Il linguaggio C

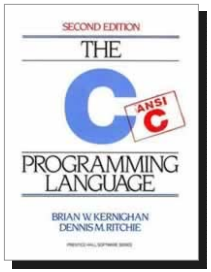
Ver. 2.4

Storia e versioni

- Sviluppato da Dennis Ritchie ai Bell Labs nel 1972 per realizzare il sistema operativo UNIX
- **K&R C**: 1978 (prima versione, "K&R" dal nome degli autori del libro che lo ha divulgato: Kernighan e Ritchie)
- **ANSI C**: 1989 (alias: Standard C, C89)
- **ISO C**: 1990 (quasi identico al C89, alias: C90)
- **C99**: 1999 (Nuovo standard ISO)

Standard ANSI 1989

- In questo corso ci si attiene strettamente alla versione ANSI 1989 (ancora il più utilizzato)
- Queste slide coprono la maggior parte del linguaggio, salvo alcuni dettagli, per i quali si rimanda al testo di riferimento:
  "Il linguaggio C", B. Kernighan, D. Ritchie, 2ª ed., 2004, Pearson/Prentice-Hall
  (2ª ed. originale: 1988)
  "Testo sacro" per generazioni di programmatori, la versione italiana è aggiornata secondo l'errata corrige dagli autori per aderire allo standard ANSI 89



Bibliografia e approfondimenti

- "Il linguaggio C", B. Kernighan, D. Ritchie, 2ª ed., 2004, Pearson/Prentice-Hall
- "ANSI C - A Lexical Guide", Mark Williams Company, 1988, Prentice Hall
- "C-FAQ-list", S. Summit, 2004, URL: http://www.eskimo.com/~scs/C-faq/top.html
- "C - Corso completo di programmazione", H. Deitel, P. Deitel, 3ª ed., Apogeo, 2007
- "Programmare in C", K. N. King, Apogeo, 2009
- Newsgroup: comp.lang.c

Caratteristiche del linguaggio

- Un compilatore C è disponibile su tutti sistemi
- Codice molto efficiente (veloce)
- Ha caratteristiche di alto livello: adatto per programmi anche complessi
- Ha caratteristiche di basso livello (accesso ai bit): permette di sfruttare le peculiarità proprie di una macchina o architettura (efficienza)
- Tantissime librerie per aggiungere funzionalità
- Tra i linguaggi più diffusi, il più usato per sviluppare software di sistema
- Interfaccia utente testuale (+libreria grafiche)
- Non è a oggetti e ha gestione manuale della memoria dinamica (nessun *garbage collector*)

Fasi di compilazione

1. Il **preprocessore**
   elabora le direttive `#include`, `#define`, ...
   modificando il sorgente
2. Il **compilatore**
   traduce il codice C in linguaggio macchina:
   1. con ottimizzazione (della velocità di esecuzione o della dimensione dell'eseguibile)
   2. senza ottimizzazione (per il debug)
3. Il **linker**
   assembla in un unico file eseguibile:
   - i file oggetto prodotti da diversi file sorgente (anche compilati da sorgenti scritti in linguaggi diversi)
   - le librerie (I/O, matematiche, network, ecc.)

Errori e warning

- Il compilatore verifica la correttezza del codice C e produce due tipi di errori:
  - **Error**:
    errori sintattici, impediscono la generazione del codice eseguibile
  - **Warning**:
    errori non sintattici che non impediscono la generazione del codice eseguibile; i warning segnalano un possibile (e altamente probabile) problema che il compilatore risolve in base a regole generiche (ma attenzione: la soluzione generica potrebbe non essere quella corretta)
- Un codice pulito non deve produrre né errori né warning

La sintassi

- I caratteri maiuscoli sono considerati diversi dai corrispondenti minuscoli (il linguaggio C è "*case sensitive* ")
- Le istruzioni sono una sequenza di caratteri terminate dal carattere ';'
- Quando l'istruzione è il solo carattere ';' essa è detta ***istruzione nulla*** e non produce alcuna azione (esempi più avanti nel corso)
- I commenti sono annotazioni sul codice fatte dal programmatore, iniziano con la coppia di caratteri `/*` e terminano con la coppia `*/`, vengono ignorati dal compilatore che li considera come un unico carattere spazio

La sintassi

- Le istruzioni possono continuare su più righe
- Si può andare a capo in ogni punto dove si può mettere uno spazio, esclusi quelli all'interno delle *stringhe* (sequenze di caratteri delimitate da doppie virgolette, es. "ciao ciao")
- Una sequenza (anche mista) di uno o più:
  - spazi
  - caratteri di tabulazione (Tab)
  - ritorni a capo
  - commenti

  è considerata dal compilatore equivalente ad un unico spazio (tranne che all'interno delle stringhe)

La sintassi

- Un ***blocco*** di codice è un insieme di istruzioni racchiuso tra parentesi graffe e costituito, nell'ordine, da due parti:
  - una sezione *opzionale* con la definizione di tutte le variabili ad uso esclusivo di quel blocco
  - una sezione con le istruzioni eseguibili
- Le parentesi graffe sono opzionali e normalmente omesse se il blocco di codice è costituito da una sola istruzione (salvo il blocco che racchiude il corpo di una funzione, in particolare il `main`)

La sintassi

- Le istruzioni di un blocco (non le eventuali parentesi graffe) si scrivono tutte ***indentate*** di un numero fisso di spazi (es. 3)

```
{
   a = 12;
   b = 23;
   c = a + b;
}
```

- L'indentazione viene ignorata dal compilatore ma aiuta il programmatore a comprendere meglio il flusso del programma per cui va fatta *mentre si programma* e non dopo