1. Calcolatori e programmi

UN PROGRAMMA è un insieme di istruzioni che indica a un calcolatore cosa deve fare. Per scrivere programmi, è quindi importante avere una visione sufficientemente chiara dell'organizzazione di un calcolatore. Lo scopo di questo capitolo, che si apre con un'introduzione al metodo informatico, è proprio quello di fornire una rassegna dell'hardware e del software di un calcolatore, di introdurre formalmente il concetto di algoritmo e la sua rappresentazione grafica mediante diagrammi di flusso e di dare una visione di insieme dei diversi linguaggi di programmazione e di come un programma scritto in un linguaggio ad alto livello venga tradotto nell'unico linguaggio comprensibile a un calcolatore (ovvero il linguaggio macchina). La maggior parte di quello che verrà detto in questo capitolo si applica alla programmazione in generale e non solo alla programmazione in Java: solo a partire dal secondo capitolo ci concentreremo su questo linguaggio di programmazione.

1.1 Il metodo informatico

L'informatica è un elemento essenziale della società moderna, che, oltre a essere indispensabile strumento della nostra quotidianità, plasma e determina lo sviluppo della società intera. Non esiste campo dell'attività umana in cui le scoperte dell'informatica non abbiano lasciato il segno. L'uso del calcolatore, infatti,

è uscito dai campi tradizionali del calcolo scientifico per entrare in tutte le aree della produzione industriale, dalla medicina all'editoria. Pensiamo, ad esempio, alla 'telematica', nata dall'applicazione dell'informatica alle telecomunicazioni, la quale ha trasformato il nostro modo di comunicare, permettendo lo scambio immediato di documenti complessi, immagini e suoni attraverso una rete di calcolatori. Regna, tuttavia, una certa ambiguità sul concetto diffuso di informatica e, per questo, è importante capire che cosa essa *non* è. L'informatica ha poco a vedere con ciò che oggi giorno è nota come 'alfabetizzazione informatica' (per intendersi, saper usare un calcolatore per scrivere un testo oppure per navigare in Internet): sarebbe come dire che studiare astrofisica consista nell'imparare a usare un telescopio (quest'affermazione è comunemente attribuita a Edsger Dijkstra, informatico ritenuto il fondatore della cosiddetta programmazione strutturata). Ugualmente, l'informatica non consiste semplicemente nello scrivere programmi, anche se è naturale aspettarsi da un informatico la capacità di farlo in modo corretto ed efficace. In base alla definizione fornita dal gruppo di lavoro dell'*Association for Computing Machinery* nel 1989, l'informatica è "*lo studio sistematico dei processi algoritmici che descrivono e trasformano l'informazione: la loro teoria, analisi, progettazione, efficienza, implementazione e applicazione*." Questa definizione utilizza implicitamente il concetto di *algoritmo*, che possiamo informalmente definire come una sequenza precisa di operazioni comprensibili ed eseguibili da uno strumento automatico, ed evidenzia come, nel campo dell'informatica, la soluzione di uno specifico problema consista, anzitutto, nel proporre per il problema stesso un algoritmo risolutivo e, successivamente, nel codificare l'algoritmo proposto in un programma che possa, quindi, essere eseguito da un calcolatore. Sebbene questo libro sia concentrato sulla fase di trasformazione di un algoritmo in un programma, è necessario comunque sapere più precisamente cosa sia un algoritmo: di questo tratteremo nel Paragrafo 1.3, mentre in questo paragrafo faremo riferimento alla definizione informale sopra esposta.

L'informatica è dunque un complesso di conoscenze scientifiche e tecnologiche che permettono di realizzare quello che si potrebbe chiamare il *metodo informatico* o *algoritmico* (si veda la Figura 1.1): così come il metodo scientifico può essere riassunto nel formulare ipotesi che spieghino un fenomeno e nel verificare tali ipotesi mediante l'esecuzione di esperimenti, il metodo informatico consiste nel formulare algoritmi che risolvano un problema, nel trasformare questi algoritmi in sequenze di istruzioni (programmi) per le macchine e nel verificare la correttezza e l'efficacia di tali programmi analizzandoli ed eseguendoli. Nella prima fase del metodo algoritmico, l'interazione con esperti di uno specifico settore applicativo deve portare l'informatico a poter formulare un modello mate-

