario scrivere `temperatura >= 30`.

Esercizi: numeri casuali e decisioni

1. Disegnare un diagramma che simuli il lancio di una moneta e comunichi "Testa" oppure "Croce".
2. Generare un numero da 1 a 20 e comunicare se è maggiore di 10.
3. Generare un numero da 1 a 100 e comunicare se è multiplo di 5.
4. Simulare il lancio di due dadi, memorizzare i risultati in `dado1` e `dado2`, calcolare la somma e comunicarla.
5. Modificare l'esercizio precedente in modo da comunicare "Doppio" quando i due dadi hanno lo stesso valore.
6. Generare un voto da 1 a 10 e comunicare "Sufficiente" quando il voto è almeno 6, altrimenti "Insufficiente".
7. Generare una temperatura da -5 a 45. Scrivere l'espressione corretta e comunicare un allarme quando è inferiore a 0 oppure superiore a 40.
8. Generare un'umidità da 0 a 100 e comunicare "Intervallo ottimale" quando è compresa tra 40 e 70, estremi inclusi.
9. Generare un numero da 1 a 50 e verificare se è contemporaneamente multiplo di 3 e di 5.
10. Generare casualmente il numero di uno dei 12 campioni di laboratorio e comunicare "Controllo aggiuntivo" quando il numero è multiplo di 4.
11. Individuare l'errore nell'algoritmo che vuole generare una percentuale da 0 a 100 utilizzando `valore = rand() % 100`.
12. Preparare almeno quattro casi di prova teorici per verificare un algoritmo che considera accettabile un pH compreso tra 6 e 8.
13. Applicazione facoltativa: simulare una temperatura tra 10 e 40 e un'umidità tra 0 e 100; attivare l'irrigazione quando la temperatura è superiore a 25 e l'umidità è inferiore a 30.

Esercizi riepilogativi

Algoritmi e programmazione

Capitolo 4

1. Riconosci gli elementi dell'algoritmo

Associa ogni descrizione al termine corretto: **input**, **output**, **variabile**, **condizione**, **iterazione**, **contatore**, **accumulatore**, **sentinella**.

1. Singola esecuzione del corpo di un ciclo:
2. Valore speciale che segnala la fine dei dati:
3. Variabile che registra il numero di ripetizioni:
4. Variabile che raccoglie progressivamente un totale:
5. Espressione che può risultare vera o falsa:
6. Dato acquisito dall'algoritmo:
7. Risultato comunicato all'esterno:
8. Contenitore identificato da un nome:

2. Segui una ripetizione passo dopo passo

Un diagramma inizializza $\text{somma} = 0$ e $i = 1$. Finché $i \leq 4$, aggiunge i a somma e poi esegue $i = i + 1$. Completa la tabella.

Iterazione	i prima	somma dopo	i dopo
1			
2			
3			
4			

3. Organizza la ripetizione

Per ogni situazione indica se conviene: controllare un dato prima di elaborarlo, eseguire almeno una volta l'azione e poi controllare, oppure utilizzare un contatore.

1. Acquisire esattamente 10 misure:
2. Ripetere la richiesta di un voto fino all'inserimento di un valore compreso tra 1 e 10:
3. Sommare valori fino all'inserimento di 0, escludendo la sentinella:
4. Comunicare i numeri da 20 a 1:

4. Trova e correggi gli errori

1. Media di tre valori: $\text{media} = a + b + c / 3$
2. Numero compreso tra 1 e 10: $(x \geq 1) \mid (x \leq 10)$
3. Conteggio da 1 a 5: $i = 1$, controllo $i \leq 5$, output di i , ma nessun aggiornamento.
4. Conteggio decrescente: $i = 5$, controllo $i \geq 1$, aggiornamento $i = i + 1$.

Esercizi riepilogativi

Dalla consegna al diagramma di flusso

Capitolo 4**1. Algoritmo sequenziale: densità di un campione**

Acquisisci la massa in grammi e un volume maggiore di zero in millilitri, calcola $\text{densita} = \text{massa} / \text{volume}$ e comunica il risultato. Prima di disegnare, individua input, elaborazione e output.

Spazio per il diagramma di flusso

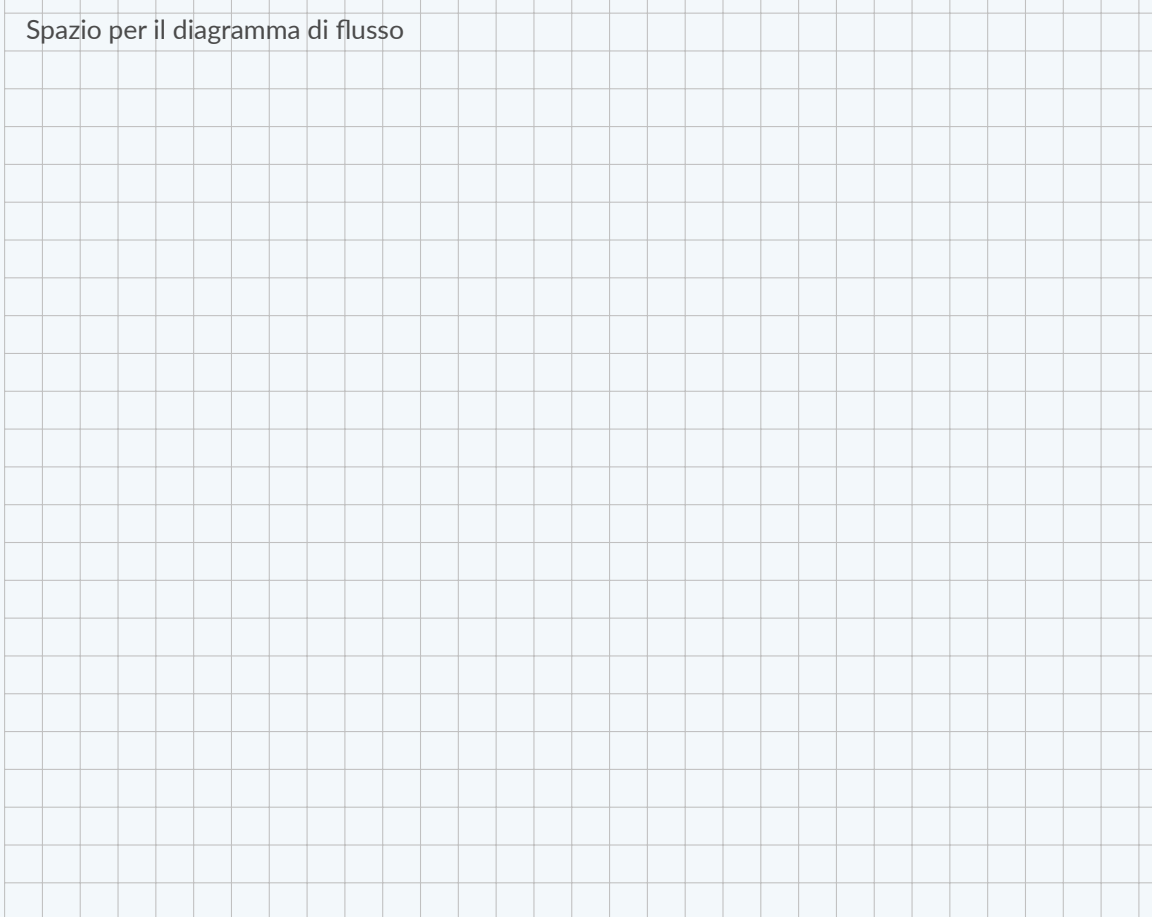
Esercizi riepilogativi**Capitolo 4**

Dalla consegna al diagramma di flusso

2. Dati fino a una sentinella

Acquisisci numeri interi fino all'inserimento di 0. La sentinella non deve essere elaborata. Calcola quanti valori sono stati inseriti e quanti di essi sono pari, quindi comunica entrambi i risultati. Evidenzia nel diagramma l'inizializzazione, la condizione, il corpo e l'aggiornamento degli accumulatori.

Spazio per il diagramma di flusso



Esercizi riepilogativi

Dalla consegna al diagramma di flusso

Capitolo 4**3. Sei misure con contatore e decisione**

Acquisisci le misure di 6 campioni. Per ogni valore, controlla se è maggiore di 50 e aggiorna un contatore. Al termine comunica quante misure superano la soglia. Nel diagramma separa chiaramente inizializzazione, condizione, corpo e aggiornamento del contatore.

Spazio per il diagramma di flusso



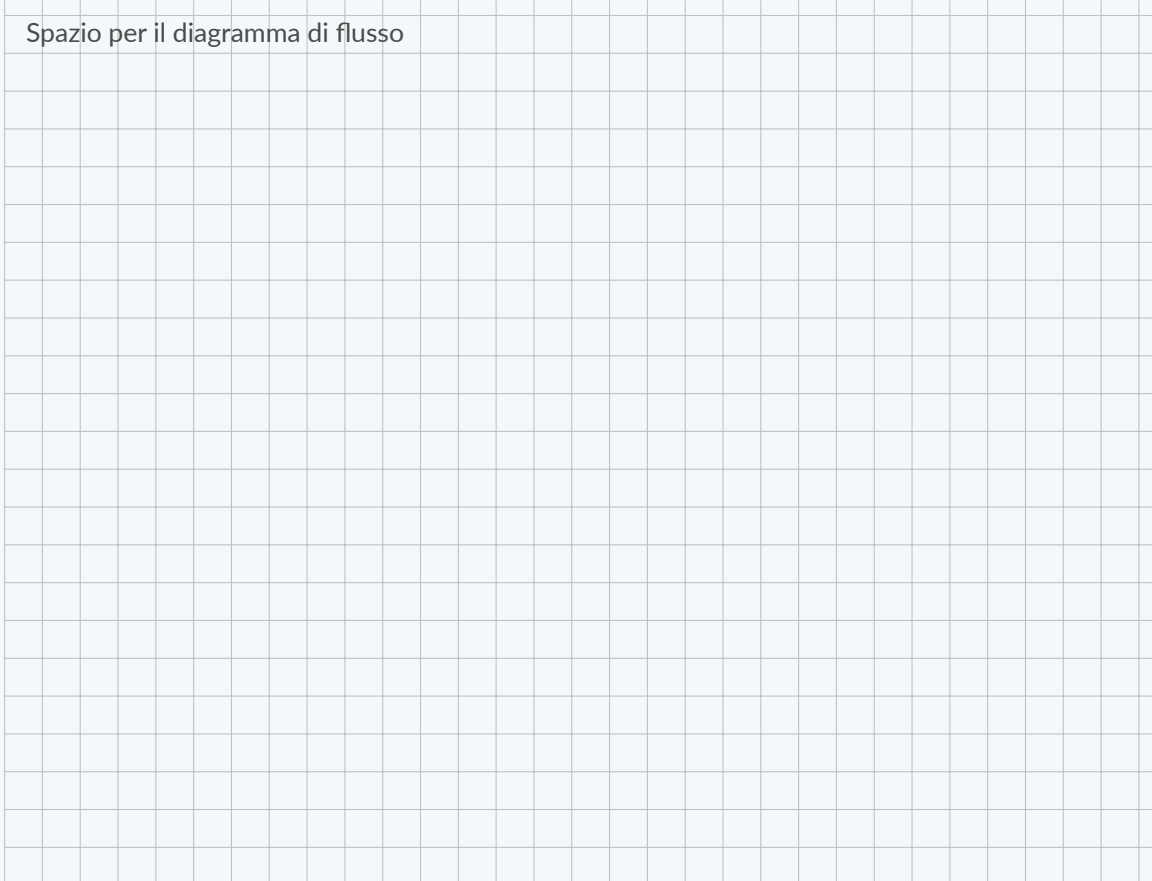
Esercizi riepilogativi**Capitolo 4**

Dalla consegna al diagramma di flusso

4. Casualità e decisione composta nella ripetizione

Ripeti 5 volte la simulazione di una temperatura intera da 10 a 40 e di un'umidità intera da 0 a 100. Conta quante simulazioni richiedono irrigazione, considerando necessaria l'irrigazione quando la temperatura è superiore a 25 e l'umidità è inferiore a 30. Comunica il contatore finale e scrivi le due espressioni con `rand()`.

Spazio per il diagramma di flusso



Parte V

Appendici operative e cittadinanza digitale

A. Arduino e il controllo del mondo fisico

Un algoritmo può elaborare numeri e prendere decisioni, ma può anche interagire con l'ambiente. Collegando sensori e attuatori a una scheda programmabile, le istruzioni studiate nei diagrammi di flusso diventano parte di un sistema capace di osservare un fenomeno e produrre un effetto reale. Arduino permette di affrontare questa trasformazione in modo graduale, unendo programmazione, elettronica e sperimentazione.

In questa appendice si farà riferimento a una scheda **Arduino Uno**. Lo scopo non è studiare in modo completo il linguaggio C++ o l'elettronica, ma riconoscere nel codice gli stessi elementi già incontrati nel Capitolo 4: variabili, assegnamenti, input, elaborazioni, output e decisioni.

Conoscenze	Abilità	Competenze
• microcontrollore e scheda Arduino; • sensori, ingressi, uscite e attuatori; • elementi principali di Arduino Uno; • ambiente di sviluppo e caricamento di un programma; • struttura <code>setup()</code> e <code>loop()</code>; • variabili e tipi fondamentali.	• distinguere sensori e attuatori; • riconoscere le parti essenziali della scheda; • seguire una procedura corretta di compilazione e caricamento; • interpretare la struttura di uno sketch; • dichiarare e inizializzare semplici variabili; • collegare diagrammi di flusso e codice.	• comprendere il funzionamento essenziale di un sistema automatico; • progettare semplici soluzioni che collegano software e mondo fisico; • utilizzare componenti e strumenti rispettando le regole di sicurezza; • applicare la programmazione a problemi di ambito agricolo, ambientale e chimico.

A.1 Dal diagramma di flusso al mondo fisico

Nel Capitolo 4 un algoritmo riceveva dati, eseguiva calcoli o confronti e produceva un risultato. I dati venivano spesso immaginati come numeri forniti dall'utente e l'output come un valore o una frase mostrati sullo schermo.

In un sistema elettronico programmabile lo stesso schema può essere applicato a fenomeni reali. Un valore può provenire da un pulsante o da un sensore; il risultato dell'elaborazione può accendere una spia, produrre un suono oppure comandare un altro dispositivo.

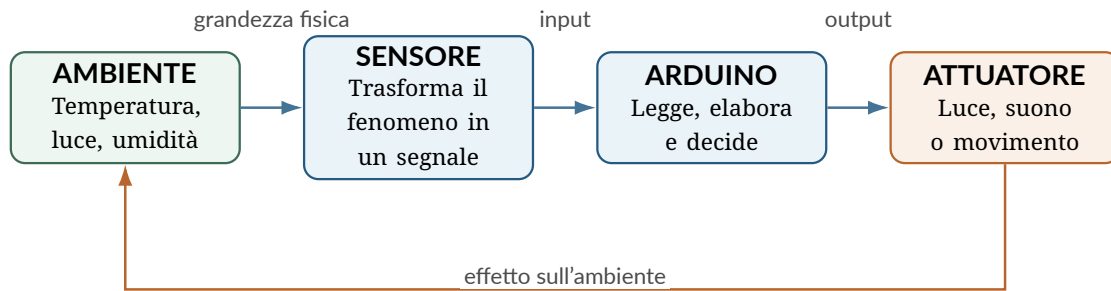


Figura A.1: Il collegamento tra ambiente, sensore, Arduino e attuatore.

A.1.1 Osservare, elaborare e agire

Un sistema automatico elementare può essere descritto attraverso quattro fasi.

Acquisizione

Un componente osserva una grandezza o uno stato dell'ambiente. Per esempio, un sensore può rilevare la temperatura, mentre un pulsante può comunicare se è stato premuto.

Input

Il segnale viene ricevuto dalla scheda e trasformato in un valore utilizzabile dal programma.

Elaborazione

Il microcontrollore esegue calcoli e confronti. Può, per esempio, verificare se la temperatura supera una soglia.

Output e azione

La scheda invia un comando a un dispositivo esterno, come un LED, un buzzer o un relè. Il dispositivo produce l'effetto previsto.

Definizione: sensore

Un **sensore** è un componente capace di rilevare una grandezza fisica o uno stato e di trasformarlo in un segnale utilizzabile da un sistema elettronico.

Definizione: attuatore

Un **attuatore** è un dispositivo che riceve un comando e produce un effetto fisico, come luce, movimento, calore o suono.

Sensore e input non sono sinonimi. Il sensore è il componente fisico che rileva il fenomeno; l'input è il dato che il programma riceve. Allo stesso modo, l'output è il comando prodotto dal programma, mentre l'attuatore è il dispositivo che lo trasforma in un'azione.

Situazione	Input ed elaborazione	Output e attuatore
Pulsante di comando	Verificare se il pulsante è premuto	Accendere o spegnere un LED
Serra	Confrontare la temperatura con una soglia	Attivare una segnalazione
Terreno coltivato	Confrontare l'umidità con un valore minimo	Segnalare che il terreno è asciutto
Laboratorio	Verificare se un valore simulato appartiene a un intervallo	Mostrare un messaggio o attivare una spia

Tabella A.1: Esempi di sistemi descritti mediante input, elaborazione e output.

A.1.2 Dal diagramma al sistema

La logica di un diagramma di flusso non cambia quando il dato proviene dal mondo fisico. Cambia invece il modo in cui vengono realizzati input e output.

Per esempio, il diagramma seguente descrive un controllo di temperatura:

IN temperatura $\rightarrow$ temperatura > 30 $\rightarrow$ OUT allarme

In una simulazione, la temperatura può essere digitata dall'utente o generata casualmente. In un sistema reale, la variabile viene aggiornata leggendo un sensore. Il confronto con 30 rimane invece uguale dal punto di vista logico.

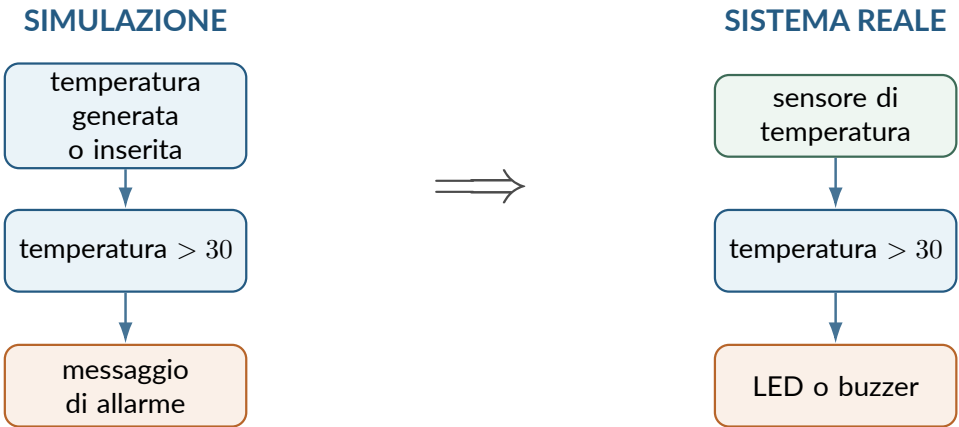


Figura A.2: La stessa struttura logica può essere applicata a dati simulati oppure rilevati da un sensore.

Approfondimento: sistemi a ciclo chiuso

In alcuni sistemi l'azione prodotta modifica l'ambiente e il sensore misura nuovamente il risultato. Un impianto di irrigazione, per esempio, può aumentare l'umidità del terreno; una nuova misura permette di decidere se continuare o interrompere l'irrigazione. Questo meccanismo prende il nome di **retroazione** o *feedback*.

Errore comune: un sensore non comprende il fenomeno

Un sensore non decide se un valore è corretto o pericoloso. Si limita a produrre un segnale. È il programma a interpretare il dato secondo le soglie e le regole stabilite dal progettista.

Esercizi: dal diagramma al mondo fisico

1. Distinguere sensore, input, output e attuatore nell'esempio di una lampada che si accende quando l'ambiente diventa buio.
2. Indicare un possibile sensore e un possibile attuatore per un sistema che segnala una temperatura troppo elevata.
3. Descrivere con il modello input-elaborazione-output un sistema che controlla l'umidità del terreno.
4. Spiegare perché un pulsante può essere considerato un dispositivo di input.
5. Spiegare la differenza tra il comando prodotto dal programma e l'azione svolta dall'attuatore.
6. Individuare input, condizione e output nell'algoritmo: "leggere il valore di pH e accendere una spia quando il valore è inferiore a 6 oppure superiore a 8".
7. Applicazione facoltativa: proporre un semplice sistema automatico utile in una serra, specificando sensore, variabile, condizione e attuatore.

A.2 Che cos'è Arduino

Arduino è una piattaforma per la realizzazione di progetti elettronici programmabili. Comprende schede dotate di microcontrollore, strumenti software e una vasta raccolta di librerie ed esempi.

Una scheda Arduino riceve un programma preparato su un computer. Dopo il caricamento, può eseguire le istruzioni anche senza rimanere collegata al computer, purché disponga di un'alimentazione adeguata.

Definizione: microcontrollore

Un **microcontrollore** è un circuito integrato che contiene un'unità di elaborazione, memoria e collegamenti per comunicare con dispositivi esterni. È progettato per eseguire programmi dedicati al controllo di sistemi elettronici.

A.2.1 Computer e microcontrollore

Un computer e un microcontrollore elaborano istruzioni, ma sono progettati per compiti differenti. Il computer è un sistema generale, capace di eseguire molte applicazioni. Il microcontrollore è normalmente inserito all'interno di un dispositivo e svolge un insieme più limitato di funzioni.



Figura A.3: Computer e microcontrollore condividono alcuni elementi fondamentali, ma sono destinati a impieghi differenti.

Aspetto	Computer	Microcontrollore
Utilizzo	Attività generali e applicazioni differenti	Controllo di un sistema specifico
Software	Esegue normalmente un sistema operativo e più programmi	Esegue direttamente un programma dedicato
Risorse	Memoria e potenza di calcolo elevate	Risorse più limitate ma consumi ridotti
Interazione	Schermo, tastiera, mouse e rete	Sensori, pulsanti, LED, motori e altri componenti
Avvio	Carica il sistema operativo e le applicazioni	Avvia il programma memorizzato nella scheda

Tabella A.2: Confronto generale tra computer e microcontrollore.

A.2.2 La scheda Arduino Uno

Arduino Uno è una scheda didattica molto utilizzata per i primi progetti. Riunisce il microcontrollore, i circuiti necessari per l'alimentazione e la comunicazione con il computer e una serie di pin ai quali possono essere collegati componenti esterni.

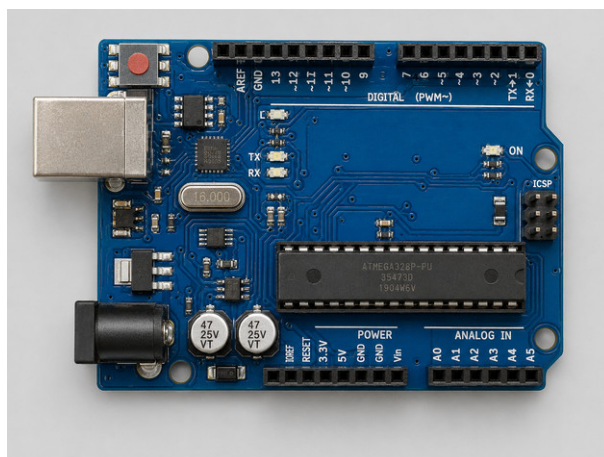


Figura A.4: Vista dall'alto di una scheda Arduino Uno. La disposizione e alcuni dettagli possono variare leggermente tra differenti revisioni o schede compatibili.

Le parti principali da riconoscere sono le seguenti.

Microcontrollore

È il circuito che esegue il programma e coordina gli ingressi e le uscite.

Porta USB

Permette di collegare la scheda al computer, caricare il programma e, durante le esercitazioni, alimentare la scheda.

