

Prefazione

Negli anni successivi alla pubblicazione della prima edizione di *Programmazione in C* sono nati diversi nuovi linguaggi basati sul C (primi tra questi Java e C#) e i linguaggi a esso collegati come C++ e Perl hanno raggiunto una maggiore importanza. Nonostante ciò il C è rimasto popolare come non mai, lavorando nell'ombra e influenzando buona parte del mondo del software. Il C è la *lingua franca* dell'universo dei computer così come lo era nel 1996.

Tuttavia anche il C deve cambiare con i tempi. La necessità di una nuova edizione di *Programmazione in C* è divenuta palese quando è stato pubblicato lo standard C99. La prima edizione, inoltre, con i suoi riferimenti al DOS e ai processori a 16 bit stava diventando obsoleta. La seconda edizione contiene numerosi aggiornamenti ed è stata migliorata in molti altri modi.

Novità nella seconda edizione

Ecco una lista delle nuove caratteristiche e dei miglioramenti presenti nella seconda edizione.

- **Copertura completa di entrambi gli standard C89 e C99.** La principale differenza tra la prima e la seconda edizione consiste nella trattazione dello standard C99. L'obiettivo è quello di illustrare tutte le differenze significative tra il C89 e il C99, incluse tutte le caratteristiche del linguaggio e le funzioni aggiunte da quest'ultimo. Ogni modifica del C99 viene segnalata chiaramente, o con la scritta "C99" nel titolo della sezione oppure, nel caso di discussioni più brevi, con una speciale icona posta nel margine sinistro della pagina. In tal modo intendiamo attirare l'attenzione sulle modifiche e fare in modo che i lettori non interessati al C99 o che non hanno accesso a un compilatore C99 possano individuare facilmente le parti da saltare. Molte delle aggiunte del C99 sono di interesse solo per specialisti, tuttavia alcune delle nuove caratteristiche saranno utili per quasi tutti i programmatori C.
- **Inclusione di una guida veloce a tutte le funzioni di libreria C89 e C99.** L'Appendice D della prima edizione descriveva tutte le funzioni della libreria standard del C89. In questa edizione, l'Appendice E (disponibile online all'indi-

C99

rizzo www.apogeeonline.com/libri/9788850328697/scheda) copre tutte le funzioni di libreria del C89 e di quelle del C99.

- **Maggiore copertura di GCC.** Negli anni successivi alla prima edizione, l'uso di GCC (originariamente chiamato *GNU C Compiler*, mentre il nome attuale è *GNU Compiler Collection*) si è diffuso. GCC presenta vantaggi significativi, che includono: l'alta qualità, la gratuità e la portabilità attraverso una varietà di piattaforme hardware e software. Riconoscendo l'importanza di GCC, nel libro ho inserito più informazioni a riguardo, includendo una trattazione sul suo utilizzo oltre a una illustrazione dei suoi messaggi di errore e dei suoi warning più comuni.
 - **Nuova trattazione dei tipi di dato astratti.** Nella prima edizione una parte significativa del Capitolo 19 era dedicata al C++. Oggi questo materiale sembra meno rilevante, dato che gli studenti potrebbero aver già studiato il C++, Java o il C# prima di leggere questo libro. In questa edizione la trattazione del C++ è stata sostituita da una discussione sulla creazione in C dei tipi di dato astratti.
 - **Estensione della trattazione delle caratteristiche internazionali.** Il Capitolo 25, che è stato dedicato alle caratteristiche internazionali, ora è più lungo e più dettagliato. Le informazioni sul set di caratteri Unicode/UCS e sulle sue codifiche sono il clou delle modifiche apportate al capitolo.
 - **Aggiornamento per rispecchiare le CPU e i sistemi operativi attuali.** Quando ho scritto la prima edizione, le architetture a 16 bit e il sistema operativo DOS erano ancora rilevanti per molti lettori, tuttavia oggi non è più così. La discussione è focalizzata maggiormente sulle architetture a 32 e a 64 bit. L'ascesa di Linux e di altre versioni di UNIX ha richiesto una maggiore attenzione a questa famiglia di sistemi operativi, sebbene vengano menzionati anche gli aspetti relativi ai sistemi operativi Windows e Mac OS.
 - **Più esercizi e progetti di programmazione.** La prima edizione del libro conteneva 311 esercizi. Questa edizione ne ha quasi 500 (498 per essere esatti) divisi in due gruppi: esercizi e progetti di programmazione.
 - **Soluzioni per gli esercizi selezionati e per i progetti di programmazione.** La richiesta più frequente che ho ricevuto dai lettori della prima edizione è stata quella di fornire le soluzioni degli esercizi. In risposta a questa istanza, nel sito knking.com/books/c2 (e www.apogeeonline.com/libri/9788850328697/scheda) vi sono le soluzioni di circa un terzo degli esercizi e dei progetti di programmazione. Questa caratteristica è particolarmente utile per i lettori che non sono inquadrati in un corso universitario e che desiderano controllare il loro lavoro.
- W Gli esercizi e i progetti per i quali vengono fornite le soluzioni sono indicati con l'icona (la W sta per "risposta disponibile sul Web").

