

Corso Python

Antonio Montano

2024-05-24

Table of contents

Preface

This is a Quarto book.

To learn more about Quarto books visit <https://quarto.org/docs/books>.

Part I

Prima parte: I fondamenti

1 I linguaggi di programmazione, i programmi e i programmatori

Partiamo da alcuni concetti basilari a cui collegare quelli che approfondiremo nel corso.

1.1 Cosa sono?

La programmazione è il processo di progettazione e scrittura di istruzioni che un computer può ricevere per eseguire compiti predefiniti. Queste istruzioni sono codificate in un linguaggio di programmazione, che traduce le idee e gli algoritmi del programmatore in un formato che il computer può comprendere ed eseguire.

Cos'è un programma informatico?

Un programma informatico è una sequenza di istruzioni scritte per eseguire una specifica operazione o un insieme di operazioni su un computer. Queste istruzioni sono codificate in un linguaggio che il computer può comprendere e seguire per eseguire attività come calcoli, manipolazione di dati, controllo di dispositivi e interazione con l'utente.

Pensate a un programma come a una ricetta di cucina. La ricetta elenca gli ingredienti necessari (dati) e fornisce istruzioni passo-passo (algoritmo) per preparare un piatto. Allo stesso modo, un programma informatico specifica i dati da usare e le istruzioni da seguire per ottenere un risultato desiderato.

Cos'è un linguaggio di programmazione?

Un linguaggio di programmazione è un linguaggio formale che fornisce un insieme di regole e sintassi per scrivere programmi informatici. Questi linguaggi permettono ai programmatori di comunicare con i computer e di creare software. Alcuni esempi di linguaggi di programmazione includono Python, Java, C++ e JavaScript.

I linguaggi di programmazione differiscono dai linguaggi naturali (come l'italiano o l'inglese) in diversi modi:

1. Precisione e rigidità: I linguaggi di programmazione sono estremamente precisi e rigidi. Ogni istruzione deve essere scritta in un modo specifico affinché il computer possa comprenderla ed eseguirla correttamente. Anche un piccolo errore di sintassi può impedire il funzionamento di un programma.

2. Ambiguità: I linguaggi naturali sono spesso ambigui e aperti a interpretazioni. Le stesse parole possono avere significati diversi a seconda del contesto. I linguaggi di programmazione, invece, sono progettati per essere privi di ambiguità; ogni istruzione ha un significato preciso e univoco.
3. Vocabolario limitato: I linguaggi naturali hanno un vocabolario vastissimo e in continua espansione. I linguaggi di programmazione, al contrario, hanno un vocabolario limitato costituito da parole chiave e comandi definiti dal linguaggio stesso.

Come un programma produce azioni in un calcolatore?

Quando un programma è scritto e salvato, il computer deve eseguirlo per produrre le azioni desiderate. Questo processo avviene in diverse fasi:

1. Compilazione o interpretazione: La maggior parte dei programmi deve essere trasformata da un linguaggio di alto livello (leggibile dall'uomo) a un linguaggio macchina (comprendibile dal computer). Questo avviene attraverso un processo chiamato compilazione (per linguaggi come C++ o Java) o interpretazione (per linguaggi come Python o JavaScript).
2. Esecuzione: Una volta che il programma è stato compilato o interpretato, il computer può eseguire le istruzioni una per una. La CPU (central processing unit) del computer legge le istruzioni e le esegue, manipolando i dati e producendo i risultati desiderati.
3. Interazione con componenti hardware: Durante l'esecuzione, il programma può interagire con vari componenti hardware del computer, come la memoria, i dischi rigidi, la rete, e i dispositivi di input/output (come tastiere e monitor).

1.2 L'Impatto dell'intelligenza artificiale generativa sulla programmazione

Con l'avvento dell'intelligenza artificiale (IA) generativa, la programmazione ha subito una trasformazione significativa. Prima dell'IA generativa, i programmatori dovevano tutti scrivere manualmente ogni riga di codice, seguendo rigorosamente la sintassi e le regole del linguaggio di programmazione scelto. Questo processo richiedeva una conoscenza approfondita degli algoritmi, delle strutture dati e delle migliori pratiche di programmazione.