Alimentazione

La scheda può ricevere energia dalla porta USB oppure, nei casi previsti, da un'alimentazione esterna adeguata.

Pin digitali

Sono collegamenti utilizzati principalmente per segnali con due stati, indicati nel programma come stato alto e stato basso.

Ingressi analogici

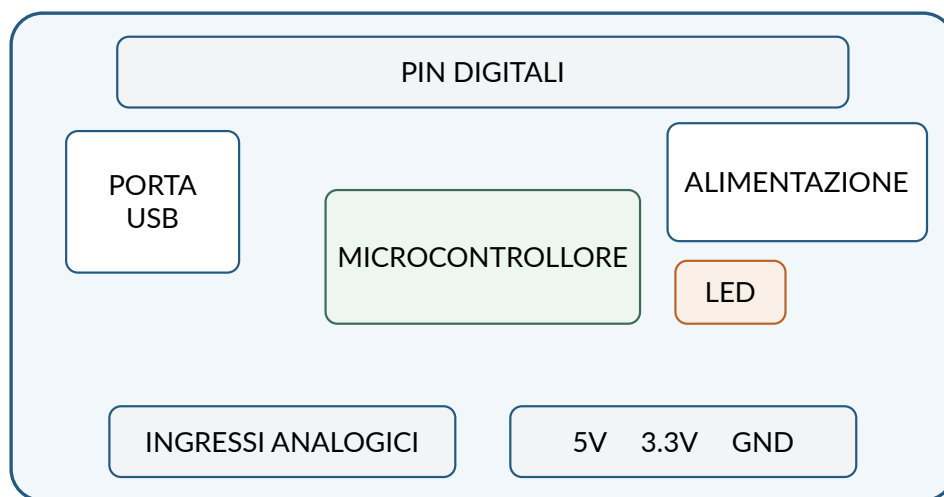
Permettono di acquisire valori variabili provenienti, per esempio, da un potenziometro o da alcuni tipi di sensore.

Pin di alimentazione

I pin indicati con 5V, 3.3V e GND permettono di alimentare semplici circuiti compatibili. GND rappresenta il riferimento comune del circuito.

LED integrati

La scheda possiede piccoli LED di segnalazione. Uno di essi può essere controllato dal programma ed è utile per i primi esperimenti senza componenti esterni.



Schema semplificato: posizione, forma e numero dei componenti non sono rappresentati in scala.

Figura A.5: Schema concettuale degli elementi essenziali presenti su una scheda Arduino Uno.

Curiosità: Arduino nei prototipi

Una scheda Arduino permette di costruire rapidamente un prototipo, cioè una prima versione funzionante di un sistema. Prima di realizzare un dispositivo definitivo è possibile provare sensori, soglie e comportamenti, correggendo il progetto con costi e tempi ridotti.

Errore comune: Arduino non è un computer completo

La scheda non sostituisce un normale computer: non possiede un'interfaccia grafica, un disco per documenti e un sistema operativo destinato all'uso quotidiano. È progettata per eseguire in modo continuo un programma di controllo.

Esercizi: conoscere Arduino

1. Definire il microcontrollore con parole proprie.
2. Indicare due differenze tra un computer e un microcontrollore.
3. Spiegare a che cosa serve la porta USB durante una normale esercitazione.
4. Distinguere pin digitali, ingressi analogici e pin di alimentazione.
5. Indicare quale componente esegue le istruzioni memorizzate nella scheda.
6. Spiegare perché il LED integrato è utile nelle prime prove.
7. Associare ciascun progetto al componente principale: lettura della luce, segnalazione luminosa, rilevamento della pressione di un pulsante, controllo della temperatura.
8. Ricercare sulla scheda utilizzata in laboratorio le etichette 5V, 3.3V, GND, A0 e LED; riportarne la posizione in uno schizzo.

A.3 Componenti e sicurezza

La programmazione di Arduino è strettamente collegata alla costruzione di semplici circuiti. Prima di effettuare i collegamenti è necessario conoscere la funzione essenziale dei componenti e rispettare alcune regole di sicurezza.

A.3.1 La breadboard

La **breadboard** è una basetta che permette di costruire e modificare circuiti senza saldature. I componenti e i cavetti vengono inseriti nei fori; sotto la superficie, gruppi di fori sono già collegati mediante contatti metallici.

Nella parte centrale, i fori sono generalmente organizzati in gruppi di cinque. Le due metà sono separate da una fessura, utile per inserire circuiti integrati senza collegare accidentalmente i terminali opposti. Lungo i bordi possono essere presenti linee destinate alla distribuzione dell'alimentazione.



Figura A.6: Organizzazione tipica dei collegamenti interni di una breadboard. Prima dell'uso è opportuno verificare il modello disponibile in laboratorio.

Errore comune: fori vicini non significa sempre fori collegati

La distanza tra due fori non garantisce che siano elettricamente collegati. È necessario conoscere la disposizione interna della breadboard e controllare le linee di alimentazione, che in alcuni modelli possono essere interrotte a metà.

A.3.2 LED e resistenza

Il **LED** è un componente che emette luce quando viene attraversato da corrente nel verso corretto. Possiede due terminali e deve essere collegato rispettando la polarità.

In molti LED comuni:

- il terminale più lungo corrisponde all'**anodo**, da collegare verso il potenziale positivo;
- il terminale più corto corrisponde al **catodo**, da collegare verso GND;
- il lato piatto dell'involucro aiuta spesso a riconoscere il catodo.

Il LED non deve essere collegato direttamente a un pin senza limitare la corrente. Si inserisce quindi una **resistenza** in serie. Nei primi circuiti alimentati a 5 V si utilizzano frequentemente resistenze dell'ordine di alcune centinaia di ohm, per esempio 220 Ω o 330 Ω , secondo il componente e il progetto.

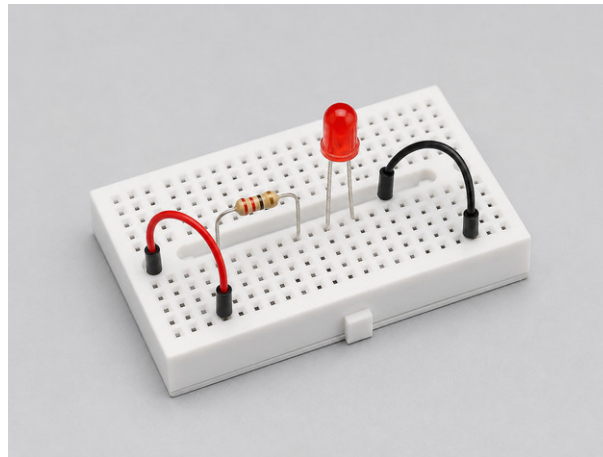


Figura A.7: Esempio di LED collegato in serie a una resistenza su breadboard.

Approfondimento: perché serve la resistenza

La resistenza limita la corrente che attraversa il LED e il pin della scheda. Senza questa protezione la corrente potrebbe diventare eccessiva e danneggiare il componente o il microcontrollore. Il valore corretto può essere determinato applicando la legge di Ohm e consultando le caratteristiche del LED.

A.3.3 Il pulsante

Un pulsante è un semplice dispositivo di input digitale. Quando viene premuto modifica il collegamento elettrico del circuito, permettendo al programma di distinguere due stati.

Un pulsante per breadboard possiede spesso quattro terminali: i due terminali dello stesso lato sono già collegati tra loro, mentre la pressione mette in comunicazione i due lati. Per questo motivo il pulsante deve essere orientato correttamente sulla breadboard.

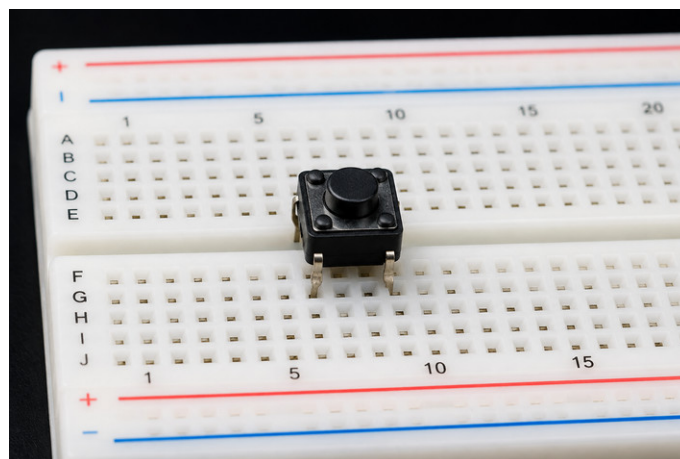


Figura A.8: Esempio di pulsante utilizzabile come ingresso digitale.

A.3.4 Sensori e attuatori

I componenti collegati ad Arduino possono essere classificati in base alla loro funzione.

Categoria	Funzione	Esempi
Sensori	Forniscono informazioni sull'ambiente	Temperatura, luce, umidità, distanza
Comandi manuali	Forniscono uno stato scelto dall'utente	Pulsante, interruttore, potenziometro
Attuatori semplici	Producono un effetto direttamente osservabile	LED, buzzer, piccolo display
Attuatori di potenza	Comandano movimenti o carichi maggiori tramite circuiti adatti	Motore, pompa, elettrovalvola, lampada

Tabella A.3: Classificazione essenziale dei componenti collegabili a una scheda.

Motori, pompe, elettrovalvole e altri carichi non devono essere collegati direttamente ai pin. Richiedono circuiti di pilotaggio, alimentazioni adeguate e protezioni specifiche. Nelle prime esercitazioni l'azione verrà quindi rappresentata da un LED o da un messaggio.

A.3.5 Regole di sicurezza

Errore comune: non modificare il circuito mentre è alimentato

Prima di spostare componenti o cavetti si deve scollegare la scheda dall'alimentazione. Una modifica eseguita a circuito attivo può provocare cortocircuiti e danneggiare la scheda.

Durante le attività si devono rispettare le seguenti regole:

1. lavorare soltanto con le basse tensioni previste dal materiale didattico;
2. controllare il circuito prima di collegare l'alimentazione;
3. evitare collegamenti diretti tra 5V e GND;
4. utilizzare una resistenza con i LED esterni;
5. non superare i limiti elettrici dei pin e dei componenti;
6. non collegare direttamente motori, pompe, relè di potenza o dispositivi alimentati dalla rete elettrica;
7. scollegare l'alimentazione prima di modificare il circuito;
8. chiedere il controllo del docente quando il collegamento non è chiaro.

Approfondimento: acqua, soluzioni e circuiti

Nelle applicazioni agricole e chimiche possono essere presenti acqua, terriccio umido o soluzioni. La parte elettronica deve rimanere separata dai liquidi. I sensori destinati all'immersione devono essere utilizzati secondo le indicazioni del produttore, mentre la scheda e i collegamenti devono restare asciutti.

Esercizi: componenti e sicurezza

1. Spiegare perché la breadboard permette di costruire circuiti senza saldature.
2. Disegnare un gruppo di cinque fori collegati e indicare quali fori non risultano collegati attraverso la fessura centrale.
3. Indicare come si riconoscono normalmente anodo e catodo di un LED.
4. Spiegare la funzione della resistenza collegata in serie al LED.
5. Classificare come sensore, comando manuale o attuatore: pulsante, LED, sensore di luce, buzzer, sensore di temperatura.
6. Spiegare perché una pompa non deve essere collegata direttamente a un pin di Arduino.
7. Individuare l'errore: uno studente sposta un cavetto mentre la scheda è collegata alla porta USB.
8. Preparare una lista di controllo di cinque punti da usare prima di alimentare un circuito.

A.4 L'ambiente di programmazione

Il programma destinato ad Arduino viene normalmente scritto mediante un ambiente di sviluppo. L'**Arduino IDE** mette a disposizione un editor, gli strumenti per controllare il codice e le funzioni necessarie per trasferirlo sulla scheda.

Un programma Arduino viene spesso chiamato **sketch**. Il termine indica il progetto software formato dal codice e dagli eventuali file collegati.

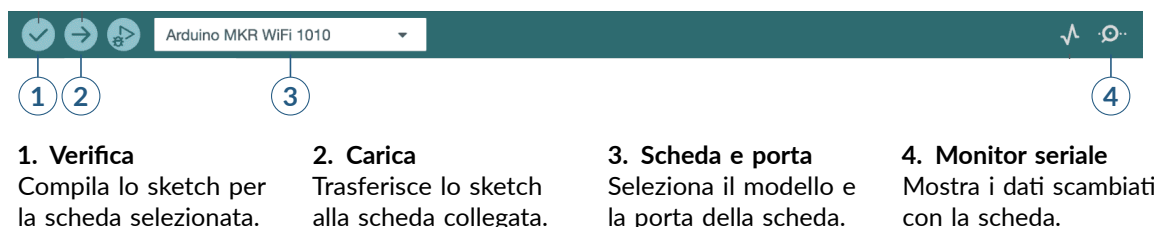


Figura A.9: I comandi principali di Arduino IDE, con richiami in italiano. Il dettaglio della schermata di esempio mostra una MKR WiFi 1010: per le attività di questa appendice selezionare invece Arduino Uno e la sua porta.

A.4.1 Dal codice all'esecuzione

Il lavoro può essere suddiviso in cinque passaggi.

1. **Scrittura:** si inseriscono le istruzioni nell'editor.
2. **Selezione della scheda:** si indica il modello corretto, per esempio Arduino Uno. Questa scelta deve precedere la compilazione.
3. **Verifica:** l'ambiente controlla la sintassi e traduce il codice in istruzioni adatte al modello selezionato.
4. **Caricamento:** dopo aver collegato la scheda e selezionato la relativa porta di comunicazione, il programma compilato viene trasferito nella sua memoria.
5. **Esecuzione:** il microcontrollore avvia il programma e continua a eseguirlo finché rimane alimentato o finché viene caricato un nuovo sketch.

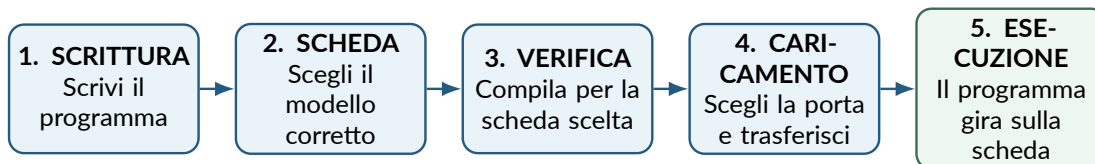


Figura A.10: Dalla scrittura dello sketch alla sua esecuzione sulla scheda.

A.4.2 Verificare non significa collaudare

Il comando di verifica individua gli errori di sintassi che impediscono la compilazione. Non può però garantire che il programma risolva correttamente il problema.

Per esempio, l'istruzione seguente è sintatticamente valida:

```
area = base + altezza;
```

Tuttavia non calcola l'area del rettangolo. L'ambiente di sviluppo può tradurre l'istruzione, ma soltanto il progettista può riconoscere l'errore logico.

Tipo di controllo	Che cosa può individuare	Esempio
Verifica/compilazione	Errori nella forma del codice	Punto e virgola mancante, nome non riconosciuto
Collaudo	Errori nel comportamento del programma	Formula sbagliata, soglia non corretta

Tabella A.4: Differenza tra controllo sintattico e collaudo del funzionamento.

A.4.3 Commenti e leggibilità

I **commenti** sono annotazioni destinate a chi legge il codice. Non vengono trasformati in istruzioni eseguibili.

Codice Arduino: commenti su una riga

```
1 // Calcola la somma di due valori
2 int somma = 5 + 7;
```

Un commento utile spiega lo scopo di un'istruzione o di un gruppo di istruzioni. Non dovrebbe limitarsi a ripetere esattamente ciò che il codice mostra già.

Esempio: commento poco utile e commento utile

Il commento // assegna 30 a soglia ripete il codice `soglia = 30;`. È più informativo scrivere // temperatura massima ammessa nella serra.

A.4.4 Maiuscole, parentesi e punto e virgola

Il linguaggio utilizzato da Arduino distingue le lettere maiuscole dalle minuscole. Per esempio, `setup` e `Setup` sono nomi differenti.

Tre elementi grafici ricorrono frequentemente:

- le parentesi tonde () accompagnano funzioni e condizioni;
- le parentesi graffe { } racchiudono gruppi di istruzioni;
- il punto e virgola ; conclude molte istruzioni.

Errore comune: leggere il primo messaggio significativo

Un errore può produrre numerosi messaggi. Spesso conviene partire dal primo messaggio che indica una riga del proprio sketch: gli errori successivi possono essere soltanto conseguenze del primo.

Esercizi: ambiente di programmazione

1. Ordinare le fasi: caricamento, esecuzione, scrittura, verifica, selezione del modello di scheda. Indicare inoltre quando è necessario selezionare la porta di comunicazione.
2. Spiegare la differenza tra uno sketch e il file eseguibile trasferito sulla scheda.
3. Indicare che cosa controlla la compilazione e che cosa deve essere verificato mediante il collaudo.
4. Stabilire se l'errore `area = base + altezza;` è sintattico oppure logico.
5. Correggere l'istruzione `int valore = 10` aggiungendo il simbolo mancante.
6. Spiegare perché `setup` e `Setup` non sono equivalenti.
7. Scrivere un commento utile per una variabile che contiene il limite minimo di umidità del terreno.
8. Preparare una breve procedura da seguire quando il caricamento non riesce, controllando collegamento, scheda e porta selezionata.

A.5 La struttura di un programma Arduino

Ogni sketch contiene due funzioni fondamentali: `setup()` e `loop()`. Anche quando una delle due non deve svolgere alcuna operazione, la sua struttura viene normalmente mantenuta.

Codice Arduino: struttura minima

```
1 void setup() {  
2   // Istruzioni eseguite una sola volta  
3 }  
4  
5 void loop() {  
6   // Istruzioni ripetute continuamente  
7 }
```

A.5.1 La funzione `setup()`

La funzione `setup()` viene eseguita una sola volta all'avvio della scheda o dopo un riavvio. Contiene le istruzioni necessarie per preparare il sistema.

In seguito verrà utilizzata, per esempio, per:

- stabilire quali pin funzionano come ingressi o uscite;
- avviare la comunicazione con il computer;
- assegnare valori iniziali;
- portare gli attuatori in uno stato sicuro.

A.5.2 La funzione `loop()`

Dopo aver completato `setup()`, Arduino esegue `loop()`. Quando raggiunge l'ultima istruzione, torna automaticamente all'inizio della funzione e la ripete.

Questo comportamento è adatto ai sistemi di controllo, che devono osservare continuamente gli ingressi e aggiornare le uscite. Un sistema può leggere un sensore, prendere una decisione e ripetere il controllo molte volte al secondo.

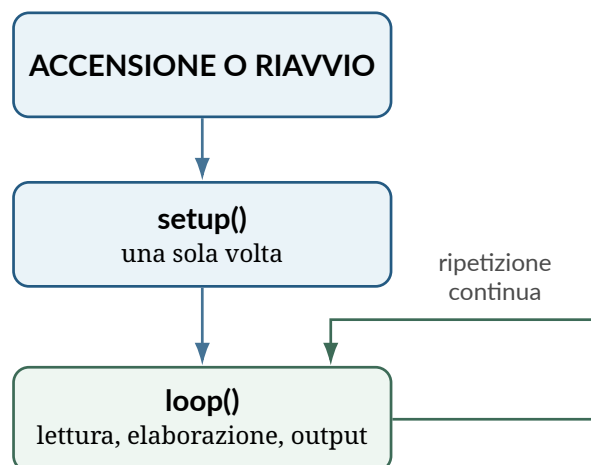


Figura A.11: Dopo l'esecuzione iniziale di `setup()`, la funzione `loop()` viene ripetuta continuamente.

A.5.3 Dove collocare le istruzioni

Le istruzioni che devono essere eseguite una sola volta appartengono generalmente a `setup()`. Le operazioni da ripetere durante il funzionamento appartengono invece a `loop()`.

Operazione	Collocazione tipica	Motivo
Configurare un pin	<code>setup()</code>	La configurazione viene eseguita all'avvio
Avviare la comunicazione	<code>setup()</code>	L'inizializzazione serve una sola volta
Leggere un sensore	<code>loop()</code>	Il valore può cambiare nel tempo
Controllare una soglia	<code>loop()</code>	La decisione deve essere aggiornata
Comandare un LED in base al sensore	<code>loop()</code>	L'uscita deve seguire il valore rilevato

Tabella A.5: Collocazione tipica delle istruzioni nelle due funzioni principali.

A.5.4 Parentesi graffe e blocchi

Le parentesi graffe delimitano il **blocco** di istruzioni appartenente a una funzione. Tutto ciò che si trova tra l'apertura e la chiusura di `setup()` viene eseguito durante la fase iniziale; ciò che si trova nel blocco di `loop()` viene ripetuto.

L'indentazione, cioè lo spostamento verso destra delle istruzioni interne al blocco, non modifica da sola il significato del programma, ma rende visibile la struttura.

Codice Arduino: indentazione leggibile

```
1 void setup() {  
2   int valore = 10;  
3   int doppio = valore * 2;  
4 }  
5  
6 void loop() {  
7 }
```

Errore comune: non confondere la ripetizione con una nuova esecuzione di `setup()`

Quando `loop()` termina, Arduino non ritorna a `setup()`. Riparte dall'inizio di `loop()`. `setup()` viene eseguita nuovamente soltanto dopo un riavvio o una nuova accensione.

Approfondimento: il programma non si ferma alla fine di `loop()`

In un programma tradizionale la fine dell'ultima istruzione può coincidere con la conclusione dell'esecuzione. In Arduino l'ambiente di esecuzione richiama nuovamente `loop()`. La scheda continua quindi a lavorare finché è alimentata.

Esercizi: struttura dello sketch

1. Spiegare quando viene eseguita `setup()`.
2. Spiegare che cosa accade dopo l'ultima istruzione di `loop()`.
3. Indicare in quale funzione inserire: configurazione di un pin, lettura ripetuta di un sensore, inizializzazione della comunicazione, controllo continuo di una soglia.
4. Correggere uno sketch nel quale manca la parentesi graffa finale di `setup()`.
5. Riscrivere con un'indentazione leggibile un blocco di quattro istruzioni.
6. Spiegare perché una lettura del sensore eseguita soltanto in `setup()` non permette di seguire le variazioni successive.
7. Disegnare uno schema che rappresenti l'ordine: accensione, `setup()`, prima esecuzione di `loop()`, seconda esecuzione di `loop()`.

A.6 Variabili e tipi

Nel Capitolo 4 una variabile è stata descritta come un contenitore dotato di nome e valore. Nei diagrammi di flusso sono state utilizzate soltanto variabili numeriche, senza dichiararne il tipo, per concentrare l'attenzione sulla logica.

Nel codice Arduino il programmatore deve invece specificare quale genere di valore il contenitore potrà memorizzare. Questa informazione prende il nome di **tipo**.

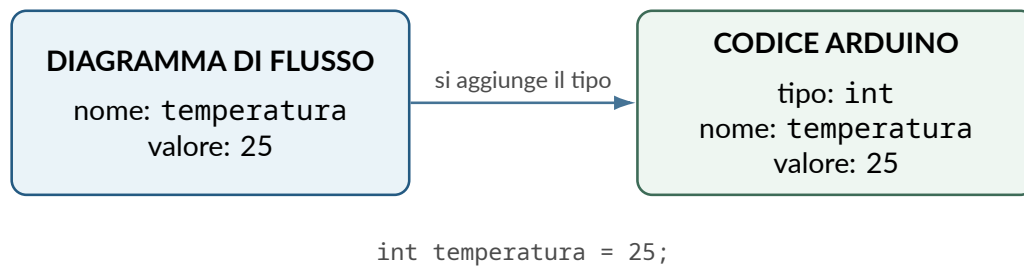


Figura A.12: Nel codice Arduino la variabile possiede un nome, un valore e un tipo dichiarato.

A.6.1 Dichiarare una variabile

La **dichiarazione** comunica al compilatore il tipo e il nome della variabile.