Infine, ho colto l'opportunità di migliorare i testi e le spiegazioni presenti nelle varie parti del libro. Le modifiche sono estese e minuziose: ogni frase è stata controllata e, se necessario, riscritta.

Sebbene molto sia stato cambiato in questa edizione, la numerazione originale dei capitoli e delle sezioni è stata mantenuta il più possibile uguale all'originale. Solamente un capitolo (l'ultimo) è interamente nuovo, tuttavia molti presentano delle sezioni

aggiuntive. In alcuni casi le sezioni esistenti sono state rinumerate. Un'appendice (la sintassi C) è stata eliminata, tuttavia ne è stata aggiunta una nuova che opera un confronto tra C99 e C89.

Obiettivi

Gli obiettivi di questa edizione sono i medesimi della prima.

- **Essere chiara, leggibile ed eventualmente gradevole.** Molti libri sul C sono troppo concisi per il lettore medio. Gli altri non sono ben scritti o sono semplicemente tediosi. Ho cercato di dare spiegazioni chiare e precise e di tenere vivo l'interesse del lettore.
- **Essere accessibile.** Ho assunto che il lettore abbia, almeno in parte, un'esperienza di programmazione pregressa, sebbene non sia richiesta la conoscenza di alcun linguaggio in particolare. Per definire i termini ho mantenuto al minimo l'utilizzo di gergo specifico. Ho cercato anche di separare il materiale di livello avanzato dagli argomenti elementari in modo che il principiante non si scoraggi.
- **Essere autorevole senza essere pedante.** Per evitare decisioni arbitrarie su cosa inserire e cosa no, ho incluso nella trattazione tutte le caratteristiche del linguaggio e della libreria. Allo stesso tempo ho cercato di non sovraccaricare il lettore con dettagli non necessari.
- **Essere organizzata per un apprendimento semplice.** La mia esperienza nell'insegnamento del C sottolinea l'importanza di presentare le caratteristiche del C in modo graduale. Ho pertanto utilizzato un approccio a spirale nel quale gli argomenti difficili vengono introdotti brevemente e successivamente rivisitati una o più volte all'interno del libro, con l'aggiunta ogni volta di nuovi dettagli. Si procede per gradi, ogni capitolo è costruito su quanto spiegato in precedenza. Per la maggior parte degli studenti questo probabilmente è l'approccio migliore: evita gli estremi della noia da una parte e del "sovraccarico di informazioni" dall'altra.
- **Motivare le caratteristiche del linguaggio.** Invece di limitarmi a descrivere ogni caratteristica del linguaggio e di fornire pochi esempi del suo utilizzo, ho cercato di fornire motivazioni e di discutere le applicazioni nelle situazioni pratiche.
- **Enfatizzare lo stile.** Per ogni programmatore C è importante sviluppare uno stile coerente. Invece di imporre uno stile, solitamente descrivo qualche possibilità e lascio al lettore la scelta di quella più "attraente". Conoscere le alternative per gli stili è di grande aiuto quando si leggono i programmi scritti da altre persone (azione cui i programmatori dedicano molto tempo).
- **Evitare la dipendenza da una particolare macchina, compilatore o sistema operativo.** Dato che il C è disponibile su una grande varietà di piattaforme, ho cercato di evitare la dipendenza da qualsiasi particolare macchina, compilatore o sistema operativo. Tutti i programmi sono progettati per essere portabili su una grande varietà di piattaforme.
- **Utilizzare illustrazioni per chiarire i concetti chiave.** Ho cercato di inserire molte illustrazioni, dato che penso siano fondamentali per molti aspetti del C. In