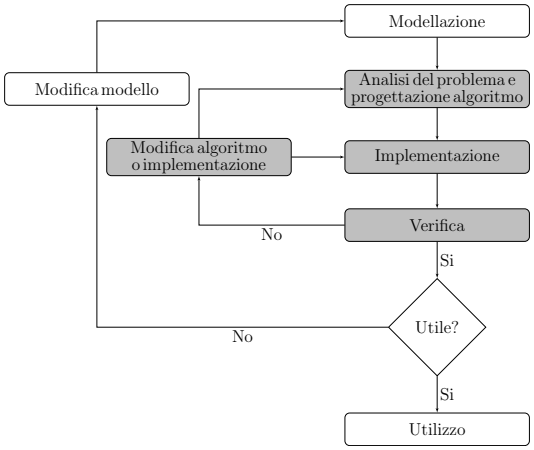

Figura 1.1: Il metodo informatico o algoritmico

matico che rappresenti il più fedelmente possibile il particolare problema oggetto di studio. Questa fase può risultare particolarmente difficile, a causa soprattutto dei diversi linguaggi che, generalmente, vengono usati in diversi contesti applicativi. Una volta sviluppato il modello, il passo successivo consiste nell'analizzarlo e nel progettare per esso un processo algoritmico in grado di risolverlo. Questa è probabilmente la fase più "creativa" dell'intero metodo informatico: non a caso, la progettazione di un algoritmo è spesso descritta come un'arte, per la quale l'informatico ha a disposizione un bagaglio molto vasto di tecniche di progettazione, ma che comunque richiede abilità e fantasia combinate con rigore e impegno. Così come non è facile spiegare come sia possibile dipingere un capolavoro quale *La Venere* del Botticelli, ugualmente non è facile spiegare come sia possibile progettare algoritmi corretti ed efficienti. Una volta progettato, però, l'algoritmo può essere abbastanza facilmente implementato: a questo scopo, è necessario conoscere almeno un linguaggio di programmazione che consenta all'informatico di "descrivere" l'algoritmo stesso al calcolatore, chiedendo a quest'ultimo di eseguirlo. Ovviamente, nessuno è perfetto e tanto meno lo è un informatico: pertanto, sia la fase di progettazione e analisi dell'algoritmo che quella della sua implementazione sono soggette a errori. Per questo motivo, la fase successiva, ovvero quella di verifica, è altrettanto importante. Essa ha lo scopo di individuare i cosiddetti *bug*, ovvero errori concettuali e/o implementativi: nel caso tali errori vengano rilevati sarà necessario modificare la progettazione dell'algoritmo e/o la sua implementazione. Una volta realizzata la corretta implementazione,

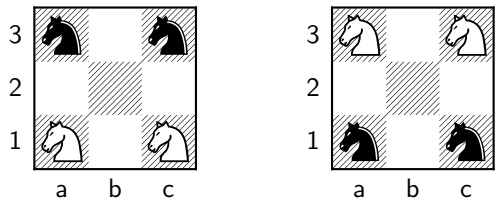

Figura 1.2: Il rompicapo di Guarini: configurazione iniziale e finale

il metodo informatico prosegue verificando che le soluzioni ottenute siano veramente utili per la risoluzione del problema originario: se così non fosse, ciò vorrebbe dire che la modellazione del problema effettuata durante la prima fase nascondeva delle debolezze che la sola verifica sperimentale è stata in grado di evidenziare. Nasce, quindi, la necessità di modificare il modello e ricominciare l'intero processo, facendo uso, ovviamente, di tutte le conoscenze e tecniche acquisite nel corso del processo. Questo ciclo, tuttavia, si dovrebbe ripetere poche volte (sperabilmente mai). Una volta verificata effettivamente l'utilità della soluzione ottenuta, il "capolavoro" prodotto dall'informatico può essere finalmente "esposto" e reso disponibile a chiunque lo desideri.

1.1.1 Trovare il giusto modello

Per chiarire quanto descritto sinora, vediamo un primo esempio di applicazione del metodo algoritmico, in cui la scelta, non immediata, del modello implica, quasi automaticamente, lo sviluppo di un algoritmo estremamente semplice. Il filosofo romagnolo del diciassettesimo secolo Guarino Guarini ideò il seguente rompicapo basato sul gioco degli scacchi.[1] Data una scacchiera di dimensione 3×3 con quattro cavalli posizionati nel modo mostrato alla sinistra della Figura 1.2, qual è la sequenza di mosse più breve che consente ai cavalli di passare da tale configurazione a quella mostrata alla destra della figura, senza mai posizionare due cavalli sulla stessa casella? Ricordiamo che negli scacchi il cavallo muove di due posizioni in orizzontale e una posizione in verticale o viceversa: ad esempio, il cavallo nero in posizione `a3` nella configurazione iniziale può spo-

[1] La presentazione di questo rompicapo e della sua soluzione prende spunto da A. Levitin e M. Levitin, *Algorithmic Puzzles*, Oxford University Press, 2011.