Inoltre, i programmatori dovevano creare ogni funzione, classe e modulo a mano, assicurandosi che ogni dettaglio fosse corretto, identificavano e correggevano gli errori nel codice con un processo lungo e laborioso, che comportava anche la scrittura di casi di test e l'esecuzione di sessioni di esecuzione di tali casi. Infine, dovevano scrivere documentazione dettagliata per spiegare il funzionamento del codice e facilitare la manutenzione futura.

1.2.1 Attività del programmatore con l'IA Generativa

L'IA generativa ha introdotto nuovi strumenti e metodologie che stanno cambiando il modo in cui i programmatori lavorano:

1. Generazione automatica del codice: Gli strumenti di IA generativa possono creare porzioni di codice basate su descrizioni ad alto livello fornite dai programmatori. Questo permette di velocizzare notevolmente lo sviluppo iniziale e ridurre gli errori di sintassi.
2. Assistenza nel debugging: L'IA può identificare potenziali bug e suggerire correzioni, rendendo il processo di debugging più efficiente e meno dispendioso in termini di tempo.
3. Ottimizzazione automatica: Gli algoritmi di IA possono analizzare il codice e suggerire o applicare automaticamente ottimizzazioni per migliorare le prestazioni.
4. Generazione di casi di test: L'IA può creare casi di test per verificare la correttezza del codice, coprendo una gamma più ampia di scenari di quanto un programmatore potrebbe fare manualmente.
5. Documentazione automatica: L'IA può generare documentazione leggendo e interpretando il codice, riducendo il carico di lavoro manuale e garantendo una documentazione coerente e aggiornata.

1.2.2 L'Importanza di imparare a programmare nell'era dell'IA generativa

Nonostante l'avvento dell'IA generativa, imparare a programmare rimane fondamentale per diverse ragioni. La programmazione non è solo una competenza tecnica, ma anche un modo di pensare e risolvere problemi. Comprendere i fondamenti della programmazione è essenziale per utilizzare efficacemente gli strumenti di IA generativa. Senza una solida base, è difficile sfruttare appieno queste tecnologie. Inoltre, la programmazione insegna a scomporre problemi complessi in parti più gestibili e a trovare soluzioni logiche e sequenziali, una competenza preziosa in molti campi.

Anche con l'IA generativa, esisteranno sempre situazioni in cui sarà necessario personalizzare o ottimizzare il codice per esigenze specifiche. La conoscenza della programmazione permette di fare queste modifiche con sicurezza. Inoltre, quando qualcosa va storto, è indispensabile sapere come leggere e comprendere il codice per identificare e risolvere i problemi. L'IA può assistere, ma la comprensione umana rimane cruciale per interventi mirati.

Imparare a programmare consente di sperimentare nuove idee e prototipare rapidamente soluzioni innovative. La creatività è potenziata dalla capacità di tradurre idee in codice funzionante. Sapere programmare aiuta anche a comprendere i limiti e le potenzialità degli strumenti di IA generativa, permettendo di usarli in modo più strategico ed efficace.

La tecnologia evolve rapidamente, e con una conoscenza della programmazione si è meglio preparati ad adattarsi alle nuove tecnologie e metodologie che emergeranno in futuro. Inoltre, la programmazione è una competenza trasversale applicabile in numerosi settori, dalla biologia

computazionale alla finanza, dall'ingegneria all'arte digitale. Avere questa competenza amplia notevolmente le opportunità di carriera.

Infine, la programmazione è una porta d'accesso a ruoli più avanzati e specializzati nel campo della tecnologia, come l'ingegneria del software, la scienza dei dati e la ricerca sull'IA. Conoscere i principi della programmazione aiuta a comprendere meglio come funzionano gli algoritmi di IA, permettendo di contribuire attivamente allo sviluppo di nuove tecnologie.