Codice Arduino: dichiarazione

```
1 int temperatura;
```

In questa istruzione:

- `int` è il tipo;
- `temperatura` è il nome;
- il punto e virgola conclude la dichiarazione.

Dopo la dichiarazione è possibile assegnare un valore:

Codice Arduino: assegnamento successivo

```
1 int temperatura;  
2 temperatura = 25;
```

A.6.2 Dichiarazione e inizializzazione

Quando il primo valore viene assegnato nello stesso momento in cui la variabile è dichiarata, si parla di **inizializzazione**.

Codice Arduino: inizializzazione

```
1 int temperatura = 25;
```

La dichiarazione crea il contenitore; l'inizializzazione gli assegna immediatamente un valore iniziale.

Errore comune: una variabile non inizializzata non contiene automaticamente zero

Una variabile locale dichiarata senza valore iniziale non deve essere considerata uguale a zero. Prima di utilizzarla in un calcolo è necessario assegnarle un valore valido.

A.6.3 Il tipo `int`

Il tipo `int` viene utilizzato per i numeri interi, cioè privi di parte decimale.

Codice Arduino: variabili intere

```
1 int numeroCampioni = 12;  
2 int temperatura = 24;  
3 int sogliaUmidita = 30;
```

È adatto, per esempio, a conteggi, numeri di pin, valori letti da molti ingressi e temperature approssimate all'intero.

A.6.4 Il tipo `float`

Il tipo `float` permette di memorizzare numeri con una parte decimale.

Codice Arduino: variabili decimali

```
1 float pH = 6.8;  
2 float temperatura = 23.5;
```

Nel codice il separatore decimale è il punto, anche quando nel testo italiano si utilizza la virgola. Si scrive quindi 6.8, non 6,8.

I valori `float` sono rappresentati con precisione limitata. Sono adatti alle misurazioni e ai calcoli approssimati, ma non devono essere considerati infinitamente precisi.

A.6.5 Il tipo `bool`

Il tipo `bool` può contenere soltanto uno dei due valori logici:

- `true`, cioè vero;
- `false`, cioè falso.

Codice Arduino: variabili logiche

```
1 bool allarme = false;  
2 bool terrenoAsciutto = true;
```

Una variabile logica è utile per rappresentare uno stato. In seguito potrà memorizzare il risultato di una condizione.

A.6.6 Scegliere nomi significativi

Le regole già utilizzate nei diagrammi di flusso valgono anche nel codice:

- il nome deve iniziare con una lettera o con il carattere di sottolineatura;

- non deve contenere spazi;
- deve distinguersi dalle parole riservate del linguaggio;
- maiuscole e minuscole identificano nomi differenti;
- un nome descrittivo rende il programma più leggibile.

Sono preferibili nomi come `temperaturaSerra`, `numeroCampioni` e `sogliaUmidita`. Nomi come `x`, `dato1` o `abc` sono accettabili soltanto quando il significato è evidente dal problema.

Una convenzione frequente consiste nello scrivere la prima parola in minuscolo e iniziare le parole successive con la maiuscola:

`temperaturaMassima` `numeroCampioni` `terrenoAsciutto`

Questa convenzione è chiamata *camel case*.

A.6.7 Assegnamento e confronto

Anche in Arduino è essenziale distinguere l'assegnamento dal confronto.

`temperatura = 25;` assegna il valore 25

`temperatura == 25` verifica se il valore è uguale a 25

Il confronto verrà utilizzato all'interno delle decisioni. L'assegnamento modifica invece il valore contenuto nella variabile.

Tipo	Valori rappresentati	Esempi di utilizzo
int	Numeri interi	Conteggi, soglie intere, numeri di pin
float	Numeri con parte decimale	Temperatura, pH, concentrazioni simulate
bool	true oppure false	Allarme attivo, terreno asciutto, pulsante premuto

Tabella A.6: I tre tipi fondamentali utilizzati nelle prime attività.

Approfondimento: costanti

Un valore che non deve cambiare durante l'esecuzione può essere dichiarato con la parola `const`. Per esempio, `const int soglia = 30;` indica che il valore di `soglia` non dovrebbe essere modificato successivamente. Le costanti rendono più chiaro il significato dei valori fissi e riducono gli errori accidentali.

Esempio: variabili per un controllo ambientale

Un semplice controllo può utilizzare:

Codice Arduino

```
1 float temperatura = 28.5;  
2 int soglia = 30;  
3 bool allarme = false;
```

La temperatura è decimale, la soglia è intera e lo stato dell'allarme è logico.

Esercizi: variabili e tipi

1. Individuare tipo, nome e valore iniziale nell'istruzione `int campioni = 8;`.
2. Dichiarare una variabile intera chiamata `numeroPianta` senza inicializzarla.
3. Dichiarare e inizializzare una variabile `temperatura` con il valore decimale 22,5.
4. Dichiarare una variabile logica `allarme` inizializzata a falso.
5. Scegliere il tipo più adatto per: numero di campioni, valore di pH, stato di un LED, temperatura con decimali.
6. Correggere l'istruzione `float pH = 6,5;`.
7. Spiegare la differenza tra dichiarazione, inizializzazione e assegnamento successivo.
8. Stabilire la differenza tra `umidita = 30;` e `umidita == 30`.
9. Trasformare i nomi poco significativi `x`, `v1` e `ok` in nomi adatti a un programma che controlla l'umidità del terreno.
10. Scrivere tre dichiarazioni per rappresentare la temperatura di una serra, il numero di finestre aperte e lo stato di un allarme.
11. Dichiarare come costante una soglia di temperatura pari a 35.
12. Individuare l'errore logico: una variabile locale viene utilizzata in una somma immediatamente dopo la dichiarazione, senza aver ricevuto alcun valore.

A.7 Comunicare con il monitor seriale

Quando Arduino è collegato al computer tramite la porta USB, il programma può inviare dati testuali al **monitor seriale**. Questo strumento è molto utile nelle prime esercitazioni perché permette di osservare il contenuto delle variabili, verificare le condizioni e descrivere ciò che il programma sta facendo.

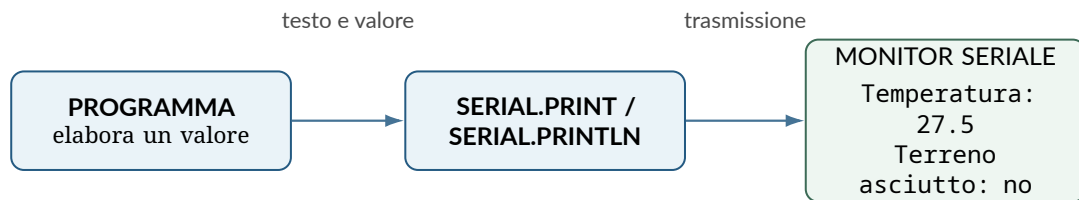


Figura A.13: Il programma può inviare testo e valori al monitor seriale per mostrare risultati e stati del sistema.

A.7.1 Avviare la comunicazione

Prima di inviare messaggi al computer è necessario inizializzare la comunicazione seriale, di solito nella funzione `setup()`.

Codice Arduino: inizializzazione della seriale

```
1 void setup() {  
2   Serial.begin(9600);  
3 }  
4  
5 void loop() {  
6 }
```

Il numero 9600 indica la velocità di comunicazione. Nelle prime attività può essere considerato un valore standard già pronto all'uso.

A.7.2 Stampare testo e valori

Le istruzioni principali per inviare informazioni al monitor seriale sono:

- `Serial.print()` scrive senza andare a capo;
- `Serial.println()` scrive e poi passa alla riga successiva.

Codice Arduino: stampare una variabile

```
1 int temperatura = 27;  
2  
3 void setup() {  
4   Serial.begin(9600);  
5   Serial.println(temperatura);  
6 }  
7  
8 void loop() {  
9 }
```

È possibile stampare anche una parola o una frase scrivendola tra virgolette.

Codice Arduino: stampare una frase

```
1 void setup() {  
2   Serial.begin(9600);  
3   Serial.println("Controllo avviato");  
4 }  
5  
6 void loop() {  
7 }
```

A.7.3 Testo e variabili nello stesso output

Nei diagrammi di flusso si è già utilizzata una forma del tipo:

OUT "Ecco il risultato: " + a

Nel codice Arduino lo stesso effetto si ottiene normalmente con due istruzioni consecutive, una per il testo e una per il valore.

Codice Arduino: testo e variabile

```
1 int a = 12;  
2  
3 void setup() {  
4   Serial.begin(9600);  
5   Serial.print("Ecco il risultato: ");  
6   Serial.println(a);  
7 }  
8  
9 void loop() {  
10 }
```

Si può quindi pensare che il programma stia **concatenando** un messaggio testuale e il contenuto di una variabile. In questo contesto il simbolo + usato nei diagrammi non rappresenta una somma numerica, ma l'idea di unire parti diverse nello stesso messaggio finale.

Approfondimento: perché usare il monitor seriale

Il monitor seriale è utile non soltanto per mostrare il risultato finale, ma anche per osservare valori intermedi durante il collaudo. Per esempio, si può stampare ripetutamente il valore letto da un sensore per capire se il collegamento è corretto e se la soglia scelta è adeguata.

Errore comune: dimenticare `Serial.begin()`

Se il programma utilizza `Serial.print()` o `Serial.println()` senza aver inizializzato la comunicazione con `Serial.begin()`, il comportamento atteso sul monitor seriale non sarà corretto.

Esercizi: monitor seriale

1. Spiegare la differenza tra `Serial.print()` e `Serial.println()`.
2. Scrivere un programma che stampi la frase "Avvio del sistema".
3. Dichiarare una variabile `int umidita = 35;` e stamparne il valore.
4. Scrivere sul monitor seriale il messaggio "Temperatura: " seguito dal valore della variabile `temperatura`.
5. Spiegare perché il testo da stampare deve essere scritto tra virgolette.
6. Individuare l'errore in un programma che utilizza `Serial.println(temperatura);` ma non contiene `Serial.begin(9600);`.
7. Preparare un output che mostri: "pH rilevato: " seguito dal valore della variabile `pH`.
8. Esercizio pratico: caricare uno sketch che invia un messaggio al monitor seriale e descrivere il risultato osservato.

A.8 Le uscite digitali

Un'**uscita digitale** può assumere due stati. Nel codice questi stati vengono indicati con le parole HIGH e LOW. Nelle prime attività l'uscita digitale sarà utilizzata per controllare un LED.



Figura A.14: Un'uscita digitale viene configurata e poi utilizzata per comandare un LED.

A.8.1 Configurare il pin come uscita

Prima di comandare un pin è necessario indicare che dovrà funzionare come uscita. Questa operazione viene normalmente eseguita in `setup()`.

Codice Arduino: configurazione di un'uscita

```
1 void setup() {  
2   pinMode(13, OUTPUT);  
3 }  
4  
5 void loop() {  
6 }
```

Nel codice precedente il numero 13 identifica il pin da configurare.

A.8.2 Impostare lo stato del pin

Per assegnare a un'uscita lo stato alto o basso si usa l'istruzione `digitalWrite()`.

Codice Arduino: accendere e spegnere un LED

```
1 void setup() {  
2   pinMode(13, OUTPUT);  
3 }  
4  
5 void loop() {  
6   digitalWrite(13, HIGH);  
7   digitalWrite(13, LOW);  
8 }
```

Nella pratica, l'effetto della seconda istruzione annulla immediatamente il precedente. Per osservare un cambiamento visibile è quindi utile introdurre una pausa.

A.8.3 La funzione `delay()`

L'istruzione `delay()` sospende il programma per un certo numero di millisecondi.

Codice Arduino: LED lampeggiante

```
1 void setup() {  
2   pinMode(13, OUTPUT);  
3 }  
4  
5 void loop() {  
6   digitalWrite(13, HIGH);  
7   delay(1000);  
8   digitalWrite(13, LOW);  
9   delay(1000);  
10 }
```

Nel programma precedente il LED rimane acceso per un secondo e spento per un secondo. Poiché `loop()` viene ripetuta continuamente, la sequenza si ripresenta senza fine.

Definizione: HIGH e LOW

In un'uscita digitale HIGH indica lo stato alto del pin, mentre LOW indica lo stato basso. A seconda del circuito, questi stati corrispondono all'accensione o allo spegnimento del componente collegato.

Errore comune: comandare un pin senza averlo configurato

Se un pin viene utilizzato con `digitalWrite()` senza essere stato configurato in precedenza mediante `pinMode(..., OUTPUT)`, il comportamento del circuito può non corrispondere a quello previsto.

Esercizi: uscite digitali

1. Spiegare che cosa significa configurare un pin come uscita.
2. Scrivere l'istruzione necessaria per configurare il pin 8 come uscita.
3. Scrivere l'istruzione necessaria per portare il pin 8 a HIGH.
4. Indicare la differenza tra HIGH e LOW.
5. Spiegare il ruolo di `delay(1000)` nel programma del LED lampeggiante.
6. Modificare il tempo di attesa in modo che il LED resti acceso per 2 secondi e spento per 0,5 secondi.
7. Individuare l'errore in uno sketch che usa `digitalWrite(13, HIGH);` ma non contiene `pinMode(13, OUTPUT);`.
8. Esercizio pratico: descrivere l'effetto osservato quando si carica il programma del LED lampeggiante su una scheda Arduino.

A.9 Gli ingressi digitali e le decisioni

Un **ingresso digitale** permette di leggere uno stato con due possibilità principali, per esempio pulsante premuto o non premuto. Dal punto di vista logico, questo tipo di ingresso si collega naturalmente alle decisioni studiate con i diagrammi di flusso.

Negli esempi seguenti il pulsante è collegato tra il pin digitale 2 e GND, e il pin è configurato come `INPUT_PULLUP`. Si usa il LED integrato della scheda Uno, associato al pin 13. A scheda scollegata, controllare che la pressione del pulsante chiuda effettivamente il collegamento tra i due punti; nei pulsanti a quattro terminali alcune coppie sono già unite internamente.

Pulsante fra D2 e GND; D2 configurato come INPUT_PULLUP

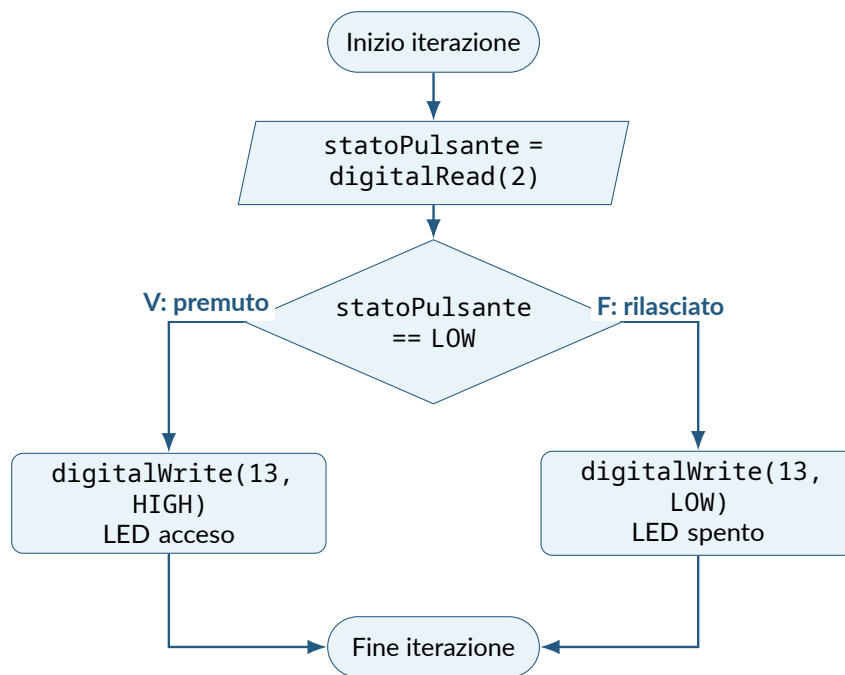


Figura A.15: Un pulsante può essere letto come ingresso digitale e utilizzato per decidere se accendere o spegnere un LED.

A.9.1 Leggere il valore di un pin

Per acquisire lo stato di un ingresso si usa l'istruzione `digitalRead()`.

Codice Arduino: lettura di un ingresso digitale

```

1 int statoPulsante;
2
3 void setup() {
4   pinMode(2, INPUT_PULLUP);
5 }
6
7 void loop() {
8   statoPulsante = digitalRead(2);
9 }
  
```

La variabile `statoPulsante` contiene il valore letto dal pin: HIGH con pulsante rilasciato e LOW con pulsante premuto.

A.9.2 Dalla decisione del diagramma all'istruzione `if`

Nel Capitolo 4 una decisione era rappresentata da un rombo e dai due rami V e F. Nel codice Arduino la stessa logica viene espressa con l'istruzione `if`; quando è necessario un comportamento alternativo, si utilizza anche `else`.

Codice Arduino: decisione con if/else

```
1 if (statoPulsante == LOW) {  
2   digitalWrite(13, HIGH);  
3 } else {  
4   digitalWrite(13, LOW);  
5 }
```

Se la condizione è vera, il LED viene acceso; se è falsa, il LED viene spento.

A.9.3 Un programma completo: pulsante e LED

Codice Arduino: pulsante che controlla un LED

```
1 int statoPulsante;  
2  
3 void setup() {  
4   pinMode(2, INPUT_PULLUP);  
5   pinMode(13, OUTPUT);  
6 }  
7  
8 void loop() {  
9   statoPulsante = digitalRead(2);  
10  
11   if (statoPulsante == LOW) {  
12     digitalWrite(13, HIGH);  
13   } else {  
14     digitalWrite(13, LOW);  
15   }  
16 }
```

Questo sketch traduce direttamente un diagramma di flusso molto semplice: leggere l'ingresso, verificare una condizione, decidere il valore dell'uscita.

Approfondimento: stati stabili dell'ingresso

La modalità INPUT_PULLUP attiva una resistenza interna che mantiene il pin a HIGH quando il pulsante è aperto. Collegando il pulsante tra il pin digitale 2 e GND, la pressione porta il pin a LOW. In questo circuito LOW significa quindi *premuto*; non è necessaria una resistenza esterna di polarizzazione dell'ingresso.

Errore comune: confondere l'assegnamento con il confronto

Nella condizione di un `if` si deve usare il confronto `==`. Scrivere `if (statoPulsante = LOW)` significa invece eseguire un'assegnazione, alterando il comportamento del programma.

Esercizi: ingressi digitali e decisioni

1. Spiegare che cosa restituisce `digitalRead(2)`.
2. Dichiarare una variabile adatta a memorizzare lo stato di un pulsante.
3. Scrivere un'istruzione `if` che accenda il LED sul pin 13 quando `statoPulsante == LOW`.
4. Completare il ramo `else` in modo che il LED venga spento.
5. Spiegare la relazione tra il rombo di un diagramma di flusso e l'istruzione `if`.
6. Individuare l'errore nell'istruzione `if (statoPulsante = LOW)`.
7. Disegnare il diagramma di flusso corrispondente al programma del pulsante che controlla un LED.
8. Esercizio pratico: descrivere che cosa accade al LED quando si preme e si rilascia il pulsante.

A.10 Gli ingressi analogici

Non tutti i fenomeni si prestano a essere descritti soltanto con due stati. Temperatura, luce, umidità e posizione di una manopola possono variare in modo continuo. In questi casi è utile un **ingresso analogico**.

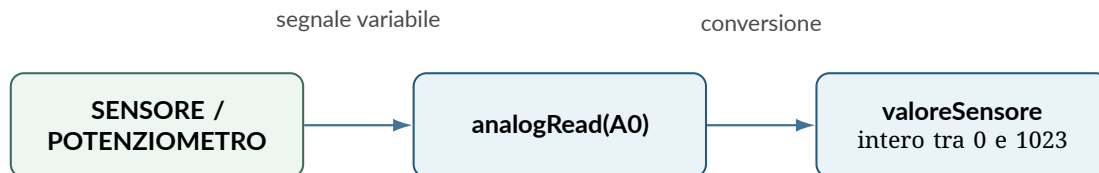


Figura A.16: Un sensore o un potenziometro può fornire un segnale analogico che Arduino trasforma in un valore numerico.

A.10.1 Leggere un ingresso analogico

Per acquisire il valore di un ingresso analogico si usa l'istruzione `analogRead()`.

Codice Arduino: lettura analogica

```

1 int valoreSensore;
2
3 void setup() {
4   Serial.begin(9600);
5 }
6
7 void loop() {
8   valoreSensore = analogRead(A0);
9   Serial.println(valoreSensore);
10 }
```

Nel caso di Arduino Uno il valore restituito è generalmente un numero intero compreso tra 0 e 1023. Più alto è il valore, maggiore è l'intensità del segnale letto dal pin.

A.10.2 Sensore reale e grandezza fisica

Il numero fornito da `analogRead()` non coincide sempre direttamente con la grandezza fisica che interessa. Può essere necessario interpretarlo o convertirlo.

Per esempio:

- un potenziometro produce un valore che dipende dalla posizione della manopola;
- un sensore di umidità può produrre valori diversi a seconda del terreno e della calibrazione;
- un sensore di temperatura può richiedere formule specifiche per trasformare la lettura in gradi.

Nelle prime esercitazioni è spesso sufficiente leggere il numero e confrontarlo con una soglia scelta sperimentalmente.

A.10.3 Osservare i valori sul monitor seriale

Il monitor seriale permette di vedere come cambia la lettura nel tempo. Questo è molto utile per scegliere una soglia realistica.

Codice Arduino: monitorare un ingresso analogico

```
1 int letturaUmidita;
2
3 void setup() {
4   Serial.begin(9600);
5 }
6
7 void loop() {
8   letturaUmidita = analogRead(A0);
9   Serial.print("Valore letto: ");
10  Serial.println(letturaUmidita);
11  delay(500);
12 }
```

Errore comune: interpretare il valore come una misura già calibrata

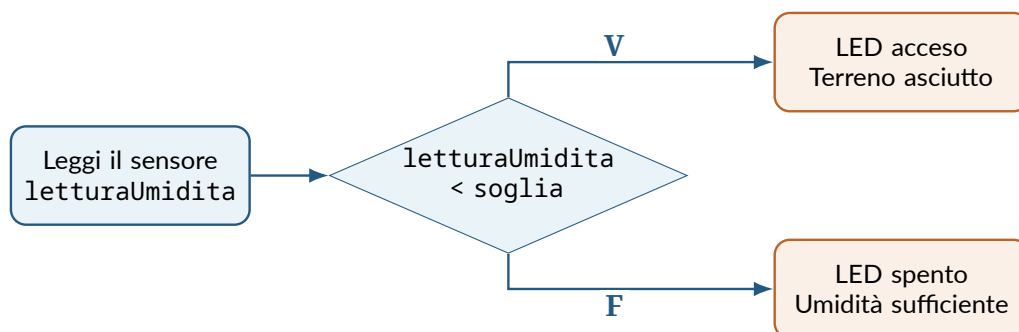
Il numero restituito da `analogRead()` non deve essere interpretato automaticamente come gradi, percentuale di umidità o altro. Prima è necessario conoscere il sensore utilizzato e il metodo di conversione scelto.

Esercizi: ingressi analogici

1. Spiegare la differenza tra ingresso digitale e ingresso analogico.
2. Scrivere l'istruzione necessaria per leggere il pin analogico A0 e salvarne il valore nella variabile lettura.
3. Indicare l'intervallo tipico dei valori restituiti da `analogRead()` su Arduino Uno.
4. Spiegare perché il monitor seriale è utile per studiare il comportamento di un sensore analogico.
5. Preparare un programma che legga un valore da A0 e lo stampi ogni mezzo secondo.
6. Descrivere un caso nel quale è sufficiente confrontare la lettura con una soglia senza convertirla in una misura fisica esatta.
7. Esercizio pratico: osservare i valori prodotti da un potenziometro o da un sensore e descriverne l'andamento.