particolare, quando possibile, ho cercato di “animare” gli algoritmi mostrando delle istantanee dei dati scattate in differenti momenti dell’elaborazione.

Il metodo

Uno degli obiettivi più importanti è stato quello di seguire un “approccio moderno” al C. Ecco alcune delle strade che ho seguito per cercare di raggiungere questo obiettivo.

- **Mettere il C in prospettiva.** Invece di trattare il C semplicemente come l’unico linguaggio di programmazione che è bene conoscere, l’ho trattato come uno dei molti linguaggi utili. Nel libro illustro il tipo di applicazioni cui è particolarmente adatto e mostro anche come capitalizzare i punti di forza del C minimizzando le sue debolezze.
- **Enfatizzare le versioni standard del C.** Ho prestato un’attenzione minima alle versioni del linguaggio precedenti al C89. Ci sono semplicemente alcuni riferimenti sparsi al K&R C (la versione del linguaggio del 1978 descritta nella prima edizione del volume *The C Programming Language* di Brian Kernighan e Dennis Ritchie). L’Appendice C elenca le maggiori differenze tra il C89 e il K&R C.
- **Sfatare i miti.** I compilatori odierni sono spesso in conflitto con le comuni assunzioni riguardo il C. Non ho esitato a sfatare alcuni di questi miti o a mettere alla prova le credenze che hanno fatto parte a lungo del folclore del C (per esempio la credenza che l’aritmetica dei puntatori sia sempre più veloce dell’indicizzazione dei vettori). Ho riesaminato le vecchie convenzioni del C, mantenendo quelle che sono ancora utili.
- **Enfatizzare l’ingegneria del software.** All’interno del libro tratto il C come uno strumento maturo per l’ingegneria del software, enfatizzando la modalità d’uso per affrontare i problemi che sorgono durante la programmazione in grande. Ho posto un forte accento sulla necessità di rendere i programmi leggibili, manutenibili, affidabili e portabili e ho enfatizzato il concetto di *information hiding*.
- **Postporre le funzionalità a basso livello del C.** Queste funzionalità, sebbene siano utili per il tipo di programmazione di sistema che veniva fatta originariamente in C, non sono più così rilevanti ora che il C viene utilizzato per una grande varietà di applicazioni. Invece di introdurle nei primi capitoli, così come fanno molti libri, le rimando al capitolo 20.
- **De-enfatizzare “l’ottimizzazione manuale”.** Molti libri insegnano al lettore a scrivere dei “trucchetti” di codice in modo da ottenere piccoli vantaggi nell’efficienza del programma. Con l’odierna abbondanza di compilatori C che ottimizzano il codice, spesso queste tecniche non sono necessarie, anzi possono rendere i programmi meno efficienti.

Domande & Risposte

Ogni capitolo termina con una sezione “Domande & Risposte”, ovvero una serie di domande e risposte relative al materiale trattato nel capitolo stesso. Gli argomenti affrontati in queste sezioni includono:

- **Frequently asked questions.** Ho cercato di rispondere alle domande che si presentano frequentemente nei miei corsi, in altri libri e nei newsgroup relativi al C.
- **Discussioni aggiuntive e chiarimenti sugli argomenti complessi.** Sebbene i lettori con una certa esperienza in diversi linguaggi possano essere soddisfatti da una breve spiegazione e da un paio di esempi, quelli con meno esperienza hanno bisogno di qualcosa in più.
- **Argomenti marginali che non appartengono al flusso principale.** Alcune domande sollevano delle questioni tecniche che non saranno d’interesse per tutti i lettori.
- **Materiale troppo avanzato o troppo “esoterico” per essere di interesse per il lettore medio.** Domande di questa natura vengono segnalate con un asterisco (*). I lettori curiosi con una buona esperienza in programmazione potrebbero desiderare di addentrarsi immediatamente in queste domande, altri dovrebbero saltarle nella prima lettura del testo. Queste domande si riferiscono spesso agli argomenti trattati in capitoli successivi.
- **Differenze comuni tra i compilatori C.** All’interno del libro discuto alcune caratteristiche utilizzate di frequente (ma non standard) che vengono fornite da alcuni particolari compilatori.

D&R

Alcune domande delle sezioni D&R sono associate direttamente a punti specifici all’interno del capitolo, segnalati da una speciale icona che indica al lettore la disponibilità di informazioni aggiuntive.

Altre caratteristiche

In aggiunta alle sezioni D&R, ho incluso un certo numero di funzionalità, molte delle quali vengono segnalate con delle icone semplici e facilmente distinguibili (alcune sono indicate sul margine sinistro della pagina).



- **Avvertimenti:** sollecitano il lettore a fare attenzione agli errori comuni. Il C è famoso per le sue trappole, elencarle tutte è un compito praticamente impossibile. Ho scelto gli errori più comuni e/o più importanti.
- **Riferimenti incrociati [riferimenti incrociati > Prefazione]:** forniscono delle indicazioni ipertestuali per individuare le informazioni. Sebbene molti siano dei puntatori ad argomenti trattati successivamente nel libro, alcuni rimandano ad argomenti illustrati precedentemente, che il lettore può voler rivedere.

PORTABILITÀ

- **Consigli per la portabilità:** sono suggerimenti per la scrittura di programmi che siano indipendenti da una particolare macchina, compilatore o sistema operativo.

- **Riquadri:** trattano argomenti che non fanno strettamente parte del C, ma che ogni programmatore esperto di C dovrebbe conoscere (per un esempio si veda il riquadro “Codice Sorgente” a pagina seguente).
- **Appendici:** forniscono utili informazioni di riferimento.

Programmi

Scegliere programmi esemplificativi non è un compito facile. Se i programmi sono troppo brevi e artificiali, i lettori potrebbero non percepire come le caratteristiche utilizzate vengono impiegate nel mondo reale. D’altro canto, se un programma è *troppo* realistico, il suo punto chiave può essere perso facilmente all’interno di una foresta di dettagli. Ho scelto una via di mezzo, utilizzando piccoli e semplici esempi per rendere chiari i concetti quando vengono introdotti per la prima volta, costruendo poi gradatamente programmi completi. Non ho incluso programmi particolarmente lunghi, secondo la mia esperienza gli insegnanti non hanno tempo di trattarli e gli studenti non hanno la pazienza di leggerli. Non trascuro tuttavia le complicazioni che nascono nella creazione di programmi di grandi dimensioni: il Capitolo 15 (“Scrivere programmi di grandi dimensioni”) e il Capitolo 19 (“Progettazione di un programma”) trattano questi argomenti in dettaglio.