A.11 Applicazioni agricole e chimiche

Le tecniche studiate nelle sezioni precedenti possono essere applicate a contesti vicini agli indirizzi agrario e chimico. Nelle attività iniziali si useranno esempi semplici, nei quali il comportamento del sistema viene rappresentato da un LED o da un messaggio sul monitor seriale.



Ipotesi verificata con la calibrazione: una lettura più bassa corrisponde a un terreno più asciutto. Se il sensore funziona al contrario, va invertito il confronto.

Figura A.17: Molte applicazioni elementari possono essere ricondotte allo schema: lettura di un valore, confronto con una soglia e attivazione di un'uscita.

A.11.1 Segnalare un terreno troppo asciutto

Un sensore misura una grandezza legata all'umidità del terreno; un potenziometro può invece simularne il segnale. In questo esempio si assume una calibrazione in cui valori analogici più bassi corrispondono a terreno più asciutto. La soglia 350 è soltanto un valore di prova da verificare sul dispositivo utilizzato. Se il sensore produce valori maggiori

quando il terreno è asciutto, occorre invertire il confronto e scegliere una soglia coerente. La lettura grezza non è una percentuale di umidità.

Codice Arduino: segnalazione di terreno asciutto

```
1 int letturaUmidita;  
2 const int soglia = 350;  
3  
4 void setup() {  
5   pinMode(13, OUTPUT);  
6   Serial.begin(9600);  
7 }  
8  
9 void loop() {  
10  letturaUmidita = analogRead(A0);  
11  
12  if (letturaUmidita < soglia) {  
13    digitalWrite(13, HIGH);  
14    Serial.println("Terreno asciutto");  
15  } else {  
16    digitalWrite(13, LOW);  
17    Serial.println("Umidità sufficiente");  
18  }  
19  
20  delay(500);  
21 }
```

A.11.2 Controllare una soglia di temperatura

In un sistema di monitoraggio ambientale si può simulare o leggere una temperatura e attivare una segnalazione quando il valore supera un limite.

Codice Arduino: allarme di temperatura

```
1 float temperatura = 32.5;  
2 const int ledAllarme = 13;  
3  
4 void setup() {  
5   pinMode(ledAllarme, OUTPUT);  
6   Serial.begin(9600);  
7 }  
8  
9 void loop() {  
10  if (temperatura > 30) {  
11    digitalWrite(ledAllarme, HIGH);  
12    Serial.println("Temperatura elevata");  
13  } else {  
14    digitalWrite(ledAllarme, LOW);  
15    Serial.println("Temperatura normale");  
16  }  
17  
18  delay(1000);  
19 }
```

A.11.3 Verificare un intervallo di pH

Alcune attività di ambito chimico possono richiedere il controllo di un intervallo di valori. Nella fase iniziale il valore di pH può essere simulato tramite una variabile.

Codice Arduino: controllo di un intervallo

```
1 float pH = 6.8;
2
3 void setup() {
4   Serial.begin(9600);
5 }
6
7 void loop() {
8   if ((pH >= 6.0) && (pH <= 8.0)) {
9     Serial.println("Valore accettabile");
10  } else {
11    Serial.println("Valore fuori intervallo");
12  }
13
14  delay(1000);
15 }
```

A.11.4 Selezionare un campione in modo pseudocasuale

Anche la generazione di numeri pseudocasuali può essere collegata ad attività concrete, per esempio per selezionare un campione da analizzare.

Codice Arduino: campione casuale

```
1 long campione;
2
3 void setup() {
4   Serial.begin(9600);
5   randomSeed(analogRead(A0));
6 }
7
8 void loop() {
9   campione = random(1, 11);
10  Serial.print("Campione estratto: ");
11  Serial.println(campione);
12  delay(1500);
13 }
```

Nel codice precedente `random(1, 11)` produce un numero intero compreso tra 1 e 10. La funzione `randomSeed()` inizializza il generatore pseudocasuale.

Approfondimento: dal prototipo al sistema reale

Nei sistemi reali un LED può essere sostituito da un buzzer, da un display, da un modulo relè o da altri dispositivi. Nelle prime attività, tuttavia, il LED costituisce un ottimo attuatore didattico: è semplice da collegare, sicuro e rende immediatamente visibile il risultato di una decisione.

Esercizi: applicazioni

1. Individuare input, condizione e output nel programma che segnala il terreno asciutto.
2. Spiegare perché il LED è adatto a rappresentare l'uscita di un prototipo.
3. Modificare il programma di controllo della temperatura in modo che la soglia diventi 28.
4. Scrivere una versione del programma del pH che distingua tra "valore troppo basso", "valore accettabile" e "valore troppo alto".
5. Spiegare il significato di `random(1, 11)`.
6. Modificare il programma del campione casuale per ottenere un numero tra 1 e 5.
7. Proporre un'applicazione agricola o chimica aggiuntiva che possa essere rappresentata con lo schema lettura-soglia-uscita.
8. Esercizio pratico: scegliere una delle quattro applicazioni presentate, realizzarne il diagramma di flusso e spiegare il funzionamento del relativo sketch.

A.12 Esercitazione conclusiva

L'appendice si conclude con una traccia che gli studenti possono utilizzare come punto di partenza per la loro scheda di laboratorio o relazione. Durante l'attività pratica il docente potrà fornire materiali, componenti, valori di soglia e indicazioni specifiche.

Preparazione alla scheda di laboratorio**Titolo dell'attività**

Scrivere qui il titolo del progetto o dell'esperienza di laboratorio.

Problema da risolvere

Descrivere in poche righe che cosa deve fare il sistema. Esempio: leggere un valore e accendere una segnalazione quando il dato supera o scende sotto una soglia.

Input

Indicare quale dato viene letto dal sistema (pulsante, sensore, valore analogico, numero simulato, ecc.).

Output

Indicare quale effetto produce il sistema (LED, messaggio sul monitor seriale, buzzer, ecc.).

Variabili e costanti

Elencare le variabili principali, il loro tipo e l'eventuale presenza di costanti come soglie o numeri di pin.

Componenti utilizzati

Elencare scheda Arduino, breadboard, cavetti, LED, resistenze, pulsanti, sensori o altri componenti.

Preparazione alla scheda di laboratorio (continua)**Descrizione dei collegamenti**

Riassumere come sono collegati i componenti. Se necessario, allegare uno schema o una fotografia del circuito.

Diagramma di flusso

Progettare il diagramma di flusso dell'algoritmo prima di scrivere il codice.

Codice

Inserire lo sketch o allegarlo alla relazione.

Casi di prova

Preparare una tabella con alcuni casi significativi: valori di ingresso, comportamento atteso e comportamento osservato.

Risultato ottenuto

Descrivere se il sistema si comporta come previsto.

Problemi incontrati e correzioni

Annotare eventuali errori, difficoltà di collegamento, problemi di sintassi o scelte modificate durante il lavoro.

Possibili miglioramenti

Indicare come il progetto potrebbe essere esteso o reso più utile in un contesto reale.

Esercizi: conclusione dell'appendice

1. Scegliere una delle applicazioni presentate e compilare una bozza della traccia di laboratorio.
2. Disegnare il diagramma di flusso di un sistema che legge un sensore e accende un LED quando il valore è fuori intervallo.
3. Individuare quali istruzioni devono comparire in `setup()` e quali in `loop()`.
4. Elencare almeno tre controlli di sicurezza da effettuare prima di alimentare il circuito.
5. Scrivere un breve commento conclusivo su ciò che cambia quando un algoritmo non rimane sullo schermo, ma controlla un dispositivo reale.

B. Creare e usare una mappa concettuale

Una mappa concettuale non è una semplice raccolta di parole racchiuse in riquadri. È una rappresentazione organizzata della conoscenza: seleziona i concetti essenziali, li dispone secondo una gerarchia e rende esplicite le relazioni attraverso parole-legame. In questa appendice il metodo viene applicato ai contenuti del Capitolo 1, dalla scelta dei concetti fino alla costruzione, revisione e presentazione di una mappa completa.

Costruire una mappa richiede prima di tutto comprensione. Per questo la mappa sarà considerata sia uno strumento di studio sia un prodotto che permette di verificare ciò che è stato realmente compreso.

Conoscenze	Abilità	Competenze
• concetto, nodo, relazione e parola-legame; • proposizione, gerarchia e collegamento trasversale; • differenza tra riassunto, schema, mappa mentale e mappa concettuale; • fasi di costruzione e criteri di revisione; • concetti principali del Capitolo 1.	• individuare e selezionare i concetti chiave; • distinguere categorie, parti ed esempi; • ordinare i concetti dal generale al particolare; • formulare parole-legame precise; • controllare correttezza e leggibilità.	• rappresentare graficamente un argomento complesso; • utilizzare la mappa per studiare, ripassare ed esporre; • rielaborare i contenuti senza copiarli meccanicamente; • motivare le scelte di selezione e organizzazione; • revisionare il proprio lavoro e quello dei compagni.

B.1 Che cos'è una mappa concettuale

Una **mappa concettuale** è una rappresentazione grafica formata da concetti collegati mediante relazioni esplicite. I concetti vengono scritti all'interno di nodi, mentre linee o frecce mostrano i collegamenti. Su ogni collegamento compare una parola o una breve espressione che spiega il rapporto tra i concetti.

Definizione: mappa concettuale

Una **mappa concettuale** è una rappresentazione organizzata di concetti e relazioni. I concetti sono disposti secondo una gerarchia e collegati da parole-legame che permettono di leggere proposizioni dotate di significato.

Il valore della mappa non dipende dal numero di riquadri o dai colori utilizzati. Una mappa è efficace quando consente di comprendere quali sono i concetti fondamentali, come dipendono da concetti più generali e quali relazioni collegano le diverse parti dell'argomento.

B.1.1 Gli elementi della mappa

Il componente fondamentale è il **concetto**: un oggetto, una categoria, un processo, una proprietà o un'idea. Nel Capitolo 1 sono concetti, per esempio, *computer*, *dato*, *hardware*, *CPU*, *software* e *sistema operativo*.

Due concetti collegati da una **parola-legame** formano una **proposizione**, cioè un'affermazione leggibile come una breve frase.



Figura B.1: Una proposizione è formata da due concetti collegati da una parola-legame.

La figura può essere letta da sinistra verso destra:

Il computer elabora dati.

La parola *elabora* non è un concetto dello stesso livello di *computer* e *dati*: descrive la relazione. Senza di essa rimarrebbero due parole vicine, ma il loro rapporto non sarebbe esplicito.

Esempio: una relazione precisa

La sequenza CPU → RAM non chiarisce il rapporto tra i due concetti. Una proposizione più informativa è:

CPU → scambia dati con → RAM

B.1.2 Gerarchia e collegamenti trasversali

In una mappa i concetti non vengono distribuiti casualmente. Il tema più generale si trova normalmente nella parte superiore o centrale; da esso derivano concetti progressivamente più specifici.

COMPUTER → HARDWARE → COMPONENTI INTERNI → CPU

Computer è più generale di *hardware*; *hardware* è più generale di *componenti interni*; *CPU* è un esempio specifico di componente interno. La posizione non indica importanza, ma livello di generalità.

Una mappa non è però soltanto un albero. Alcuni concetti appartenenti a rami differenti possono essere collegati da relazioni significative, dette **collegamenti trasversali**. Per esempio:

- la CPU, appartenente all'hardware, *esegue istruzioni* dei programmi software;
- il sistema operativo, appartenente al software, *gestisce* componenti hardware;
- i dispositivi di input *acquisiscono* dati.

I collegamenti trasversali mostrano che le aree non sono isolate, ma devono essere aggiunti solo quando chiariscono realmente l'argomento.

B.1.3 Utilità e confronto con altri strumenti

La costruzione di una mappa aiuta a individuare le informazioni essenziali, distinguere categorie ed esempi, preparare un'esposizione orale e controllare se l'argomento è stato compreso. Non sostituisce sempre il testo: il manuale può spiegare motivazioni, esempi ed eccezioni con una precisione difficilmente raggiungibile in pochi nodi.

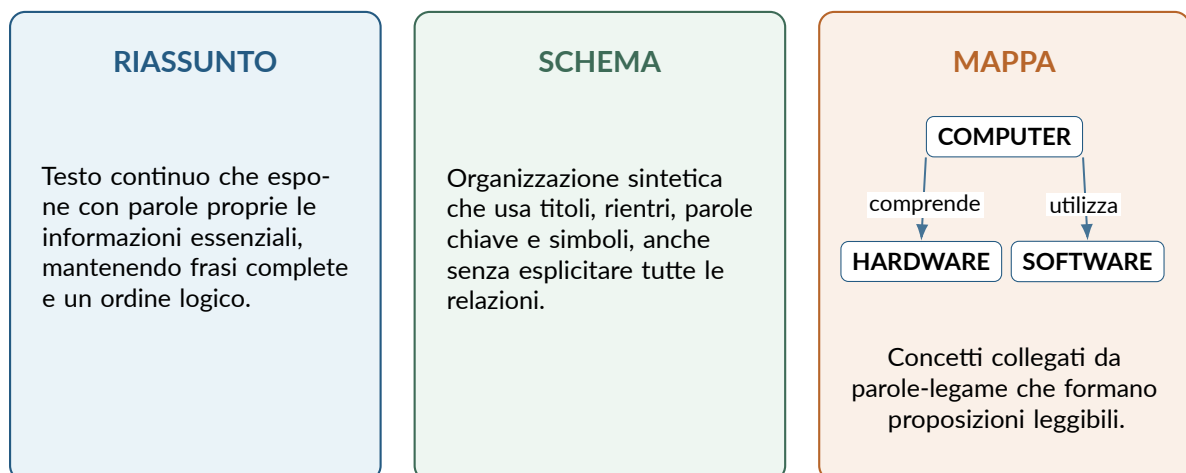


Figura B.2: Riassunto, schema e mappa concettuale organizzano le informazioni in modi differenti.

Riassunto

È un testo continuo che espone le informazioni principali con frasi complete.

Schema

Utilizza titoli, elenchi, rientri o tabelle, ma non richiede necessariamente che ogni collegamento sia espresso da una parola-legame.

Mappa concettuale

Rende esplicite gerarchia e relazioni: ogni percorso dovrebbe essere leggibile come una successione di proposizioni.

Mappa mentale

Parte spesso da un tema centrale e sviluppa rami associativi con parole, immagini e colori. È utile per raccogliere idee; la mappa concettuale pone maggiore attenzione alla precisione delle relazioni.

Approfondimento: la mappa come strumento di verifica

Quando non si riesce a trovare una parola-legame adatta, il problema può dipendere da una comprensione incompleta. La difficoltà nel collegare due concetti suggerisce quindi di rileggere il testo e chiarire il rapporto prima di completare la mappa.

Esercizi: comprendere la mappa concettuale

1. Definire con parole proprie una mappa concettuale.
2. Distinguere concetto, parola-legame e proposizione.
3. Trasformare la coppia RAM -- dati temporanei in una proposizione completa.
4. Ordinare dal generale al particolare: CPU, hardware, componenti interni, computer.
5. Individuare un possibile collegamento trasversale tra hardware e software.
6. Confrontare riassunto e mappa concettuale indicando un vantaggio di ciascuno.

B.2 Come si costruisce una mappa concettuale

La costruzione non dovrebbe iniziare disegnando immediatamente riquadri e frecce. Prima occorre definire lo scopo, comprendere il testo, selezionare i concetti e stabilire un ordine. Il lavoro può essere organizzato in sei fasi.

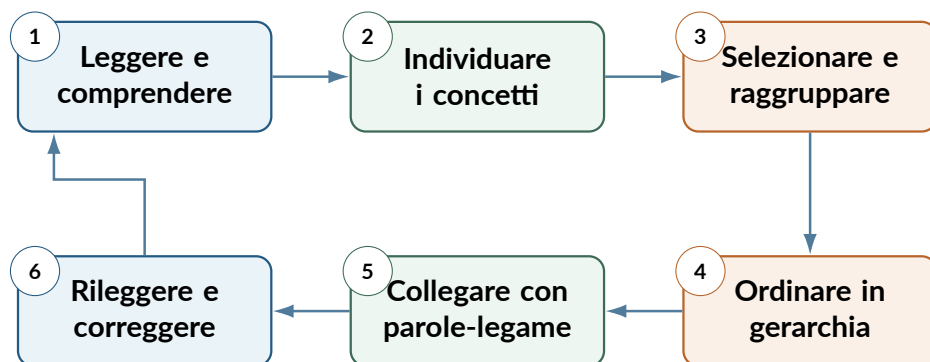


Figura B.3: Le principali fasi di costruzione di una mappa concettuale.

1. Definire lo scopo e comprendere il testo

La domanda guida stabilisce il livello di dettaglio. Durante una prima lettura si riconosce il tema generale; una seconda lettura si concentra su definizioni, classificazioni e rapporti tra le parti. Per il Capitolo 1, alcune domande possibili sono:

- che cos'è un computer?
- quali operazioni svolge?
- che cosa sono dati e informazioni?
- come collaborano hardware e software?

2. Individuare e selezionare i concetti

I concetti sono spesso espressi da nomi o brevi gruppi nominali. Si riconoscono nelle definizioni, nei titoli, nelle parole evidenziate e nei termini ricorrenti. Non tutte le parole del testo devono entrare nella mappa: vanno eliminati duplicazioni, dettagli secondari e concetti troppo specifici rispetto allo scopo.

Esempio: dal testo ai concetti

Testo: "La RAM conserva temporaneamente dati e istruzioni utilizzati durante l'elaborazione."

Concetti: RAM -- dati e istruzioni -- elaborazione.

Parola-legame: conserva temporaneamente.

3. Raggruppare e ordinare

I concetti selezionati vengono raccolti in categorie e disposti dal generale al particolare. È importante distinguere:

- **categorie**, come *periferiche*;
- **parti**, come *CPU* o *RAM*;
- **esempi**, come *tastiera*, *SSD* o *browser*.

Una procedura pratica consiste nel collocare i concetti su piccoli foglietti o riquadri mobili, spostarli finché la gerarchia risulta coerente e tracciare le frecce soltanto dopo aver stabilito le posizioni.

4. Formulare le proposizioni

Un nodo dovrebbe contenere un concetto, non un intero paragrafo. La frase “la memoria che conserva temporaneamente dati e istruzioni” può essere scomposta in:

RAM → conserva temporaneamente → dati e istruzioni

Le parole-legame devono essere informative. Formule generiche come *ha* o *fa* possono essere sostituite da relazioni più precise.

Relazione	Formula generica	Formula più precisa
Computer e dati	ha dati	riceve, elabora e memorizza dati
Hardware e CPU	ha una CPU	comprende la CPU
CPU e istruzioni	fa istruzioni	esegue istruzioni
Sistema operativo e risorse	usa le risorse	gestisce le risorse
SSD e file	ha file	conserva file in modo permanente

Tabella B.1: Esempi di parole-legame rese più informative.

Una freccia stabilisce anche l'ordine della lettura: CPU esegue istruzioni è corretta, mentre istruzioni eseguono CPU attribuisce l'azione al concetto sbagliato.

Molte relazioni appartengono a tre tipi ricorrenti:

- **categoria:** il software *si divide in* software di sistema e applicativo;
- **composizione:** l'hardware *comprende* componenti fisici;
- **funzione:** il sistema operativo *gestisce* risorse hardware.

Prima del disegno definitivo conviene compilare una tabella di progettazione.

Concetto iniziale	Parola-legame	Concetto finale
computer	elabora	dati
hardware	comprende	componenti fisici
CPU	esegue	istruzioni
sistema operativo	gestisce	risorse hardware
software applicativo	permette di svolgere	attività specifiche

Tabella B.2: Controllo preliminare di alcune proposizioni.

5. Disegnare la prima bozza

Nella prima bozza si costruisce la struttura, senza cercare subito una grafica definitiva. Occorre:

- collocare il concetto più generale nella parte superiore o centrale;
- disporre le grandi categorie al livello successivo;
- aggiungere parti, tipi ed esempi;
- mantenere distanze regolari e caratteri leggibili;
- evitare linee che attraversano nodi o parole-legame;
- usare forme e colori secondo una regola coerente.

Un programma per mappe concettuali facilita lo spostamento dei nodi, ma non sceglie i concetti e non verifica automaticamente la correttezza delle relazioni.

6. Rileggere e revisionare

Ogni percorso va letto seguendo il verso delle frecce. La revisione controlla correttezza, completezza e leggibilità. Una mappa dovrebbe rimanere comprensibile anche in scala di grigi e non dovrebbe affidare il significato soltanto ai colori.

Approfondimento: lista di controllo finale

Prima di considerare conclusa la mappa, verificare:

- lo scopo è chiaro e la mappa risponde alla domanda guida?
- i nodi contengono concetti brevi e non frasi intere?
- la gerarchia distingue categorie, parti ed esempi?
- ogni freccia forma una proposizione corretta?
- i collegamenti trasversali aggiungono informazioni utili?
- il testo è leggibile e gli elementi non si sovrappongono?

Esercizi: applicare il metodo

1. Individuare quattro concetti nella frase: "Il sistema operativo gestisce le risorse hardware e permette l'esecuzione dei programmi applicativi."
2. Eliminare i termini meno adatti a una mappa generale: CPU, RAM, vite del case, software, singolo tasto della tastiera, hardware.
3. Raggruppare monitor, immagine, browser, CPU, testo, sistema operativo e tastiera nelle categorie hardware, software e dati.
4. Rendere più precisa la relazione CPU → fa → calcoli.
5. Correggere la direzione di programmi → eseguono → computer.
6. Scrivere un collegamento trasversale tra hardware e software.
7. Compilare una tabella con cinque proposizioni dedicate al sistema operativo.
8. Preparare una lista di controllo personale con cinque domande.

B.3 Esempio guidato: la mappa del Capitolo 1

L'esempio seguente mostra l'intero processo applicato al Capitolo 1. L'obiettivo non è inserire ogni definizione, ma rappresentare i nuclei principali e le relazioni essenziali.

B.3.1 Domanda guida e concetti essenziali

La mappa deve rispondere alla domanda:

*Che cos'è un **computer** e quali elementi permettono il suo funzionamento?*

La domanda richiede di rappresentare definizione, operazioni fondamentali, dati e informazioni, hardware e software. Non richiede invece di inserire tutti gli esempi di porte, licenze o modelli commerciali.

Una prima raccolta può contenere:

- computer;
- elettronico;
- digitale;
- programmabile;
- dati;
- informazioni;
- input;
- elaborazione;
- memoria;
- output;
- hardware;
- periferiche;
- componenti interni;
- CPU;
- RAM;
- memorie di massa;
- software;
- sistema operativo;
- software applicativo;
- programmi;
- istruzioni;
- licenze.

La lista non è ancora ordinata e contiene concetti di livelli diversi. Per una mappa sintetica si possono distinguere cinque aree.

Area	Concetti essenziali	Possibili approfondimenti
Computer	elettronico, digitale, programmabile	programma, istruzioni
Funzionamento	input, elaborazione, memoria, output	esempi di dispositivi
Dati	dato, informazione, contesto	metadati, qualità, tipi di dato
Hardware	periferiche, componenti interni	porte, von Neumann, CPU, RAM, SSD
Software	sistema, applicativo, sistema operativo	licenze e distribuzione

Tabella B.3: Una selezione dei concetti del Capitolo 1.

B.3.2 Dalla lista alla struttura

Il concetto più generale è *computer*. Per progettare la struttura si raggruppano i contenuti relativi a caratteristiche del computer, operazioni fondamentali, dati e informazioni, hardware e software. CPU e RAM non devono essere poste sullo stesso livello di hardware, perché appartengono alla categoria dei componenti interni.

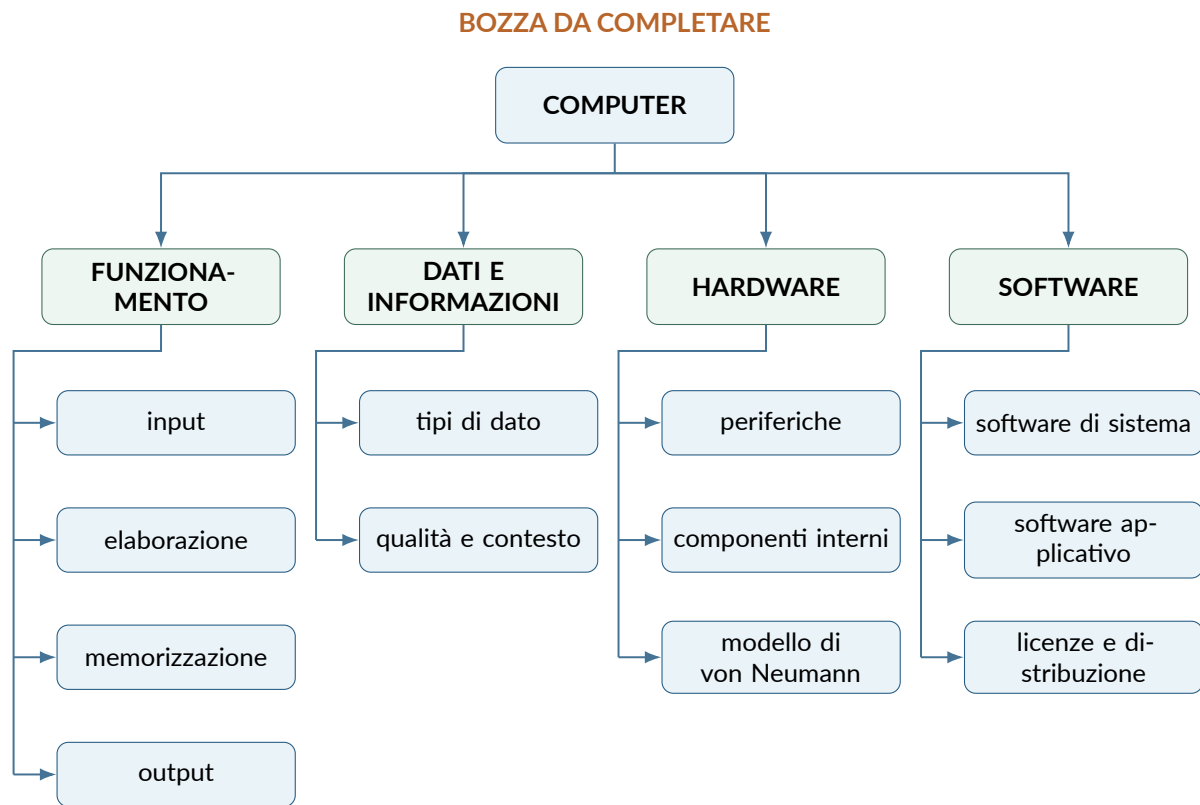


Figura B.4: Gerarchia preliminare dei concetti del Capitolo 1: bozza da completare e correggere prima di usarla come mappa.

La figura è una bozza, non ancora una mappa completa: serve a verificare la posizione dei concetti prima di formulare tutte le relazioni. Le etichette che raggruppano temi, come *definizione*, vanno poi trasformate in concetti adatti a costruire proposizioni corrette.

Le proposizioni principali possono essere progettate in anticipo:

1. il computer è una macchina elettronica, digitale e programmabile;
2. il computer riceve, elabora, memorizza e comunica dati;
3. i dati interpretati in un contesto producono informazioni;
4. il computer comprende hardware;
5. l'hardware comprende periferiche e componenti interni;
6. i componenti interni comprendono CPU, RAM e memorie di massa;
7. il computer utilizza software;
8. il software si divide in software di sistema e applicativo;

9. il sistema operativo gestisce le risorse hardware;
10. i programmi contengono istruzioni;
11. l'uso e la distribuzione del software sono regolati da licenze.

Questa lista costituisce il progetto logico della mappa.

B.3.3 Dalla prima bozza alla mappa completa

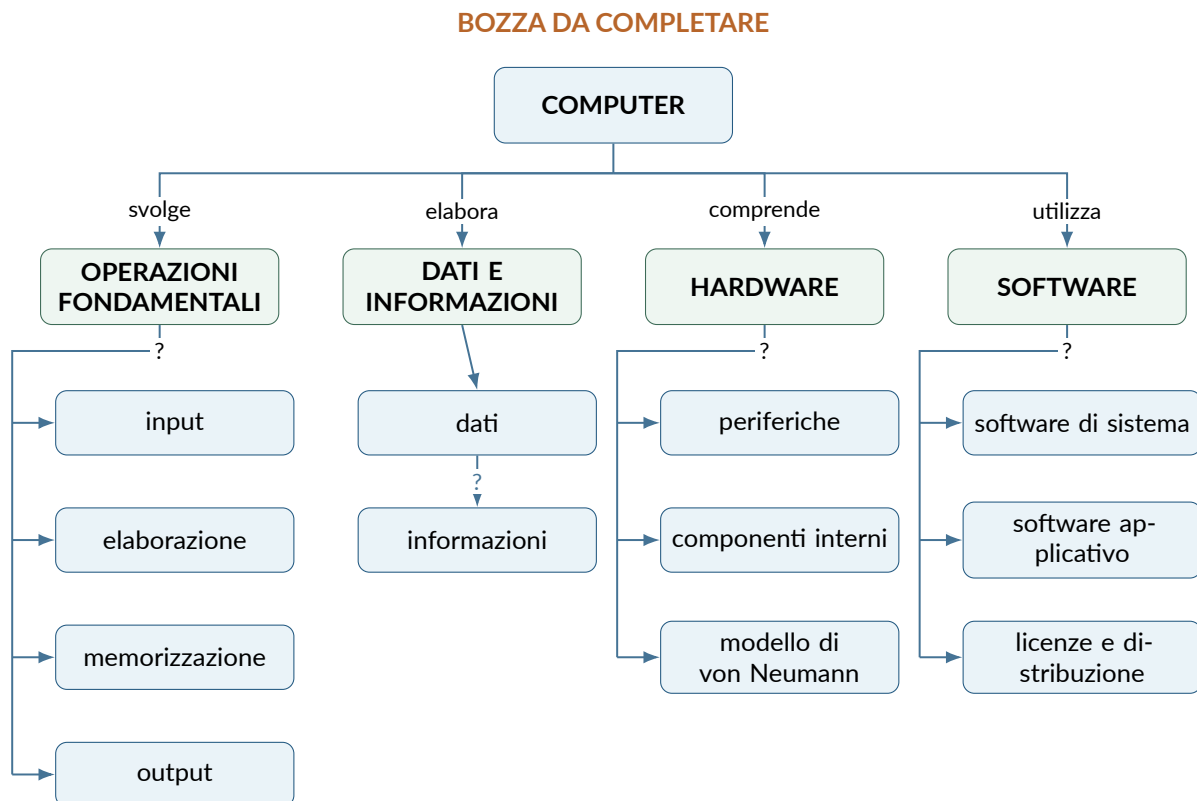


Figura B.5: Prima bozza guidata della mappa del Capitolo 1: relazioni da completare e correggere.

La prima bozza rende riconoscibili i rami principali, ma richiede ancora alcuni interventi:

- completare le parole-legame dei collegamenti secondari;
- verificare la relazione tra dati e informazioni;
- aggiungere un collegamento tra sistema operativo e hardware;
- aggiungere la definizione mediante la proposizione *computer è una macchina elettronica, digitale e programmabile*;
- separare le categorie del software, sistema e applicativo, dalle condizioni di licenza e distribuzione;
- evitare sovrapposizioni tra linee, nodi ed etichette.

Esempio: revisione di un ramo

Prima versione:

HARDWARE → COMPONENTI INTERNI → CPU

Versione completata:

HARDWARE → comprende → COMPONENTI INTERNI
COMPONENTI INTERNI → comprendono → CPU

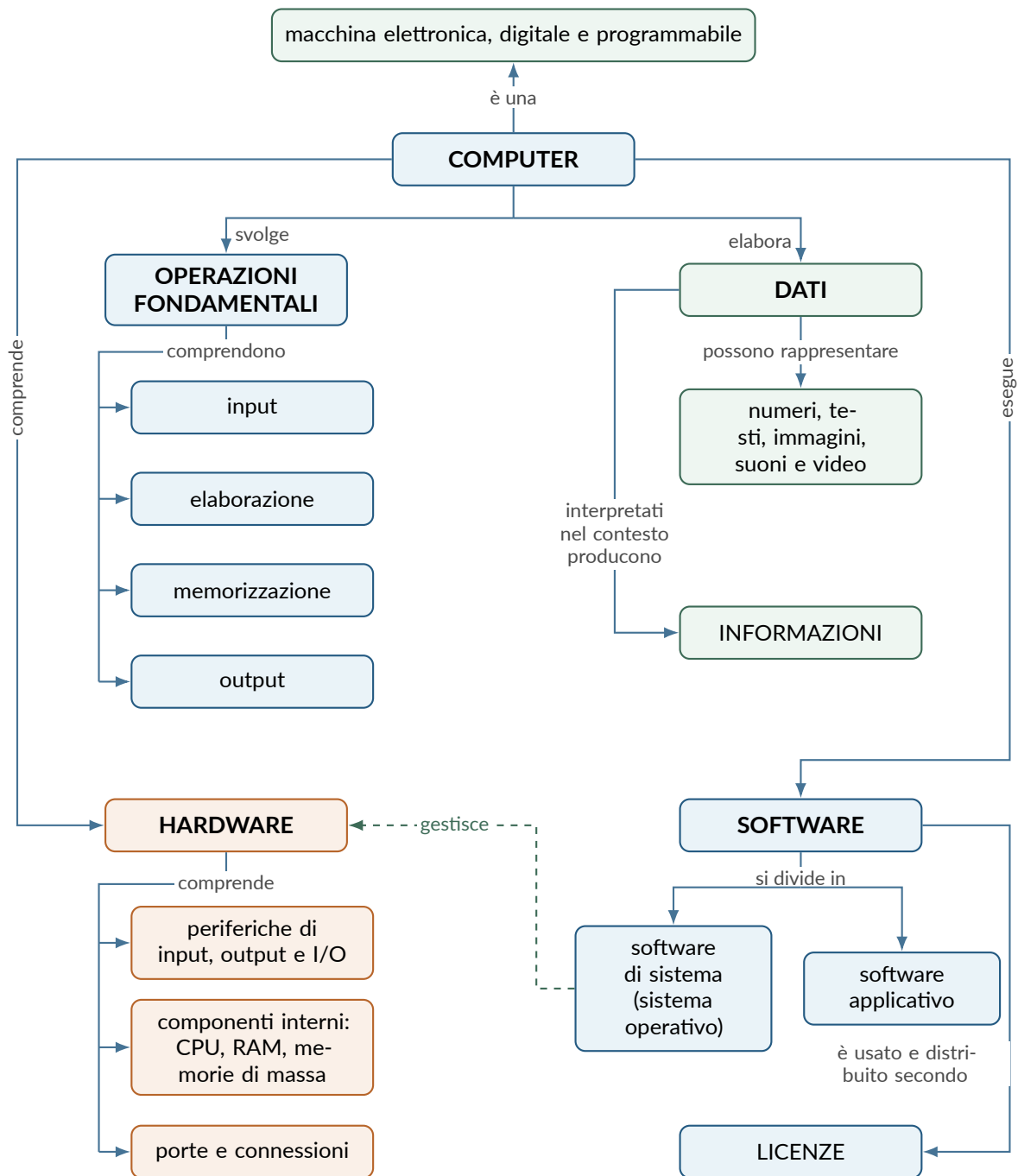


Figura B.6: Una possibile mappa concettuale completa e sintetica del Capitolo 1.

La versione finale mantiene una struttura generale e rinuncia ai dettagli che produrrebbero una pagina troppo affollata. Non esiste un'unica disposizione corretta: studenti diversi possono scegliere un livello di dettaglio differente, purché rispettino contenuti, gerarchia e correttezza delle proposizioni.

B.3.4 Leggere e utilizzare la mappa

La lettura non deve limitarsi ai singoli nodi. Occorre seguire le frecce e pronunciare proposizioni complete. Per esempio:

- il computer svolge operazioni di input, elaborazione, memorizzazione e output;
- i dati interpretati nel contesto producono informazioni;
- l'hardware si articola in periferiche, componenti interni, porte e connessioni;
- il software si divide in software di sistema e software applicativo;
- il sistema operativo gestisce l'hardware.

L'ultima relazione è trasversale e mostra che hardware e software non funzionano in modo indipendente.

I colori possono distinguere i rami, ma il significato deve dipendere anche da parole, posizione e collegamenti. La mappa dovrebbe rimanere comprensibile quando viene stampata in scala di grigi o visualizzata su uno schermo piccolo.

Una mappa generale può inoltre rimandare a mappe di approfondimento dedicate, per esempio, alle periferiche, ai componenti interni, al modello di von Neumann, ai tipi di software o alle licenze.

Per studiare è possibile:

1. coprire alcuni nodi e provare a ricostruirli;
2. leggere ogni ramo come una breve spiegazione orale;
3. trasformare le proposizioni in domande;
4. confrontare la mappa con il testo e cercare omissioni;
5. realizzare una versione ridotta senza consultare l'originale.

Approfondimento: una mappa è una scelta motivata

La completezza non coincide con la quantità di nodi. Una mappa è completa rispetto al proprio scopo quando rappresenta tutti i concetti necessari per rispondere alla domanda iniziale. Aggiungere informazioni non utili può ridurne l'efficacia.

Esercizi: analizzare l'esempio

1. Individuare i cinque rami principali della Figura B.6.
2. Leggere ad alta voce cinque proposizioni complete seguendo le frecce.
3. Individuare un collegamento trasversale e spiegarne l'utilità.
4. Aggiungere il nodo *scheda madre* nella posizione gerarchica corretta.
5. Aggiungere il nodo *browser* e indicare la parola-legame.
6. Eliminare due nodi per ottenere una versione ancora più sintetica, motivando la scelta.
7. Proporre una mappa di approfondimento derivata dal nodo *componenti interni*.
8. Esporre oralmente per due minuti un ramo della mappa senza leggere il capitolo.

B.4 Errori comuni e controllo di qualità

Gli errori più frequenti non riguardano soltanto l'aspetto grafico. Spesso derivano da una selezione poco accurata dei concetti o da relazioni che non formano proposizioni corrette.

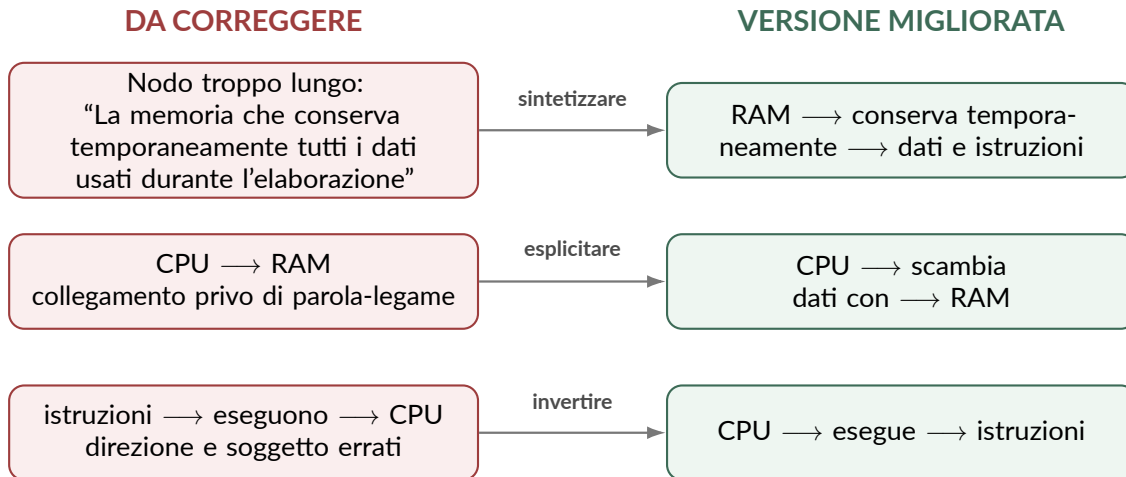


Figura B.7: Alcuni errori frequenti nella costruzione di una mappa concettuale.

Errore	Perché crea problemi	Come correggerlo
Frase lunghe nei nodi	Nascondono la relazione e occupano spazio	Separare concetto iniziale, parola-legame e concetto finale
Freccie senza parole-legame	Il rapporto rimane ambiguo	Scrivere un verbo o un'espressione precisa
Direzione invertita	La proposizione diventa falsa o priva di senso	Leggere seguendo il verso della freccia
Categorie ed esempi confusi	La gerarchia non è coerente	Inserire livelli intermedi appropriati
Troppi concetti	La struttura principale non è riconoscibile	Creare una mappa generale e mappe di approfondimento
Troppi collegamenti	Le linee nascondono i rami principali	Conservare solo le relazioni realmente utili
Significato affidato ai colori	La mappa è poco accessibile	Usare anche testo, posizione e forme coerenti
Copia senza comprensione	Non avviene rielaborazione	Spiegare e modificare la mappa motivando le scelte

Tabella B.4: Errori ricorrenti e possibili correzioni.

Errore comune: la mappa più grande non è necessariamente la migliore

Una mappa con molti nodi può sembrare più completa, ma risultare meno efficace di una rappresentazione sintetica e ben organizzata. La qualità dipende da correttezza, selezione e leggibilità.

Approfondimento: controllo di qualità

Una buona revisione risponde a quattro domande:

1. **Correttezza:** ogni proposizione è vera e grammaticalmente leggibile?
2. **Gerarchia:** categorie, parti ed esempi sono collocati al livello appropriato?
3. **Essenzialità:** ogni nodo è necessario rispetto alla domanda guida?
4. **Leggibilità:** nodi, frecce e parole-legame sono distinguibili senza sovrapposizioni?

Esercizi: riconoscere e correggere

1. Trasformare il nodo “il dispositivo che permette di inserire testi nel computer” in una proposizione con nodi brevi.
2. Aggiungere una parola-legame alla coppia sistema operativo--hardware.
3. Correggere la direzione di dati → acquisiscono → tastiera.
4. Ordinare correttamente: hardware, periferiche, tastiera, periferiche di input.
5. Valutare se il collegamento CPU--colore del case sia utile in una mappa sul funzionamento del computer.
6. Costruire una piccola mappa con tre errori e consegnarla a un compagno perché li individui.

B.5 Laboratorio: costruire la propria mappa

L'attività seguente conduce alla realizzazione autonoma di una mappa del Capitolo 1. Il prodotto non deve essere una copia identica della Figura B.6: ogni scelta deve essere coerente con lo scopo e motivata.

B.5.1 Consegna e vincoli

Attività guidata: costruire e presentare una mappa**Obiettivo**

Realizzare una mappa concettuale che risponda alla domanda:

Che cos'è un computer e come collaborano hardware, software e dati?

Vincoli minimi

- utilizzare da 12 a 20 nodi;
- inserire almeno tre livelli gerarchici;
- rappresentare hardware, software e dati;
- utilizzare una parola-legame per ogni freccia;

Attività guidata: costruire e presentare una mappa (continua)

- aggiungere almeno un collegamento trasversale;
- includere almeno due componenti interni e due esempi di software;
- accompagnare la mappa con una breve esposizione orale.

Materiali

Capitolo 1, foglio grande o strumento digitale, tabella delle proposizioni, eventuali colori e immagini essenziali.

B.5.2 Realizzazione e revisione

1. Progettazione

Scrivere la domanda guida e preparare una lista di almeno venticinque termini. Ridurla poi ai concetti indispensabili e compilare almeno dodici righe secondo il modello:

concetto iniziale -- parola-legame -- concetto finale

La selezione deve includere almeno:

computer -- dati -- hardware -- software -- CPU -- RAM -- sistema
operativo

2. Realizzazione

Disporre i concetti in livelli gerarchici senza curare inizialmente i dettagli estetici. La bozza deve rendere riconoscibili il concetto centrale, le grandi categorie, i concetti specifici e il collegamento trasversale. Solo dopo la disposizione si aggiungono frecce e parole-legame.

3. Revisione tra pari ed esposizione

Scambiare la bozza con un compagno. Il revisore legge almeno dieci proposizioni e annota:

- una relazione particolarmente chiara;
- una parola-legame da migliorare;
- un concetto eliminabile o eventualmente mancante;
- un problema grafico, se presente.

L'autore decide quali modifiche applicare e ne motiva almeno una. L'esposizione finale, della durata indicativa di tre-cinque minuti, presenta il concetto centrale, i rami principali, alcune proposizioni significative, il collegamento trasversale e una scelta di organizzazione.

B.5.3 Valutazione e autovalutazione

Criterio	Indicatore	Punti
Correttezza	Concetti e proposizioni coerenti con il Capitolo 1	0–4
Selezione	Presenza dei nuclei essenziali senza dettagli inutili	0–3
Gerarchia	Livelli chiari e categorie collocate correttamente	0–3
Parole-legame	Relazioni precise e leggibili nella direzione delle frecce	0–4
Organizzazione grafica	Nodi, etichette e linee non sovrapposti	0–3
Esposizione	Spiegazione autonoma e uso della mappa come guida	0–3
Totale		20

Tabella B.5: Possibile griglia di valutazione dell'attività.

Al termine del lavoro, indicare per ciascuna affermazione il livello raggiunto.

Affermazione	Sì	In parte	Non ancora
So individuare i concetti principali in un testo.	☐	☐	☐
So distinguere una categoria da un esempio.	☐	☐	☐
So ordinare i concetti in una gerarchia.	☐	☐	☐
So scegliere parole-legame precise.	☐	☐	☐
So leggere una mappa come una serie di proposizioni.	☐	☐	☐
So riconoscere e correggere una relazione sbagliata.	☐	☐	☐
So costruire una mappa leggibile.	☐	☐	☐
So utilizzare la mappa per spiegare un argomento.	☐	☐	☐

Tabella B.6: Scheda di autovalutazione sulle mappe concettuali.

Esercizi conclusivi

1. Realizzare una mappa di massimo otto nodi dedicata alle operazioni fondamentali del computer.
2. Realizzare una mappa su dato, informazione, contesto, metadati e tipi di dato.
3. Realizzare una mappa di approfondimento sui componenti interni.
4. Realizzare una mappa sui tipi di software con almeno due esempi per categoria.
5. Trasformare una pagina del Capitolo 1 prima in schema e poi in mappa concettuale; confrontare i due prodotti.
6. Preparare una spiegazione di due minuti del ramo hardware.

7. Creare una versione incompleta della mappa, lasciando cinque nodi vuoti da far completare a un compagno.

Esercizi: prodotto conclusivo

Realizzare una versione definitiva della mappa concettuale del Capitolo 1. Allegare:

1. la lista iniziale dei concetti;
2. la tabella con almeno dodici proposizioni;
3. la prima bozza;
4. la revisione ricevuta da un compagno;
5. la mappa definitiva;
6. un breve testo che spieghi due modifiche effettuate durante il lavoro.

C. Uso consapevole delle tecnologie e benessere digitale

Computer, smartphone, piattaforme online e dispositivi connessi permettono di comunicare, imparare, creare e accedere rapidamente a informazioni e servizi. Gli stessi strumenti, però, possono sottrarre tempo, interrompere l'attenzione, favorire abitudini sedentarie o interferire con il riposo quando vengono utilizzati senza uno scopo chiaro e senza limiti adeguati.

Un uso consapevole non richiede di rifiutare la tecnologia. Richiede piuttosto di riconoscere ciò che lo strumento sta permettendo di fare, osservare le proprie abitudini e scegliere quando iniziare, continuare o interrompere un'attività digitale. L'obiettivo di questa appendice è quindi sviluppare autonomia, non imporre regole identiche per tutti.