Ho resistito alla tentazione di riscrivere i programmi e di avvantaggiarmi delle funzionalità del C99, dato che non tutti i lettori potrebbero voler utilizzare il C99 o avere accesso a un compilatore conforme a questo standard. Tuttavia ho utilizzato l’header `<stdbool.h>` del C99 perché definisce le macro chiamate `bool`, `true` e `false`. Se il vostro compilatore non supporta questo header dovrete fornire la vostra definizione per questi nomi.

I programmi di questa edizione hanno subito una modifica minore. La funzione `main` ora segue la forma `int main(void){...}` nella maggior parte dei casi. Questa modifica rispecchia la pratica raccomandata ed è compatibile con il C99, il quale richiede un esplicito tipo restituito per ogni funzione.

Codice sorgente

I codici sorgente di tutti i programmi sono disponibili presso il sito www.apogeeonline.com/libri/9788850328697/scheda (e www.knking.com/books/c2, all’interno di questo sito si possono trovare anche aggiornamenti, correzioni e notizie sul libro a cura dell’Autore).

Pubblico

Questo libro è stato pensato per un corso di C a livello universitario. Una precedente esperienza di programmazione con un linguaggio ad alto livello o con l’assembler può essere utile, ma non è necessaria per il lettore di livello intermedio.

Dato che il libro non necessita di riferimenti esterni ed è utilizzabile sia come guida di riferimento sia per imparare, si rivela un compagno eccellente per i corsi di strutture dati, progettazione dei compilatori, sistemi operativi, computer grafica, sistemi embedded e tutti gli altri corsi che utilizzano il C per lo sviluppo dei progetti.

Grazie alle sezioni D&R e all'enfasi posta sui problemi pratici, il libro è interessante anche per i lettori che sono iscritti a corsi di formazione o che stanno studiando il C da autodidatti.

Organizzazione

Il libro è suddiviso in quattro parti.

- **Caratteristiche base del C.** I Capitoli 1-10 trattano un numero sufficiente di caratteristiche del C da permettere al lettore di scrivere programmi costituiti da singoli file utilizzando vettori e funzioni.
- **Caratteristiche avanzate del C.** I Capitoli 11-20 sono basati sul materiale affrontato nei capitoli precedenti. Gli argomenti diventano più difficili e viene fornita una trattazione approfondita di puntatori, stringhe, preprocessore, strutture, unioni, enumerazioni e caratteristiche a basso livello del C. Due capitoli (il 15 e il 19) offrono una guida alla progettazione dei programmi.
- **La libreria standard del C.** I Capitoli 21-27 si focalizzano sulla libreria del C, ovvero una grande collezione di funzioni che viene fornita da tutti i compilatori. Molto probabilmente questi capitoli verranno utilizzati principalmente come guida di riferimento, sebbene alcune loro parti siano adatte alle lezioni.
- **Riferimento.** L'Appendice A fornisce una lista completa degli operatori C. L'Appendice B descrive le principali differenze tra il C99 e il C89, mentre l'Appendice C tratta le differenze tra il C89 e il K&R C. L'Appendice D elenca il set di caratteri ASCII. L'Appendice E (disponibile online all'indirizzo www.apogeeonline.com/libri/9788850328697/scheda) descrive in dettaglio tutte le funzioni presenti nelle librerie standard del C89 e del C99. Una bibliografia commentata indica al lettore ulteriori fonti di informazione.

Un corso completo sul C deve trattare in sequenza tutti i capitoli dall'1 al 20, aggiungendo quando necessario degli argomenti presi dai capitoli che vanno dal 21 al 27 (il Capitolo 22, che include una trattazione dell'input/output sui file è il capitolo più importante di questo gruppo). Un corso più breve, senza perdere di continuità, può omettere questi argomenti: Sezione 8.3 ("Vettori di lunghezza variabile"), Sezione 9.6 ("Ricorsione"), Sezione 12.4 ("Puntatori e vettori multidimensionali"), Sezione 12.5 ("Puntatori e vettori a lunghezza variabile"), Sezione 14.5 ("Direttive varie"), Sezione 17.7 ("Puntatori a funzione"), Sezione 17.8 ("Puntatori restricted"), Sezione 17.9 ("Membri vettore flessibili"), Sezione 18.6 ("Funzioni inline"), Capitolo 19 ("Progettazione di un programma"), Sezione 20.2 ("Campi di bit nelle strutture"), Sezione 20.3 ("Altre tecniche a basso livello").