Conoscenze	Abilità	Competenze
• opportunità e possibili criticità delle tecnologie digitali; • attenzione, interruzione e passaggio tra attività; • funzioni che favoriscono la prosecuzione dell'uso; • rapporti tra dispositivi, sonno, postura, vista, movimento e ascolto; • social network, identità digitale, confronto sociale e uso problematico.	• distinguere un uso scelto da un uso automatico; • riconoscere le principali fonti di interruzione; • osservare il proprio tempo digitale senza giudizi affrettati; • organizzare una postazione e una routine più sostenibili; • riconoscere contenuti e comportamenti che possono danneggiare sé o gli altri.	• utilizzare le tecnologie in funzione di uno scopo; • proteggere attenzione e tempo di studio; • adottare comportamenti compatibili con il benessere fisico; • modificare in modo realistico un'abitudine digitale poco utile; • partecipare agli ambienti online in modo rispettoso e responsabile.

C.1 Tecnologia: opportunità e responsabilità

Le tecnologie digitali non producono sempre lo stesso effetto. Una videochiamata può mantenere una relazione a distanza; la stessa applicazione può interrompere continuamente un'attività. Un video può spiegare un procedimento difficile; una successione automatica di video può occupare molto più tempo del previsto. Un gruppo di messaggistica può favorire la collaborazione; può anche creare pressione a rispondere immediatamente.

Non è quindi sufficiente chiedersi se una tecnologia sia “buona” o “cattiva”. È più utile osservare:

- **lo scopo:** perché la si sta utilizzando;
- **il tempo:** quanto dura l'attività e se la durata è stata scelta;
- **il contesto:** studio, riposo, conversazione, spostamento o tempo libero;
- **le conseguenze:** che cosa viene facilitato e che cosa viene trascurato;
- **il controllo:** se è possibile interrompere senza particolare difficoltà.

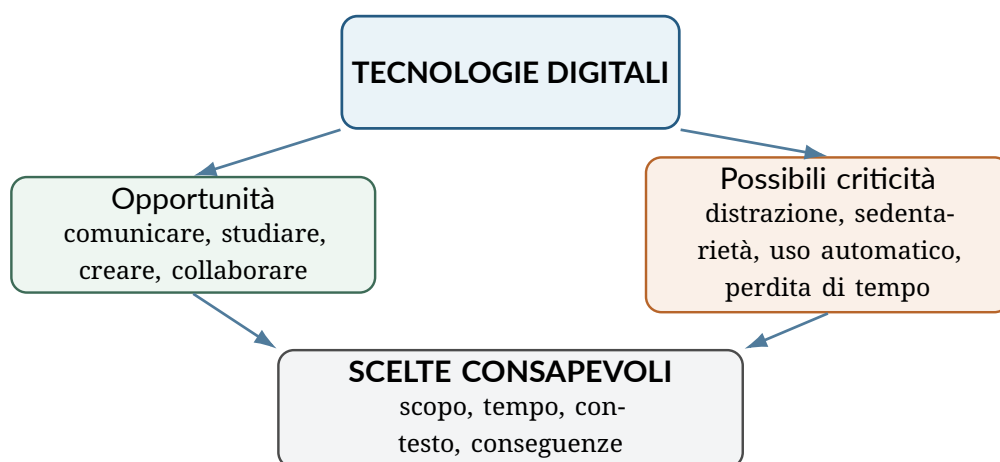


Figura C.1: Le tecnologie possono offrire opportunità o creare criticità; le conseguenze dipendono anche dalle scelte d'uso.

C.1.1 Usare uno strumento per raggiungere uno scopo

Un comportamento è più consapevole quando lo scopo precede l'apertura dell'applicazione. Per esempio:

devo cercare un'informazione precisa → apro il browser

è diverso da:

apro il telefono senza uno scopo → seguo ciò che compare

Nel primo caso lo strumento viene scelto per svolgere un'attività. Nel secondo è l'ambiente digitale a proporre continuamente nuove azioni. Non tutte le aperture automatiche sono

dannose, ma riconoscerle permette di decidere se continuare o tornare a ciò che si stava facendo.

Definizione: uso consapevole

L'**uso consapevole** delle tecnologie è la capacità di scegliere uno strumento digitale in funzione di uno scopo, valutarne gli effetti e interromperne l'utilizzo quando non è più utile o appropriato.

C.1.2 La quantità di tempo non racconta tutto

Il tempo trascorso davanti a uno schermo è un'informazione utile, ma non descrive da solo la qualità dell'esperienza. Due ore utilizzate per preparare una presentazione, comunicare con un familiare e seguire una lezione non sono equivalenti a due ore trascorse passando automaticamente da un contenuto all'altro.

Per valutare un'attività digitale è quindi opportuno considerare almeno tre domande:

1. Che cosa è stato fatto durante quel tempo?
2. L'attività era stata scelta oppure è proseguita senza accorgersene?
3. Al termine ci si sente informati, soddisfatti e riposati oppure distratti, affaticati o in ritardo?

Errore comune: misurare soltanto le ore

Un numero totale può nascondere attività molto diverse. Ridurre il tempo digitale senza distinguere studio, creatività, comunicazione e uso automatico può portare a conclusioni poco utili. È preferibile osservare anche scopo, contesto e conseguenze.

C.1.3 Responsabilità verso se stessi e verso gli altri

La tecnologia amplia la possibilità di agire. Un contenuto può essere creato, copiato e condiviso in pochi secondi; un messaggio può raggiungere molte persone; una fotografia può uscire dal contesto nel quale era stata scattata. Per questo l'uso consapevole riguarda sia il proprio benessere sia gli effetti prodotti sugli altri.

Prima di pubblicare, inoltrare o commentare è utile chiedersi:

- il contenuto è corretto e comprensibile nel suo contesto?
- riguarda anche altre persone?
- potrebbe creare imbarazzo, pressione o danno?
- sarei disposto a sostenerne la condivisione anche fuori dal gruppo attuale?

Questi temi verranno ripresi nelle sezioni dedicate ai social network e alle relazioni digitali. In questa prima parte è sufficiente riconoscere che un'azione online non è separata dalla vita quotidiana: può avere conseguenze reali e durature.

Attività guidata: osservare un'attività digitale

Scegliere un'attività digitale svolta di recente e annotare: scopo iniziale, tempo previsto, tempo effettivo, utilità ottenuta e una possibile modifica per la prossima volta. L'obiettivo non è giudicare la tecnologia, ma descrivere con precisione l'abitudine.

Esercizi: opportunità e responsabilità

1. Spiegare perché la stessa applicazione può risultare utile in un contesto e inopportuna in un altro.
2. Distinguere tra uso scelto e uso automatico mediante un esempio.
3. Indicare quali informazioni, oltre al numero di ore, aiutano a valutare un'attività digitale.
4. Commentare la frase: "Un uso consapevole non coincide con il rifiuto della tecnologia".

C.2 Attenzione, notifiche e tempo digitale

L'attenzione permette di selezionare le informazioni utili e mantenere uno scopo mentre vengono ignorati stimoli meno importanti. Non è una risorsa illimitata: leggere, comprendere, risolvere un problema e ricordare istruzioni richiedono continuità.

Una notifica può interrompere questa continuità anche quando non viene aperta. Il suono, la vibrazione o l'anteprima possono spostare per alcuni istanti i pensieri verso il messaggio ricevuto. Studi sperimentali hanno osservato che le notifiche possono rallentare o peggiorare la prestazione in compiti che richiedono attenzione, con effetti particolarmente visibili in alcuni gruppi di adolescenti. I risultati non significano che ogni notifica provochi necessariamente un danno, ma mostrano che le interruzioni hanno un costo cognitivo.¹

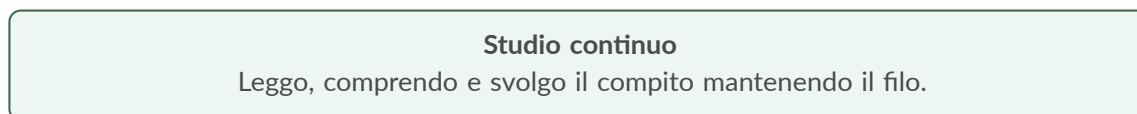
SENZA INTERRUZIONI**CON UN'INTERRUZIONE**

Figura C.2: Dopo un'interruzione, prima di riprendere il compito occorre ritrovare il punto raggiunto. Lo schema mostra la sequenza delle attività, non la loro durata.

¹Si vedano, tra gli altri, Stothart, Mitchum e Yehnert, *The attentional cost of receiving a cell phone notification*, 2015, e Whiting e Murdock, *Notification alert!*, 2021, DOI: [10.1177/17470218211041851](https://doi.org/10.1177/17470218211041851).

C.2.1 Interruzione e ripresa del compito

Quando si passa da un'attività all'altra, il primo compito non scompare immediatamente dalla mente. Al ritorno è necessario ricordare:

- che cosa si stava facendo;
- quale passaggio era già stato completato;
- quale informazione doveva essere mantenuta in memoria;
- quale decisione doveva essere presa successivamente.

Questo lavoro di ricostruzione può essere breve, ma ripetuto molte volte consuma tempo e aumenta la possibilità di errori. Per attività semplici il costo può essere quasi impercettibile; per un testo complesso, un esercizio o un programma può diventare molto più evidente.

Definizione: attenzione frammentata

Si parla di **attenzione frammentata** quando un'attività viene interrotta ripetutamente e la concentrazione deve essere ricostruita più volte, invece di essere mantenuta per un periodo continuo.

C.2.2 Il falso mito del multitasking

Nel linguaggio quotidiano il termine *multitasking* indica lo svolgimento contemporaneo di più attività. Alcune combinazioni sono possibili, soprattutto quando una delle azioni è automatica. Tuttavia, quando due compiti richiedono entrambi lettura, ragionamento o decisioni, spesso non vengono svolti realmente nello stesso momento: l'attenzione passa rapidamente dall'uno all'altro.

Per esempio, seguire una spiegazione e rispondere a una conversazione scritta richiedono entrambi di comprendere parole e mantenere informazioni in memoria. Il passaggio frequente può far perdere parti della spiegazione e rendere più superficiale la risposta.

Esempio: due situazioni differenti

Compatibilità maggiore: ascoltare musica strumentale a volume moderato mentre si riordina il materiale.

Conflitto maggiore: leggere un testo nuovo mentre si risponde continuamente a messaggi e si osservano brevi video.

C.2.3 Perché è facile continuare

Molte applicazioni sono progettate per rendere semplice la prosecuzione dell'uso. Tra le funzioni più comuni si trovano:

- notifiche che segnalano nuovi contenuti;
- scorrimento continuo senza una pagina finale;
- riproduzione automatica del contenuto successivo;

- suggerimenti personalizzati in base alle attività precedenti;
- conteggi, serie giornaliere e indicatori di novità;
- messaggi che mostrano immediatamente nuovi aggiornamenti.

Queste funzioni non sono necessariamente dannose e possono migliorare l'esperienza. Tuttavia, riducono i momenti naturali nei quali l'utente potrebbe fermarsi e decidere se proseguire.

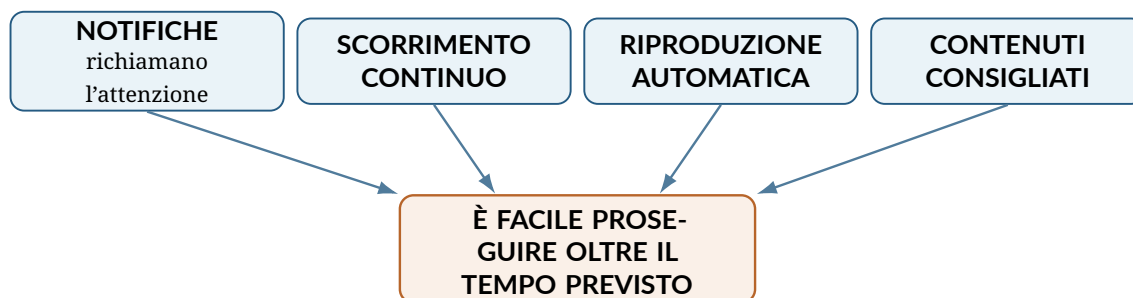


Figura C.3: Alcune funzioni riducono le interruzioni naturali e facilitano la prosecuzione dell'uso.

Approfondimento: economia dell'attenzione

Per molte piattaforme il tempo e l'interazione degli utenti hanno valore economico: permettono di mostrare contenuti, pubblicità e servizi. Per questo alcune interfacce cercano di ridurre l'abbandono e invitano a proseguire. Comprendere questo modello non significa rifiutare le piattaforme, ma ricordare che l'obiettivo del servizio può non coincidere sempre con l'obiettivo dell'utente.

C.2.4 Proteggere il tempo di studio

Una strategia efficace non consiste necessariamente nello spegnere ogni dispositivo per tutta la giornata. È più realistico creare periodi nei quali le interruzioni vengono ridotte intenzionalmente.

Un possibile metodo consiste nel definire un compito concreto, preparare il materiale, silenziare le notifiche non urgenti, tenere lo smartphone fuori dal campo visivo e controllare i messaggi soltanto durante una pausa scelta. La durata dell'intervallo dipende dall'attività: l'elemento importante è che inizio e fine siano decisi prima.

Attività guidata: una sessione di studio osservata

Durante un periodo di studio annotare soltanto le interruzioni digitali e distinguere quelle urgenti da quelle rinviabili. Scegliere poi una modifica realistica da sperimentare.

Esercizi: attenzione e notifiche

1. Spiegare perché una notifica può interrompere anche senza essere aperta.
2. Indicare quattro funzioni che facilitano la prosecuzione dell'uso di un'applicazione.
3. Progettare una sessione di studio di 30 minuti con poche interruzioni.

C.3 Sonno, postura e benessere fisico

L'uso delle tecnologie coinvolge il corpo. La posizione mantenuta, la distanza dallo schermo, il tempo trascorso seduti, il volume delle cuffie e l'orario di utilizzo possono influenzare comfort, affaticamento e riposo. Non si tratta di stabilire che gli schermi siano sempre dannosi, ma di evitare esposizioni prolungate e abitudini che interferiscono con altre necessità.

C.3.1 Dispositivi e sonno

La relazione tra uso digitale e sonno è complessa. Conta il tempo complessivo, ma contano anche l'orario, il luogo e il tipo di attività. Le ricerche sugli adolescenti mostrano associazioni tra uso serale o notturno dei dispositivi e orari di addormentamento più tardivi, sonno più breve o qualità peggiore, soprattutto quando l'attività prosegue dopo essersi messi a letto o è molto interattiva. Non tutti gli studi rilevano effetti della stessa intensità e non è corretto attribuire ogni problema soltanto alla luce dello schermo.²

I dispositivi possono interferire con il riposo in diversi modi:

- spostano più avanti il momento nel quale si decide di dormire;
- offrono contenuti stimolanti che rendono difficile interrompere;
- notifiche, suoni o vibrazioni possono disturbare durante la notte;
- conversazioni e attività sociali possono creare la sensazione di dover restare disponibili;
- l'uso a letto associa il luogo del riposo a un'attività continua.

Approfondimento: non esiste una sola causa

Dormire poco può dipendere da orari scolastici, impegni, stress, ambiente, abitudini e condizioni personali. Il dispositivo è uno degli elementi modificabili, ma non deve diventare una spiegazione automatica per ogni difficoltà. In caso di problemi persistenti è opportuno parlarne con un adulto e, quando necessario, con un professionista sanitario.

²Per una sintesi recente si vedano la revisione di Fiore et al., *Digital Devices Use and Sleep in Adolescents*, *International Journal of Environmental Research and Public Health*, 22(10), 1517, 2025, DOI: [10.3390/ijerph22101517](https://doi.org/10.3390/ijerph22101517), e il consenso della National Sleep Foundation, 2024. Uno studio con misure dirette ha evidenziato soprattutto l'effetto dell'uso una volta a letto: Brosnan et al., 2024.

Per ridurre le interferenze è utile terminare per tempo le attività più coinvolgenti, silenziare le notifiche notturne non necessarie, evitare di tenere il telefono nel letto e osservare se la modifica scelta migliora davvero il riposo.

C.3.2 Affaticamento visivo

L'uso prolungato di schermi può essere accompagnato da secchezza, bruciore, visione temporaneamente sfocata, mal di testa o difficoltà a mantenere la messa a fuoco. Questo insieme di disturbi viene spesso indicato come **affaticamento visivo digitale**. Durante l'osservazione ravvicinata si tende inoltre a battere meno frequentemente le palpebre.

Per ridurre l'affaticamento è utile mantenere una distanza comoda, aumentare la dimensione del testo invece di sporgersi, limitare i riflessi, ricordarsi di ammiccare e fare pause brevi guardando anche verso oggetti lontani.

La regola "20-20-20" suggerisce ogni 20 minuti di guardare per circa 20 secondi un oggetto distante almeno 20 piedi, cioè circa 6 metri. È una strategia semplice, non una misura medica universale; studi sperimentali indicano che pause frequenti possono ridurre alcuni sintomi, ma non sostituiscono una valutazione in caso di disturbi persistenti.³

Errore comune: attribuire ogni disturbo alla luce blu

Il disagio durante l'uso degli schermi non dipende da un solo fattore. Durata, distanza, dimensione del testo, riflessi, postura, riduzione dell'ammicciamento e problemi visivi non corretti possono avere un ruolo. I filtri per la luce blu non sostituiscono pause, illuminazione adeguata e corretta ergonomia.

C.3.3 Postura e organizzazione della postazione

Non esiste una posizione perfetta da mantenere immobile per molte ore. Una buona postazione permette però di evitare posture forzate e di cambiare posizione con facilità. Le indicazioni ergonomiche più comuni suggeriscono di tenere il monitor davanti al corpo, con il bordo superiore all'altezza degli occhi o poco sotto, schiena e piedi sostenuti, spalle rilassate e polsi in posizione neutra.⁴

³American Academy of Ophthalmology, *Computer Vision Syndrome*; Talens-Estarells et al., *The effects of breaks on digital eye strain*, pubblicato online nel 2022 e in fascicolo nel 2023, DOI: [10.1016/j.clae.2022.101744](https://doi.org/10.1016/j.clae.2022.101744).

⁴Occupational Safety and Health Administration, *Computer Workstations eTool*, sezioni su monitor, tastiere e posture.

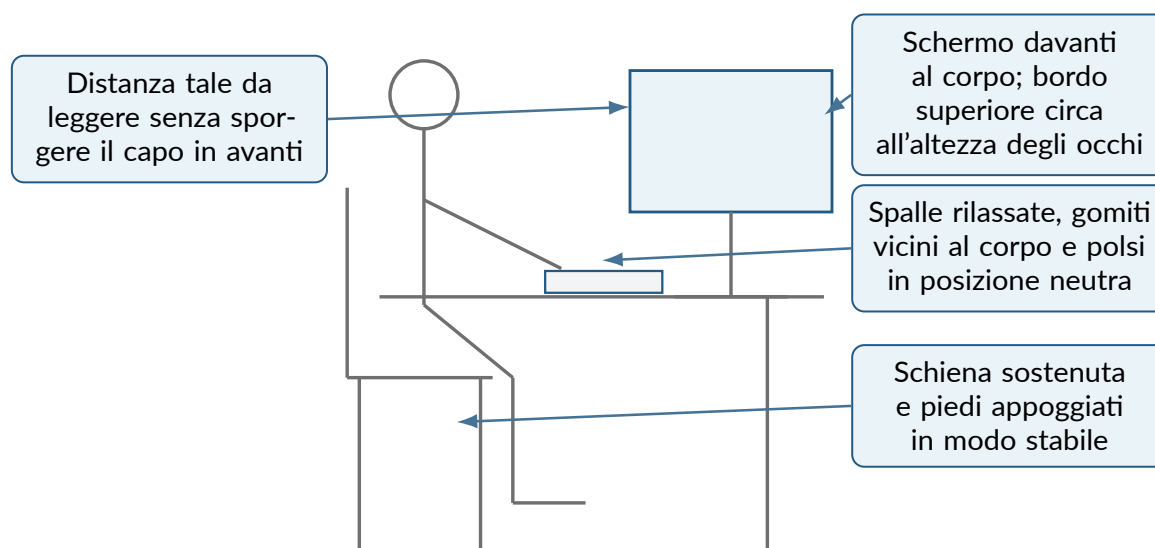


Figura C.4: Una postazione regolata correttamente riduce la necessità di inclinare il capo, sollevare le spalle o piegare i polsi.

C.3.4 Sedentarietà e movimento

Un'attività digitale può essere utile e, nello stesso tempo, richiedere di restare seduti a lungo. Le linee guida dell'Organizzazione mondiale della sanità raccomandano agli adolescenti di limitare i periodi sedentari e ricordano che qualsiasi quantità di movimento è preferibile all'assenza totale di attività.⁵

Durante lo studio è quindi opportuno alzarsi nelle pause, cambiare posizione, alternare compiti digitali e non digitali e mantenere nella giornata spazi per movimento o attività all'aperto.

C.3.5 Cuffie e ascolto sicuro

Il rischio per l'udito dipende sia dal volume sia dalla durata dell'esposizione. Un suono più intenso può essere ascoltato in sicurezza per un tempo molto più breve. L'Organizzazione mondiale della sanità suggerisce, come indicazione pratica, di mantenere il volume dei dispositivi personali sotto circa il 60% del massimo, utilizzare cuffie ben adattate e concedere pause all'udito.⁶

In ambienti rumorosi si tende ad alzare il volume; cuffie ben isolate possono ridurre questa necessità. Ronzii persistenti o difficoltà di ascolto richiedono il confronto con un adulto e una valutazione professionale.

Attività guidata: una modifica realistica

Controllare postazione, pause, notifiche notturne e volume delle cuffie. Scegliere una sola modifica realizzabile e osservarne l'effetto per alcuni giorni.

⁵World Health Organization, *Guidelines on physical activity and sedentary behaviour*, 2020.

⁶World Health Organization, *Deafness and hearing loss: Safe listening*, aggiornamento 2026.

Esercizi: benessere fisico

1. Indicare tre modi nei quali l'uso digitale può interferire con il sonno.
2. Elencare quattro accorgimenti per ridurre l'affaticamento visivo.
3. Descrivere una postazione corretta e proporre una routine serale realistica.

Le indicazioni presentate sono generali e non sostituiscono valutazioni mediche o psicologiche.

C.4 Social network, identità e confronto sociale

I social network permettono di mantenere relazioni, condividere interessi, trovare comunità e partecipare alla vita pubblica. Allo stesso tempo, ciò che viene mostrato in un profilo non coincide con l'intera esperienza di una persona. Ogni pubblicazione è il risultato di una selezione: si sceglie un momento, un'immagine, una frase e il modo nel quale presentarli.

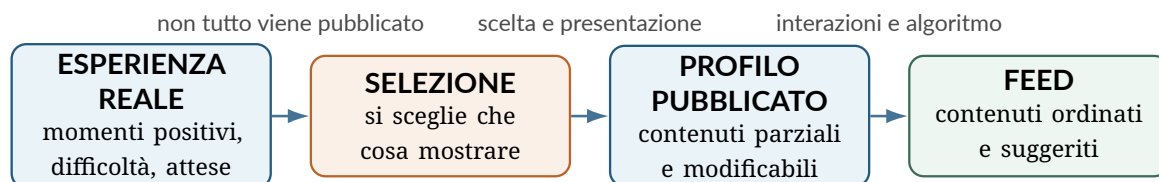


Figura C.5: Il contenuto che compare in un feed deriva da più selezioni e non rappresenta l'intera realtà.

C.4.1 Il profilo è una rappresentazione parziale

Una persona può pubblicare soprattutto successi, viaggi, risultati o immagini curate e lasciare fuori attese, difficoltà e giornate ordinarie. Filtri, inquadrature e modifiche possono rendere ancora più distante il contenuto dalla situazione originale. Questo non significa che ogni pubblicazione sia falsa: significa che è **parziale**.

Quando ci si confronta con molti profili, si rischia di mettere a confronto la propria vita completa, con momenti positivi e negativi, con una raccolta dei momenti selezionati dagli altri. Il confronto può diventare particolarmente delicato quando riguarda aspetto fisico, popolarità, risultati scolastici o stile di vita. L'American Psychological Association invita a limitare il confronto sociale centrato sull'aspetto e a sviluppare la capacità di interrogare accuratezza e rappresentatività dei contenuti.⁷

Errore comune: considerare il feed una fotografia neutrale della realtà

Un feed non è un campione casuale di ciò che accade nel mondo. Contiene ciò che le persone decidono di pubblicare e ciò che la piattaforma decide di mostrare. Per interpretarlo è necessario chiedersi che cosa manca e perché quel contenuto è stato selezionato.

⁷American Psychological Association, *Health Advisory on Social Media Use in Adolescence*, 2023.

C.4.2 Reazioni, visualizzazioni e ricerca di approvazione

Molte piattaforme rendono visibili numeri come visualizzazioni, reazioni, commenti e follower. Questi indicatori possono essere utili per capire la diffusione di un contenuto, ma non misurano il valore della persona che lo ha pubblicato né la qualità delle sue relazioni.

L'attesa di una risposta può spingere a controllare frequentemente il telefono o a modificare ciò che si pubblica per ottenere maggiore approvazione. Una reazione scarsa può essere interpretata come un giudizio personale, anche quando dipende dall'orario, dall'algoritmo o dal fatto che molte persone non abbiano visto il contenuto.

Definizione: confronto sociale

Il **confronto sociale** è il processo mediante il quale una persona valuta caratteristiche, risultati o comportamenti mettendoli in relazione con quelli degli altri. Può essere utile, ma diventa poco realistico quando si confrontano esperienze complete con rappresentazioni selezionate.

C.4.3 Perché due persone vedono feed diversi

Le piattaforme devono ordinare una quantità di contenuti molto superiore a quella che una persona potrebbe osservare. Per questo utilizzano sistemi di raccomandazione che tengono conto, tra le altre cose, di interazioni precedenti, tempo di visualizzazione, profili seguiti e contenuti simili.

La personalizzazione può rendere il servizio più utile, ma può anche riproporre molte volte lo stesso tipo di messaggio. Fermarsi su un video per curiosità non significa necessariamente desiderare che l'intero feed venga riempito con contenuti analoghi. È quindi importante utilizzare consapevolmente funzioni come “non mi interessa”, smettere di seguire un profilo o cercare fonti differenti.

L'Organizzazione mondiale della sanità ricorda che gli ambienti digitali possono connettere e informare, ma anche amplificare ideali irrealistici, contenuti dannosi e raccolta invasiva di dati, soprattutto quando la raccomandazione algoritmica è poco trasparente.

8

Attività guidata: leggere un feed con spirito critico

Scegliere, senza indicare account personali, tre contenuti di uno stesso tema. Per ciascuno annotare: che cosa mostra, che cosa potrebbe essere rimasto fuori, quale emozione cerca di suscitare e perché potrebbe essere stato consigliato.

Esercizi: social network e confronto

1. Spiegare perché un profilo online rappresenta soltanto una parte dell'esperienza di una persona.
2. Indicare due motivi per cui il numero di reazioni non misura il valore di un contenuto o di chi lo pubblica.

⁸World Health Organization, Regional Office for Europe, *Mental health: digital mental health of adolescents*, 2026.

Esercizi: social network e confronto (continua)

3. Descrivere in modo semplice come un sistema di raccomandazione può personalizzare un feed.
4. Proporre due azioni per rendere più vario o utile ciò che compare nella propria esperienza online.

C.5 Uso problematico, relazioni e comportamenti dannosi

Utilizzare spesso una tecnologia non significa automaticamente aver sviluppato una dipendenza. Il tempo trascorso online può essere elevato per studio, relazioni, creatività o attività organizzate. Diventa più importante osservare se la persona riesce a scegliere e interrompere, e se l'uso interferisce ripetutamente con sonno, studio, movimento, relazioni o altre attività necessarie.

C.5.1 Riconoscere un uso problematico

Alcuni segnali meritano attenzione quando si presentano con continuità e producono conseguenze negative:

- difficoltà a interrompere nonostante l'intenzione di farlo;
- utilizzo molto più lungo del previsto;
- trascurare compiti, riposo, sport o relazioni;
- irritazione intensa quando non è possibile collegarsi;
- controllo quasi automatico di notifiche e aggiornamenti;
- tentativi ripetuti di ridurre l'uso che non producono alcun cambiamento;
- conflitti frequenti causati dal comportamento digitale.

Nel rapporto HBSC 2021/2022 dell'Organizzazione mondiale della sanità, l'11% degli adolescenti intervistati in 44 paesi e regioni mostrava segnali di uso problematico dei social media. Il dato descrive un gruppo che riferisce difficoltà di controllo e conseguenze negative; non permette di diagnosticare un singolo studente osservando soltanto il numero di ore.⁹

Definizione: uso problematico

Si parla di **uso problematico** quando il comportamento digitale diventa difficile da controllare e interferisce in modo significativo con attività, responsabilità o benessere. Una diagnosi clinica non può essere ricavata da un'autovalutazione scolastica e richiede professionisti qualificati.

⁹World Health Organization, Regional Office for Europe, *A focus on adolescent social media use and gaming in Europe, central Asia and Canada*, 2024.

Errore comune: etichettare una persona in base al tempo

Dire che una persona è “dipendente” soltanto perché usa spesso un dispositivo è una semplificazione. Occorre considerare scopo, capacità di interrompere, conseguenze, durata del problema e situazione individuale.

C.5.2 Le azioni online hanno effetti reali

Una fotografia inoltrata senza consenso, una battuta ripetuta in un gruppo o un commento pubblico possono raggiungere rapidamente molte persone. La distanza dello schermo può far dimenticare tono, contesto ed effetti emotivi. Inoltre, un contenuto può essere copiato o fotografato prima che venga cancellato.

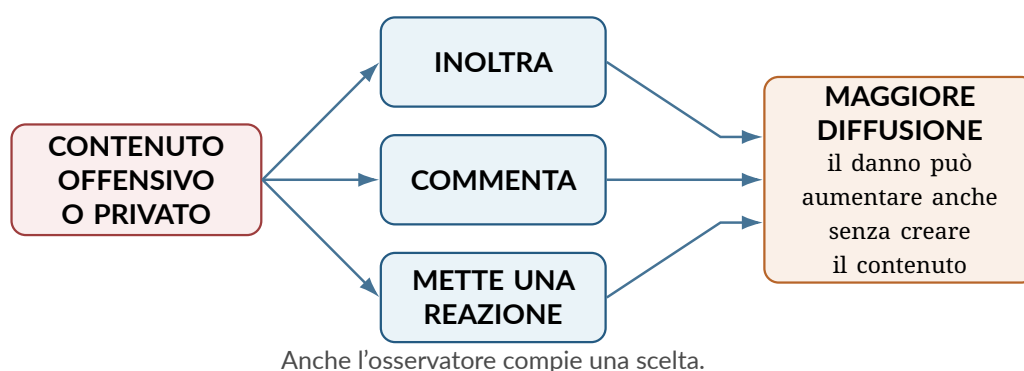


Figura C.6: Inoltrare, commentare o reagire può aumentare la diffusione di un contenuto dannoso.

Chi osserva non è necessariamente responsabile del contenuto iniziale, ma può scegliere se amplificarlo, ignorarlo, segnalarlo o sostenere la persona coinvolta. Anche una reazione apparentemente ironica può incoraggiare la prosecuzione del comportamento.

C.5.3 Cyberbullismo ed esclusione digitale

Il **cyberbullismo** comprende comportamenti aggressivi o umilianti realizzati mediante strumenti digitali, come messaggi offensivi ripetuti, diffusione di immagini, esclusione intenzionale, impersonificazione o creazione di contenuti contro una persona. L'ambiente online può rendere il danno più persistente perché il contenuto resta accessibile, viene ricondiviso e raggiunge la vittima anche fuori dalla scuola.

Nel rapporto HBSC 2021/2022 circa il 15% degli adolescenti dichiarava di aver subito cyberbullismo, una quota vicina a un adolescente su sei.¹⁰

Davanti a un episodio è preferibile non reagire impulsivamente né affrontarlo da soli. È utile conservare prove, usare gli strumenti di blocco e segnalazione e rivolgersi rapidamente a una persona adulta affidabile.

¹⁰World Health Organization, Regional Office for Europe, *A focus on adolescent peer violence and bullying in Europe, central Asia and Canada*, 2024.

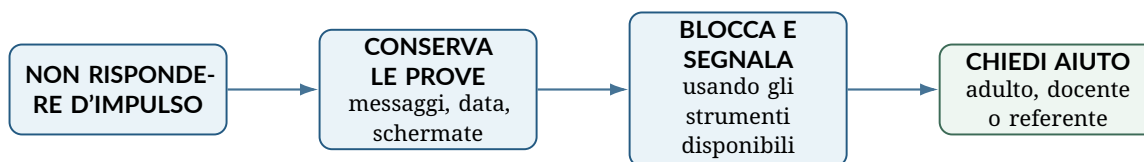


Figura C.7: Una procedura essenziale davanti a un contenuto offensivo, minaccioso o diffuso senza consenso.

Conservare le prove non significa continuare a diffondere il contenuto. È sufficiente mantenere messaggi, nomi degli account, data, ora e schermate utili per spiegare l'accaduto a chi può intervenire. UNICEF raccomanda, tra le azioni possibili, di chiedere aiuto, bloccare e segnalare l'autore e documentare ciò che è successo. ¹¹

Attività guidata: che cosa può fare chi osserva

Considerare un gruppo nel quale viene condivisa una fotografia umiliante. Distinguere le possibili azioni di chi osserva in quattro categorie: amplificare il danno, restare passivo, sostenere la persona coinvolta, chiedere l'intervento di un adulto.

Esercizi: uso problematico e relazioni

1. Indicare perché il solo tempo di utilizzo non è sufficiente per riconoscere un uso problematico.
2. Elencare quattro segnali che possono indicare una perdita di controllo significativa.
3. Spiegare come una semplice reazione possa contribuire alla diffusione di un contenuto dannoso.
4. Descrivere la procedura da seguire davanti a un episodio di cyberbullismo.

C.6 Costruire abitudini digitali equilibrate

Non esiste una quantità di tecnologia adatta in modo identico a tutte le persone e a tutte le giornate. Una strategia efficace deve considerare età, attività, impegni, relazioni e difficoltà personali. Le modifiche troppo rigide vengono spesso abbandonate; è più utile intervenire su un comportamento preciso e verificarne l'effetto.

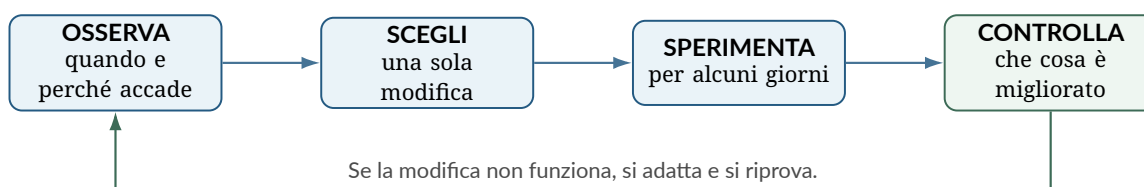


Figura C.8: Un'abitudine può essere modificata attraverso osservazione, sperimentazione e verifica.

¹¹UNICEF, *Cyberbullying: what is it and how to stop it*, indicazioni per adolescenti.

C.6.1 Modificare l'ambiente, non soltanto affidarsi alla volontà

È più difficile ignorare una distrazione quando è sempre visibile, produce suoni e richiede una decisione ogni pochi minuti. Per questo molte strategie agiscono sull'ambiente:

- disattivare notifiche non necessarie;
- tenere il telefono fuori dal campo visivo durante un'attività impegnativa;
- stabilire momenti precisi per controllare messaggi e aggiornamenti;
- rimuovere dalla schermata iniziale le applicazioni aperte automaticamente;
- utilizzare modalità che riducono le interruzioni durante studio e riposo;
- lasciare il dispositivo lontano dal letto durante la notte;
- alternare attività sedentarie, movimento e relazioni in presenza.

Queste azioni non eliminano la libertà di scelta: riducono il numero di richiami automatici e permettono di decidere con maggiore calma.

C.6.2 Scegliere una modifica osservabile

Obiettivi generici come “usare meno il telefono” sono difficili da verificare. È preferibile formulare una modifica concreta:

durante 30 minuti di studio lascio il telefono nello zaino

oppure:

dopo le 22 disattivo le notifiche non urgenti

Dopo alcuni giorni si può controllare se la modifica ha migliorato concentrazione, puntualità, sonno o tranquillità. Se non funziona, non significa che ogni tentativo sia inutile: può essere necessario cambiare durata, contesto o strategia.

Approfondimento: autonomia e aiuto

Essere autonomi non significa dover risolvere ogni difficoltà da soli. Quando l'uso digitale provoca sofferenza, conflitti persistenti, perdita di sonno o isolamento, chiedere aiuto a familiari, docenti, educatori o professionisti è una scelta responsabile.

C.6.3 Un piano personale di benessere digitale

Un breve piano può includere:

1. un'attività digitale utile da mantenere;
2. un comportamento automatico da ridurre;
3. una fonte di interruzione da modificare;

4. un momento della giornata senza notifiche;
5. una strategia per il sonno o il movimento;
6. una persona alla quale rivolgersi in caso di difficoltà.

Il piano non deve essere pubblico e non deve contenere informazioni private. Può essere utilizzato come strumento di osservazione personale e rivisto dopo una settimana.

Attività guidata: esperimento di sette giorni

Scegliere una sola modifica, descrivere il comportamento di partenza e stabilire come valutarne l'effetto. Al termine annotare ciò che è cambiato, ciò che è rimasto difficile e l'eventuale adattamento da provare.

Esercizi: abitudini equilibrate

1. Spiegare perché modificare l'ambiente può essere più efficace che affidarsi soltanto alla forza di volontà.
2. Trasformare l'obiettivo generico "distrarmi meno" in un comportamento concreto e verificabile.
3. Preparare un piano personale usando i sei punti proposti, senza inserire dati riservati.
4. Progettare un breve decalogo di classe per un uso rispettoso e sostenibile delle tecnologie.

Curiosità: la tecnologia come oggetto di progettazione

Le applicazioni non sono ambienti naturali: ogni pulsante, notifica e suggerimento è il risultato di una scelta di progettazione. Anche l'utente può progettare il proprio ambiente digitale, scegliendo impostazioni, tempi e strumenti coerenti con i propri obiettivi.

Glossario

Il glossario raccoglie i principali termini utilizzati nei capitoli e nelle appendici. Le definizioni sono sintetiche e servono come riferimento rapido; per esempi, figure e spiegazioni complete è opportuno consultare le sezioni del manuale nelle quali il termine viene introdotto.

A

Abbonamento software. Modello di distribuzione nel quale il diritto di usare un programma dipende dal pagamento periodico di una quota.

Access point. Dispositivo che permette ai dispositivi wireless di collegarsi a una rete locale, spesso mettendo in comunicazione Wi-Fi ed Ethernet.

Accumulatore. Variabile che raccoglie progressivamente un risultato durante un ciclo, per esempio una somma. Viene normalmente inizializzata prima delle iterazioni.

Affidabilità. Capacità di un sistema o di una rete di continuare a funzionare correttamente e di ridurre la probabilità di guasti o interruzioni.

Affaticamento visivo digitale. Insieme di disturbi temporanei, come secchezza, bruciore, mal di testa o difficoltà di messa a fuoco, che possono comparire dopo un uso prolungato degli schermi.

Algoritmo. Sequenza finita, ordinata e non ambigua di istruzioni che permette di risolvere un problema o raggiungere un risultato.

Algoritmo sequenziale. Algoritmo nel quale le istruzioni vengono eseguite una dopo l'altra, nell'ordine in cui sono state scritte, senza decisioni.

ALU. Parte della CPU che esegue operazioni aritmetiche e logiche. Il nome deriva da *Arithmetic Logic Unit*.

Anodo. Terminale del LED che, nei collegamenti più comuni, viene orientato verso il potenziale positivo.

Arduino. Piattaforma composta da schede programmabili, ambiente di sviluppo e librerie, utilizzata per realizzare prototipi elettronici.

Arduino IDE. Ambiente di sviluppo usato per scrivere, verificare e caricare gli sketch sulle schede Arduino.

Arduino Uno. Scheda Arduino molto diffusa nella didattica, dotata di microcontrollore, ingressi e uscite digitali e ingressi analogici.

Arrotondamento. Approssimazione di un numero mediante un valore rappresentabile con la precisione scelta. Nell'arrotondamento al più vicino si sceglie il valore meno distante e si dichiara una regola per i casi di parità.

Assegnamento. Istruzione che calcola un valore e lo memorizza in una variabile, sostituendo l'eventuale valore precedente.

Attenzione frammentata. Condizione nella quale un'attività viene interrotta ripetutamente e la concentrazione deve essere ricostruita più volte.

Attuatore. Dispositivo che riceve un comando e produce un effetto fisico, come luce, suono, movimento o calore.

B

Base di un sistema di numerazione. Numero di simboli disponibili e valore che determina i pesi delle posizioni. Il sistema binario ha base 2, il decimale base 10 e l'esadecimale base 16.

Benessere digitale. Condizione nella quale le tecnologie vengono utilizzate in modo compatibile con attenzione, riposo, salute, relazioni e attività quotidiane.

Bilanciamento di carico. Dispositivo o servizio che distribuisce le richieste tra più server per migliorare prestazioni, continuità e affidabilità.

Bit. Unità minima dell'informazione digitale, che può assumere soltanto i valori 0 oppure 1.

Bit frazionario. Cifra binaria dopo la virgola: la posizione k ha peso $2^{-k} = 1/2^k$.

Bit meno significativo. Bit posto nella posizione di peso minore di un numero binario; viene spesso abbreviato con LSB.

Bit più significativo. Bit posto nella posizione di peso maggiore di un numero binario; viene spesso abbreviato con MSB.

BitTorrent. Protocollo di condivisione distribuita nel quale un file viene scambiato tra più peer appartenenti a uno sciame.

Bluetooth. Tecnologia wireless a corto raggio usata per collegare periferiche e dispositivi personali.

bool. Tipo di variabile che può contenere soltanto i valori logici true e false.

Breadboard. Basetta che permette di costruire circuiti temporanei inserendo componenti e cavetti senza usare saldature.

Bus. Canale di comunicazione che trasporta dati, indirizzi o segnali di controllo tra componenti del computer.

Byte. Gruppo di 8 bit. È una delle unità fondamentali usate per misurare dati e capacità di memoria.

C

Cache. Memoria veloce che conserva temporaneamente dati usati di frequente, riducendo i tempi di accesso.

Case. Involucro che contiene e protegge i principali componenti interni di un computer.

Caso limite. Valore posto al confine tra due situazioni, utile per verificare se condizioni e confronti sono stati progettati correttamente.

Catodo. Terminale del LED che, nei collegamenti più comuni, viene orientato verso GND.

Cavo coassiale. Mezzo trasmissivo formato da un conduttore centrale, materiale isolante, schermatura e rivestimento esterno.

Cella. Area geografica coperta da una stazione radio di una rete mobile. Il dispositivo passa da una cella all'altra durante lo spostamento.

Centro dati. Struttura che ospita server, sistemi di archiviazione, apparati di rete, alimentazione e raffreddamento.

Client. Dispositivo o programma che richiede un servizio a un server.

Ciclo. Struttura algoritmica nella quale una freccia riporta l'esecuzione verso un punto precedente, ripetendo un gruppo di istruzioni fino al verificarsi di una condizione o per un numero stabilito di volte.

Ciclo infinito. Ciclo che non raggiunge mai la condizione di arresto e continua a essere eseguito senza terminare.

Client-server. Modello di comunicazione nel quale uno o più client richiedono risorse o servizi forniti da un server.

Codice di stato HTTP. Numero restituito da un server per indicare l'esito di una richiesta HTTP, per esempio successo, reindirizzamento o errore.

Colore RGB a 24 bit. Rappresentazione nella quale rosso, verde e blu usano 8 bit ciascuno, per un totale di 24 bit per pixel.

Commento. Testo inserito nel codice per spiegare il programma a chi lo legge; non viene eseguito dal microcontrollore.

Complemento a 2. Metodo usato per rappresentare numeri interi negativi in binario e permettere addizioni e sottrazioni con gli stessi circuiti.

Contatore. Variabile che registra il numero di iterazioni o la posizione corrente in una ripetizione, spesso aggiornata di una unità alla volta.

Corpo del ciclo. Gruppo di istruzioni eseguito a ogni iterazione di un ciclo.

Compressione. Tecnica che riduce la quantità di dati necessaria per rappresentare un file, con o senza perdita di informazione.

Computer. Elaboratore elettronico digitale programmabile capace di acquisire dati, elaborarli, memorizzarli e comunicare risultati.

Computer per uso generale. Computer progettato per svolgere attività differenti attraverso programmi diversi.

Computer specializzato. Elaboratore progettato principalmente per una funzione specifica o inserito all'interno di un dispositivo.

Concatenazione. Unione di più elementi, per esempio una frase e il valore di una variabile, in un unico output testuale.

Concetto. Idea o informazione essenziale rappresentata da un nodo in una mappa concettuale.

Condizione. Espressione che può risultare vera o falsa e che viene usata per prendere una decisione.

Confronto sociale. Processo con cui una persona valuta caratteristiche, risultati o comportamenti mettendoli in relazione con quelli degli altri.

Connettore. Parte fisica che permette di collegare un cavo o un dispositivo a una porta.

Core. Nucleo di elaborazione presente all'interno di una CPU. Una CPU multicore può eseguire più attività in parallelo.

CPU. Unità centrale di elaborazione che interpreta ed esegue le istruzioni del programma e coordina le altre parti del computer.

Cyberbullismo. Comportamento aggressivo, offensivo o umiliante realizzato mediante strumenti digitali e capace di diffondersi o persistere online.

D

Dato. Rappresentazione grezza di un fatto, una misura, un simbolo o un'osservazione che può essere elaborata.

Decisione. Struttura algoritmica che sceglie quale percorso seguire in base al valore vero o falso di una condizione.

Decodifica. Fase nella quale la CPU interpreta un'istruzione prelevata dalla memoria e determina quali operazioni eseguire.

DHCP. Protocollo che assegna automaticamente ai dispositivi i parametri di rete, come indirizzo IP, gateway e server DNS.

DHT. Tabella hash distribuita usata in alcune reti peer-to-peer per trovare risorse senza dipendere da un unico server centrale.

Diagramma di flusso. Rappresentazione grafica di un algoritmo mediante simboli collegati da frecce.

Dichiarazione. Istruzione che specifica il tipo e il nome di una variabile prima del suo utilizzo nel codice.

Digitale. Sistema che rappresenta informazioni mediante valori discreti; nei computer si usano normalmente 0 e 1.

DisplayPort. Interfaccia digitale usata soprattutto per trasmettere segnali video e audio da un computer a un monitor.

Dispositivo di indirizzamento. Apparato che usa informazioni di indirizzo per inoltrare dati verso la destinazione corretta, come uno switch o un router.

DNS. Sistema che associa nomi leggibili, come quelli dei siti web, agli indirizzi IP necessari alla comunicazione.

Dorsale. Collegamento di rete ad alta capacità che trasporta grandi quantità di traffico tra reti e aree geografiche.

Driver. Software che permette al sistema operativo di comunicare correttamente con una periferica o un componente hardware.

E

Economia dell'attenzione. Modello nel quale il tempo e le interazioni degli utenti hanno valore economico e le piattaforme cercano di mantenere attiva la loro attenzione.

Elaborazione. Trasformazione dei dati attraverso calcoli, confronti, ordinamenti o altre operazioni.

Elettronica. Tecnologia basata sull'uso controllato di segnali e componenti elettrici per elaborare o trasmettere informazioni.

Ethernet. Famiglia di tecnologie usate soprattutto nelle reti locali cablate.