Esercizi e progetti di programmazione

Ovviamente, per un libro di testo è essenziale possedere una buona varietà di esercizi. Questo volume contiene sia esercizi (problemi che non richiedono la scrittura di un programma completo) sia progetti di programmazione (problemi che richiedono la scrittura o la modifica di un intero programma).

Alcuni esercizi richiedono risposte non ovvie (alcuni le definirebbero “domande difficili”). Dato che i programmi C contengono spesso numerosi esempi di questo tipo di codice, penso sia necessario fornire un po’ di pratica a riguardo. Tuttavia, sono stato corretto segnalando questi esercizi con un asterisco (*): quando affrontate un esercizio di questo tipo fate molta attenzione e ragionate approfonditamente, oppure è meglio se lo evitate del tutto.

Errori, dimenticanze

Ho fatto un grande sforzo per assicurare l’accuratezza di questo testo. Inevitabilmente, però, ogni libro di queste dimensioni contiene qualche errore. Se ne individuate vi prego di contattarmi presso cbook@knking.com. Inoltre apprezzo qualsiasi opinione sulle caratteristiche del libro che avete trovato più interessanti, su quelle delle quali avreste fatto a meno e su quelle che avreste voluto fossero state aggiunte.

Ringraziamenti

Per prima cosa vorrei ringraziare i miei editor alla Norton: Fred McFarland e Aaron Javiskas. Fred è stato coinvolto nella seconda edizione dal principio, mentre Aaron è intervenuto con rapida efficienza per portarla a compimento. Vorrei ringraziare anche il caporedattore Kim Yi, la redattrice Mary Kelly, il responsabile di produzione Roy Tedoff e l’assistente editoriale Carly Fraser.

Sono profondamente in debito con i seguenti colleghi, che hanno rivisto alcuni o tutti i manoscritti della seconda edizione:

Markus Bussmann, dell’Università di Toronto
Jim Clarke, dell’Università di Toronto
Karen Reid, dell’Università di Toronto
Peter Seebach, moderatore di *comp.lang.c.moderated*

Jim e Peter meritano uno speciale riconoscimento per le loro revisioni dettagliate, che mi hanno evitato un buon numero di errori imbarazzanti. I revisori della prima edizione erano, in ordine alfabetico: Susan Anderson-Freed, Manuel E. Bermudez, Lisa J. Brown, Steven C. Cater, Patrick Harrison, Brian Harvey, Henry H. Leitner, Darrel Long, Arthur B. Maccabe, Carolyn Rosner e Patrick Terry.

Ho ricevuto molti commenti utili dai lettori della prima edizione: voglio ringraziare tutti quelli che mi hanno scritto. Anche gli studenti e i colleghi della Georgia State University hanno fornito un prezioso feedback. Ed Bullwinkel e sua moglie Nancy sono stati così gentili da leggere la maggior parte del manoscritto. Sono particolarmente grato al mio capo dipartimento, Yi Pan, che ha supportato il progetto.

Mia moglie, Susan Cole, è stato un pilastro di forza come sempre. Anche i nostri gatti, Dennis, Pounce e Tex hanno contribuito al completamento del libro: le loro occasionali lotte feline mi hanno aiutato a rimanere sveglio quando lavoravo fino a notte fonda.

Infine vorrei ringraziare Alan J. Perlis, che non è più tra noi. Ho avuto il privilegio di studiare brevemente sotto la sua guida a Yale nella metà degli anni Settanta.