EULA. Contratto di licenza con l'utente finale che stabilisce condizioni e limiti d'uso di un software.

F

Feed. Sequenza personalizzata di contenuti mostrata da una piattaforma, selezionata in base a pubblicazioni, preferenze e sistemi di raccomandazione.

Fibra ottica. Mezzo trasmissivo che usa impulsi luminosi in sottili filamenti di vetro o materiale plastico, offrendo velocità elevate e grande distanza.

Firmware. Software memorizzato in un dispositivo elettronico che ne controlla le funzioni di base.

float. Tipo di variabile usato per rappresentare numeri con parte decimale, con precisione limitata.

Freeware. Software utilizzabile gratuitamente, ma non necessariamente open source e comunque soggetto a una licenza.

FTTC. Collegamento in fibra ottica che arriva fino a un armadio stradale; l'ultimo tratto verso l'edificio usa generalmente rame.

FTTB. Collegamento nel quale la fibra termina in un punto comune dell'edificio; il tratto finale verso le singole unità può utilizzare il rame.

FTTH. Collegamento nel quale la fibra ottica arriva direttamente alla singola abitazione o unità immobiliare dell'utente.

G

GAN. Rete geografica globale che può collegare sistemi distribuiti su scala mondiale.

GIGO. Principio secondo cui dati di ingresso errati o inadeguati producono risultati errati o inadeguati: *Garbage In, Garbage Out*.

GPU. Processore specializzato nell'elaborazione grafica e in calcoli altamente paralleli; può essere integrato o dedicato.

H

Hardware. Insieme delle parti fisiche di un sistema informatico, comprese componenti interne e periferiche.

HDD. Unità di memoria di massa che conserva i dati su dischi magnetici rotanti.

HDMI. Interfaccia digitale che trasporta audio e video tra dispositivi.

HTTP. Protocollo applicativo usato per scambiare richieste e risposte tra browser e server web.

HTTPS. Versione di HTTP protetta mediante TLS, che cifra la comunicazione tra client e server.

Hub. Apparato di rete semplice che inoltra i segnali ricevuti verso tutte le proprie porte, senza scegliere la destinazione.

Identità digitale. Insieme di informazioni, contenuti, comportamenti e tracce che rappresentano una persona negli ambienti digitali.

IMAP. Protocollo usato per consultare e organizzare la posta elettronica mantenendo i messaggi sul server.

Iterazione. Singola esecuzione del corpo di un ciclo.

Incapsulamento. Procedimento con cui ogni livello di una comunicazione aggiunge proprie informazioni di controllo ai dati ricevuti dal livello superiore.

Indirizzo IP. Identificatore numerico assegnato a un'interfaccia di rete per permettere l'instradamento dei dati.

Indirizzo MAC. Identificatore associato a un'interfaccia di rete e usato principalmente nelle comunicazioni all'interno della rete locale.

Indirizzo privato. Indirizzo IP destinato alle reti interne e non instradato direttamente su Internet.

Indirizzo pubblico. Indirizzo IP visibile e instradabile su Internet.

Informazione. Dato interpretato e inserito in un contesto, in modo da assumere significato per chi lo utilizza.

Inizializzazione. Assegnazione del primo valore a una variabile nel momento in cui viene dichiarata.

Input. Dato o segnale acquisito da un programma o da un dispositivo.

int. Tipo di variabile usato per rappresentare numeri interi.

Interfaccia. Punto di contatto che permette a persone, programmi o dispositivi di comunicare e scambiarsi comandi o dati.

Internet. Rete mondiale ottenuta collegando molte reti differenti mediante protocolli comuni.

Internet of Things. Insieme di sensori, elettrodomestici, veicoli e altri oggetti fisici collegati alla rete e capaci di scambiare dati.

IPv4. Versione del protocollo IP che usa indirizzi di 32 bit, normalmente scritti come quattro numeri decimali separati da punti.

IPv6. Versione del protocollo IP che usa indirizzi di 128 bit, progettata per offrire uno spazio di indirizzamento molto più ampio.

ISP. Azienda o ente che fornisce l'accesso a Internet e altri servizi di rete. La sigla significa *Internet Service Provider*.

J

Jack da 3,5 mm. Connettore analogico usato comunemente per cuffie, microfoni e segnali audio.

L

LAN. Rete locale che collega dispositivi in un'area limitata, come un'aula, un edificio o un laboratorio.

LED. Diodo che emette luce quando viene attraversato da corrente nel verso corretto.

Licenza software. Insieme di condizioni che stabilisce come un programma può essere installato, usato, copiato, modificato o distribuito.

Licenza perpetua. Licenza che permette di usare una determinata versione del software senza pagamenti periodici obbligatori.

M

M.2. Formato compatto usato per dispositivi interni, soprattutto unità SSD e moduli di comunicazione.

Magnet link. Collegamento che identifica una risorsa peer-to-peer attraverso informazioni come l'impronta del contenuto, senza indicare un normale file da scaricare.

MAN. Rete metropolitana che collega più reti locali all'interno di una città o di un'area urbana.

Mappa concettuale. Rappresentazione organizzata di concetti e relazioni, nella quale nodi e parole-legame formano proposizioni leggibili.

Mappa mentale. Rappresentazione che sviluppa rami associativi intorno a un tema centrale, spesso con parole, immagini e colori.

Megapixel. Unità pari a un milione di pixel, usata per indicare la risoluzione di immagini e sensori fotografici.

Memoria centrale. Memoria direttamente usata dalla CPU durante l'esecuzione dei programmi; nel modello di von Neumann comprende soprattutto la RAM.

Memoria di massa. Dispositivo che conserva dati e programmi anche quando il computer è spento, come HDD o SSD.

Memorizzazione. Conservazione di dati e programmi in una memoria per poterli usare successivamente.

Metadato. Dato che descrive altri dati, per esempio autore, data, formato, dimensione o posizione di un file.

Mezzo trasmissivo. Supporto attraverso il quale viaggia un segnale di rete, come rame, fibra ottica o onde radio.

Microcontrollore. Circuito integrato che contiene unità di elaborazione, memoria e collegamenti per controllare dispositivi elettronici.

Modello di von Neumann. Architettura nella quale dati e istruzioni sono memorizzati nella stessa memoria e vengono elaborati da una CPU.

Modello RGB. Sistema che rappresenta i colori combinando diverse intensità di rosso, verde e blu.

Monitor. Dispositivo di output che visualizza testi, immagini e interfacce grafiche.

Monitor seriale. Strumento dell'Arduino IDE che mostra i dati testuali inviati dalla scheda attraverso la comunicazione seriale.

Multitasking. Passaggio rapido tra più attività; quando i compiti richiedono attenzione, spesso non vengono svolti davvero nello stesso momento.

N

NAT. Tecnica con cui un router traduce gli indirizzi privati della rete locale in uno o più indirizzi pubblici.

Nodo di rete. Dispositivo o punto di una rete capace di inviare, ricevere o inoltrare dati.

Nodo di una mappa. Elemento grafico che contiene un concetto, collegato ad altri nodi mediante relazioni.

Notifica. Segnale visivo, sonoro o tattile con cui un'applicazione informa l'utente di un nuovo evento o contenuto.

Numero pseudocasuale. Numero prodotto da un algoritmo deterministico che genera una sequenza simile a una sequenza casuale.

NVMe. Protocollo progettato per unità SSD collegate tramite PCI Express, capace di offrire prestazioni elevate.

O

Operatore logico. Simbolo che combina o modifica condizioni; gli operatori principali studiati sono AND && e OR | |.

Operatore modulo. Operatore % che restituisce il resto della divisione intera.

Ottetto. Gruppo di 8 bit. In IPv4 ogni parte dell'indirizzo è rappresentata da un ottetto.

Output. Risultato prodotto da un programma o comando inviato a un dispositivo di uscita.

Overflow. Situazione nella quale un risultato supera l'intervallo di valori rappresentabile con il numero di bit disponibile.

P

P2P. Abbreviazione di *peer-to-peer*, modello nel quale i dispositivi possono offrire e richiedere risorse direttamente.

PAN. Rete personale che collega dispositivi vicini a una singola persona, spesso mediante Bluetooth.

Parola-legame. Espressione posta tra due nodi di una mappa concettuale per precisare il significato della relazione.

Parte frazionaria. Per un numero non negativo, la parte compresa tra 0 incluso e 1 escluso che resta sottraendo la parte intera; in 13,625 è 0,625.

Payware. Software distribuito a pagamento, con condizioni definite dalla licenza.

Peer. Dispositivo o programma che partecipa a una rete peer-to-peer con un ruolo paritario rispetto agli altri partecipanti.

Peer-to-peer. Modello di rete nel quale ogni partecipante può agire sia da client sia da fornitore di risorse.

Periferica. Dispositivo collegato al computer per acquisire dati, comunicare risultati o conservare informazioni.

Periodo di un numero. Gruppo di cifre che si ripete indefinitamente nello sviluppo di un numero, eventualmente dopo alcune cifre iniziali.

Pixel. Elemento più piccolo di un'immagine digitale raster, al quale viene assegnato un valore di colore.

Pixelatura. Effetto per cui i singoli pixel diventano visibili quando un'immagine viene ingrandita oltre la sua risoluzione.

PoE. Tecnologia che permette di trasportare alimentazione elettrica e dati sullo stesso cavo Ethernet. La sigla significa *Power over Ethernet*.

POP3. Protocollo usato per scaricare messaggi di posta elettronica dal server verso il dispositivo dell'utente.

Porta di comunicazione. Collegamento fisico o logico attraverso il quale un dispositivo scambia dati con altri componenti.

Profondità di colore. Numero di bit usati per rappresentare il colore di un pixel; determina quanti colori differenti possono essere codificati.

Programma. Insieme di istruzioni scritte in un linguaggio comprensibile o traducibile dal computer.

Programma memorizzato. Principio secondo cui le istruzioni vengono conservate in memoria insieme ai dati e prelevate durante l'esecuzione.

Protocollo. Insieme di regole condivise che stabilisce formato, ordine e significato dei messaggi scambiati.

Protocollo IP. Protocollo responsabile dell'indirizzamento e dell'inoltro dei pacchetti tra reti.

Pseudocodice. Descrizione testuale e strutturata di un algoritmo, simile a un linguaggio di programmazione ma non legata a una sintassi precisa.

Q

Quartetto di bit. Gruppo di 4 bit, chiamato anche semibyte; corrisponde esattamente a una cifra esadecimale.

R

Raccomandazione algoritmica. Selezione automatica dei contenuti proposta da una piattaforma in base ai dati e alle interazioni dell'utente.

RAM. Memoria centrale veloce e volatile che contiene temporaneamente programmi e dati in uso.

Rete informatica. Insieme di dispositivi collegati per scambiare dati e condividere risorse o servizi.

Rete mobile 3G, 4G e 5G. Generazioni successive di tecnologie cellulari, caratterizzate da prestazioni, capacità e servizi differenti.

Retroazione. Meccanismo nel quale l'effetto prodotto da un sistema modifica l'ambiente e viene misurato nuovamente per aggiornare il controllo.

Richiesta HTTP. Messaggio inviato da un client a un server web per chiedere una risorsa o eseguire un'operazione.

Ridondanza. Presenza di componenti o percorsi alternativi che permette al sistema di continuare a funzionare in caso di guasto.

Riproduzione automatica. Funzione che avvia il contenuto successivo senza richiedere una nuova scelta esplicita dell'utente.

Risoluzione. Numero di pixel che compongono un'immagine o un display, spesso espresso come larghezza per altezza.

Risposta HTTP. Messaggio restituito dal server che contiene un codice di stato, intestazioni e, quando previsto, il contenuto richiesto.

RJ45. Connettore modulare usato comunemente nei collegamenti Ethernet su cavo di rame.

Router. Dispositivo che collega reti differenti e inoltra i pacchetti in base agli indirizzi IP.

Router Wi-Fi. Apparato domestico o scolastico che può combinare funzioni di router, switch, access point, NAT e DHCP.

S

Scheda madre. Circuito principale del computer, sul quale sono collegati CPU, RAM, unità di memoria, porte e altri componenti.

Sciame. Insieme dei peer che stanno condividendo lo stesso contenuto in una rete BitTorrent.

Scorrimento continuo. Interfaccia che carica nuovi contenuti mentre si procede verso il basso, senza una conclusione naturale della pagina.

Sedentarietà. Condizione caratterizzata da lunghi periodi trascorsi seduti o con movimento molto ridotto.

Seed in BitTorrent. Peer che possiede un file completo e continua a condividerlo con altri partecipanti allo sciame.

Seed di un generatore. Valore iniziale usato da un generatore pseudocasuale; lo stesso algoritmo con lo stesso seed produce la stessa sequenza.

Selezione algoritmica. Struttura che sceglie tra percorsi diversi in base a una o più condizioni.

Selezione della scheda. Scelta del modello di Arduino usato per compilare correttamente lo sketch. La porta di comunicazione va selezionata per il caricamento.

Semibyte. Gruppo di 4 bit, equivalente a una cifra esadecimale e chiamato anche quartetto di bit.

Sentinella. Valore speciale che non viene elaborato come dato, ma segnala la fine di una sequenza di input.

Sensore. Componente che rileva una grandezza fisica o uno stato e lo trasforma in un segnale utilizzabile dal sistema.

Sequenza. Ordine con cui le istruzioni di un algoritmo vengono eseguite.

Server. Dispositivo o programma che fornisce risorse o servizi ai client attraverso una rete.

Sistema binario. Sistema di numerazione posizionale in base 2 che usa soltanto le cifre 0 e 1.

Sistema di numerazione posizionale. Sistema nel quale il valore di una cifra dipende sia dal simbolo sia dalla posizione occupata.

Sistema esadecimale. Sistema di numerazione posizionale in base 16 che usa le cifre da 0 a 9 e le lettere da A a F.

Sistema incorporato. Sistema informatico integrato in un oggetto e progettato per svolgere funzioni specifiche.

Sistema operativo. Software di sistema che gestisce hardware, file, utenti, periferiche e avvio delle applicazioni.

Sketch. Nome usato nell'ambiente Arduino per indicare un programma e i file che lo compongono.

SMTP. Protocollo usato per inviare messaggi di posta elettronica tra client e server o tra server.

Social network. Piattaforma digitale che permette di creare profili, pubblicare contenuti e interagire con altre persone o comunità.

Software. Insieme dei programmi, dati e istruzioni che permettono all'hardware di svolgere attività.

Software applicativo. Programma destinato a svolgere attività specifiche per l'utente, come scrivere testi, navigare o elaborare immagini.

Software di sistema. Software che gestisce le risorse del computer e crea l'ambiente necessario alle applicazioni.

Software open source. Software il cui codice sorgente è disponibile secondo una licenza che ne stabilisce uso, modifica e distribuzione.

Software proprietario. Software controllato da un produttore e distribuito con condizioni che normalmente limitano modifica e ridistribuzione.

SSD. Unità di memoria di massa che conserva i dati in memoria elettronica, senza parti meccaniche in movimento.

Starlink. Servizio di accesso a Internet che utilizza una costellazione di satelliti in orbita bassa e terminali installati presso gli utenti.

Switch. Apparato che collega dispositivi in una rete locale e inoltra i dati verso la porta associata all'indirizzo MAC di destinazione.

System on a Chip. Circuito integrato che riunisce nello stesso componente processore e altre funzioni, come grafica, memoria o comunicazione.

T

Tastiera. Dispositivo di input usato per inserire caratteri, numeri e comandi.

TCP. Protocollo di trasporto orientato alla connessione, che fornisce un flusso affidabile e ordinato di byte. I dati possono essere consegnati progressivamente all'applicazione.

TLS. Protocollo crittografico che protegge riservatezza e integrità delle comunicazioni, usato per esempio da HTTPS.

Tipo. Informazione che stabilisce quale categoria di valori può contenere una variabile e quali operazioni sono ammesse.

Topologia a bus. Struttura di rete nella quale i nodi condividono un unico collegamento principale.

Topologia ad albero. Struttura gerarchica composta da più rami collegati a livelli successivi.

Topologia ad anello. Struttura nella quale ogni nodo è collegato al precedente e al successivo formando un percorso chiuso.

Topologia a maglia. Struttura nella quale i nodi possiedono più collegamenti reciproci, offrendo percorsi alternativi.

Topologia a stella. Struttura nella quale tutti i nodi sono collegati a un apparato centrale.

Topologia di rete. Disposizione logica o fisica dei nodi e dei collegamenti di una rete.

Topologia ibrida. Rete ottenuta combinando due o più topologie differenti.

Tracker. Server che aiuta i peer BitTorrent a trovare altri partecipanti allo stesso sciame.

Troncamento. Approssimazione ottenuta eliminando le cifre oltre la posizione scelta, senza modificare quelle conservate.

U

UDP. Protocollo di trasporto senza connessione, che invia datagrammi senza garantire da solo consegna, unicità o ordine. Le prestazioni dipendono anche dall'applicazione e dalla rete.

Unità di controllo. Parte della CPU che coordina l'esecuzione delle istruzioni e controlla il trasferimento dei dati.

URL. Indirizzo che identifica la posizione di una risorsa sul Web e può contenere protocollo, host e percorso.

USB. Standard usato per collegare, alimentare e scambiare dati con numerose periferiche.

USB Type-A. Connettore USB rettangolare tradizionale, inseribile in un solo verso.

USB-C. Connettore USB compatto e reversibile che può trasportare dati, alimentazione e, in alcuni casi, segnali video.

Uso consapevole. Capacità di scegliere una tecnologia in funzione di uno scopo, valutarne gli effetti e interromperne l'uso quando non è più utile o appropriato.

Uso problematico. Uso digitale difficile da controllare che interferisce in modo significativo con attività, responsabilità, relazioni o benessere.

Utility di sistema. Programma che svolge attività di manutenzione, controllo, configurazione o protezione del sistema.

V

Variabile. Spazio di memoria identificato da un nome, il cui valore può cambiare durante l'esecuzione.

Verifica o compilazione. Fase nella quale il codice viene controllato e tradotto; individua errori sintattici ma non garantisce la correttezza logica.

Volatile. Proprietà di una memoria che perde il contenuto quando viene interrotta l'alimentazione, come avviene normalmente per la RAM.

W

WAN. Rete geografica che collega dispositivi e reti locali su distanze molto ampie.

Web. Servizio di Internet basato su pagine e risorse collegate, normalmente consultate mediante browser e protocolli HTTP o HTTPS.

Wi-Fi. Famiglia di tecnologie wireless usate per collegare dispositivi a una rete locale tramite onde radio.

Indice analitico

L'indice analitico rimanda alle pagine nelle quali i concetti vengono introdotti, spiegati o applicati. Per una definizione sintetica è possibile consultare anche il glossario.

A

Abbonamento software 32
Access point 116
Accumulatore 216
Affaticamento visivo digitale 302
Affidabilità 122
Algoritmo 180, 181
Algoritmo sequenziale 195
ALU 16
Anodo 250
Arduino 246
Arduino IDE 253
Arduino Uno 243
Arrotondamento 58
Assegnamento 185, 190
Attenzione frammentata 299
Attuatore 245

B

Base di un sistema di numerazione 49
Benessere digitale 295
Bilanciatore di carico 168
Binario
 numeri con la virgola 54
Bit 42
Bit frazionari 58
Bit meno significativo 61
Bit più significativo 61
BitTorrent 171
Bluetooth 116
bool 260
Breadboard 250
Bus 15, 25
Byte 43

C

Cache 16, 162
Case 14
Caso limite 206
Catodo 250
Cavo coassiale 111
Cella 107
Centro dati 168
Ciclo 215
Ciclo infinito 216
Client 166
Client-server 165, 166
Codice di stato HTTP 160

Colore RGB a 24 bit 83
Commento 254
Complemento a 2 67
Compressione 93
Computer 4, 6, 284
Computer per uso generale 4
Computer specializzato 5
Concatenazione 186
Concetto 278
Condizione 201
Confronto sociale 305
Connettore 21
Contatore 216
Core 16
Corpo del ciclo 215
CPU 15
Cyberbullismo 307

D

Dato 9
Decisione 200
Decodifica 16
DHCP 148
DHT 172
Diagramma di flusso 179, 188
Dichiarazione 259
Digitale 6
DisplayPort 22
Dispositivo di indirizzamento 124
DNS 162
Dorsale 106
Driver 28

E

Economia dell'attenzione 300
Elaborazione 6, 182
Elettronica 6
Errore off-by-one 223
Ethernet 23
EULA 30

F

Feed 304
Fibra ottica 114
Firmware 28
float 260
Freeware 32
FTTB 115
FTTC 115
FTTH 115

G

GAN 142
GIGO 12
GPU 19

H

Hardware 13
HDD 17
HDMI 22
HTTP 159
HTTPS 161
Hub 124, 125

I

Identità digitale 304
IMAP 156
Incapsulamento 152
Indirizzo IP 127
Indirizzo MAC 126
Indirizzo privato 146
Indirizzo pubblico 146
Informazione 9
Inizializzazione 259
Input 182, 186
int 260
Interfaccia 21
Internet 105
Internet of Things 146
IPv4 145
IPv6 148
ISP 105
Iterazione 215

J

Jack da 3,5 mm 23

L

LAN 140
LED 250
Licenza perpetua 32
Licenza software 30

M

M.2 18
Magnet link 172
MAN 141
Mappa concettuale 278
Mappa mentale 280
Megapixel 89
Memoria centrale 25
Memoria di massa 17
Memorizzazione 6
Metadato 11
Mezzo trasmissivo 111
Microcontrollore 246
Modello di von Neumann 24

Modello RGB 83
Moltiplicazioni successive per 2 55
Monitor 14
Monitor seriale 262
Multitasking 299

N

NAT 147
Nodo
 di rete 130
 mappa concettuale 277
Notifica 298
Numero pseudocasuale 226
NVMe 18

O

Operatore logico 207
Operatore modulo 212
Ottetto 145
Output 182, 186
Overflow 63, 74

P

P2P 170
PAN 139
Parola-legame 278
Payware 32
Peer 170
Peer-to-peer 165, 170
Periferica 284
Pixel 88
Pixelatura 89
PoE 127
POP3 156
Porta di comunicazione 21
Profondità di colore 90
Programma 8, 181
Programma memorizzato 24, 25
Protocollo 21
Protocollo IP 144
Pseudocodice 188

Q

Quartetto di bit 77

R

Raccomandazione algoritmica 305
RAM 17
Rete informatica 108
Rete mobile 3G, 4G e 5G 118
Retroazione 246
Richiesta HTTP 159
Ridondanza 135
Riproduzione automatica 299
Risoluzione 89
Risposta HTTP 159

RJ45 23, 112
Router 124, 127
Router Wi-Fi 105

S

Scheda madre 15
Sciame 171
Scorrimento continuo 299
Sedentarietà 303
Seed
 BitTorrent 172
 generatore pseudocasuale 226
Selezione
 scheda Arduino 254
 struttura algoritmica 200
Semibyte 77
Sensore 244
Sentinella 216
Sequenza 195
Server 107, 166
Sistema binario 49
Sistema di numerazione posizionale 48
Sistema esadecimale 76
Sistema incorporato 4
Sistema operativo 28
Sketch 253
SMTP 155
Social network 304
Software 27
Software applicativo 29
Software di sistema 28
Software open source 30
Software proprietario 31
SSD 18
Starlink 120
Switch 124, 126
System on a Chip 26

T

Tastiera 13
TCP 153
Tipo 258
TLS 161
Topologia a bus 130
Topologia a maglia 134
Topologia a stella 132
Topologia ad albero 133
Topologia ad anello 131
Topologia di rete 130
Topologia ibrida 135
Tracker 172
Troncamento 58

U

UDP 154
Unità di controllo 15
URL 158
USB 21
USB Type-A 21
USB-C 21
Uso consapevole 297
Uso problematico 306
Utility di sistema 28

V

Variabile 184
Verifica o compilazione 254
Volatile 17

W

WAN 141
Web 158
Wi-Fi 104, 116