03-10-2003
Il linguaggio C

Linguaggi di Programmazione I

Ferdinando Cicalese

Il linguaggio C

- Linguaggio *general-purpose*
- Creato nel 1972 nei Bell Labs da Dennis Ritchie su PDP11
- Usato per il Sistema Operativo Unix da Ken Thompson
- Negli anni '80 nasce ANSI C (ANSI=American National Standards Institute)
- ANSI C viene approvato dall'ISO (International Standardization Organization) nel 1990
- Alla base di linguaggi più moderni (C++)

Caratteristiche del C

- Il C è "piccolo"
  - Ha poche parole chiave (riservate) (meno del Pascal)
  - Può essere appreso facilmente
- E' il linguaggio di sviluppo standard per personal computer
  - gran parte di MS-DOS è scritta in C
- è utilizzato nello sviluppo di molti pacchetti applicativi: programmi per la gestione di basi di dati, librerie grafiche, ecc.

Caratteristiche del C

- Il C è portabile
  - Il codice per una macchina può essere facilmente trasferito su un'altra
  - Libreria standard di funzioni il cui comportamento è lo stesso su ogni macchina
- Il C è potente
  - operatori potenti che permettono di accedere alla macchina a livello dei bit
- Il C è efficiente
  - operatori efficienti
  - aritmetica degli indirizzi

Caratteristiche del C

- Il C è modulare
  – permette facilmente la realizzazione di software di grandi dimensioni
- Il C è alla base del C++…. ed il C++ è alla base di Java
- Il C non è un linguaggio perfetto
  – assenza di controllo automatico su dimensione degli array
  – sintassi complicata
  – alcuni operatori sono molto simili

Il programma "Hello world"

commento

inclusione di file di intestazione

funzione main

```
/* File: hello.c
* Questo programma
* stampa il messaggio
* "Hello world"
*/

#include <stdio.h>

main ()
{
printf ("Hello World!\n");
}
```


Il programma "Hello world"

I commenti

```
/* File: hello.c
* Questo programma
* stampa il messaggio
* "Hello world"
*/

#include <stdio.h>

main ()
{
printf ("Hello World!\n");
}
```

- Come e dove compaiono:
  – racchiusi tra /* e */
  – su più linee
  – in qualunque punto del programma
- A che servono:
  – leggibilità
  – riutilizzo software
  – mantenimento del software

Il programma "Hello world"

Inclusione di librerie

```
/* File: hello.c
* Questo programma
* stampa il messaggio
* "Hello world"
*/

#include <stdio.h>

main ()
{
printf ("Hello World!\n");
}
```

- Cosa sono:
  – collezioni di strumenti (funzioni, etc.) scritti da altri

  Esempio: *stdio* (standard I/O)
  – linkate dal linker al file oggetto
- Per usare una libreria
  – includere l' *header file*
  – specifica cosa offre la libreria (prototipi)
  – estensione .h
  – nome tra < e >

  Esempio: <stdio.h>

Il programma "Hello world"

Funzione main

```
/* File: hello.c
* Questo programma
* stampa il messaggio
* "Hello world"
*/

#include <stdio.h>

main ()
{
printf ("Hello World!\n");
}
```

- Funzione:
  - sequenza di passi di programma con un nome
- Una funzione ha:
  - un nome
  - parametri input/output
  - un corpo (racchiuso tra graffe)
- La funzione *main*
  - deve esistere in qualsiasi programma C completo
  - viene eseguita per prima

Il programma "Hello world"

Funzioni

```
/* File: hello.c
* Questo programma
* stampa il messaggio
* "Hello world"
*/

#include <stdio.h>

main ()
{
printf ("Hello World!\n");
}
```

- Parametri passati tra parentesi

Esempio:

printf ("Hello World!\n")

il parametro è la stringa (costante di tipo stringa)

"Hello World!\n"

i due caratteri \n rappresentano un unico carattere che fa andare il cursore all'inizio della linea succesiva

Il programma "Hello world"

Istruzioni

```
/* File: hello.c
* Questo programma
* stampa il messaggio
* "Hello world"
*/

#include <stdio.h>

main ()
{
printf ("Hello World!\n");
}
```

- Sono terminate da ;

Esempio:

printf ("Hello World!\n");

rappresenta una chiamata alla funzione *printf*

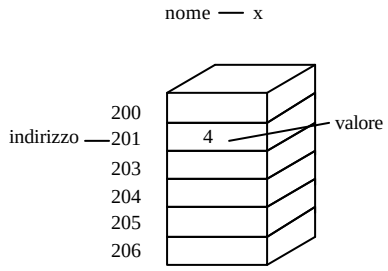
la funzione *printf* in questo caso stampa *Hello World*

e poi sposta il cursore all'inizio della linea successiva

Le variabili

- sono un'importante *facility* dei linguaggi ad alto livello
- sono "contenitori" di dati (il cui valore può non essere noto quando si scrive il programma)
- una variabile è caratterizzata da:
  - nome (identificativo)
  - valore
  - indirizzo
- si accede ad una variabile specificandone il nome

Le variabili



Esempio dell'uso delle variabili

```
/* File: somma.c
* Somma due interi
*/
#include <stdio.h>
main ()
{
int x, y, sum;
x = 4;
y =7;
sum = x + y;
printf ("Somma=%d", sum);
}
```

x, y e sum sono variabili

Esempio dell'uso delle variabili

```
/* File: somma.c
* Somma due interi
*/
#include <stdio.h>
main ()
{
int x, y, sum;
x = 4;
y =7;
sum = x + y;
printf ("Somma=%d", sum);
}
```

Dichiarazione delle variabili x, y, e sum

Assegnamento a x e y dei loro valori iniziali (inizializzazione)

Assegnamento a sum del valore risultante dalla somma di x e y

Le variabili

- Le variabili devono essere *dichiarate* prima di essere usate
  - nome di ciascuna variabile
  - tipo di ciascuna variabile
- La dichiarazione di una variabile istruisce il compilatore:
  - su quanta memoria deve essere *allocata* alla variabile
  - sul tipo di operazioni consentite per la variabile

Esempio: int x, y,sum; è una dichiarazione delle tre variabili di tipo *int* x, y e z

Le variabili

- In un programma una variabile
  - può essere letta (per prelevarne il contento)

    Esempio:

    printf(“Somma=%d”,sum); legge il contenuto di sum
  - può essere scritta (per assegnarle un valore o modificarne il valore)

    Esempio: x = 4; attribuisce ad x il valore 4
  - se ne può prelevare l’indirizzo tramite l’operatore &

    Esempio: &x è l’indirizzo di x

Esempio dell’uso delle variabili

```
/* File: somma.c
* Somma due interi
*/
#include <stdio.h>
main ()
{
int x, y, sum;
x = 4;
y =7;
sum = x + y;
printf (“Somma=%d”, sum);
}
```

Dichiarazione :

Nomi: x, y, e sum

Tipo: int

Scrittura:

Assegnamento a x e y dei valori 4 e 7

Lettura:

I valori di x e y sono letti e usati nella somma

Scrittura:

Assegnamento a sum del valore risultante dalla somma di x e y

Un altro esempio dell’uso delle variabili

```
/* File: maratona.c
* Converti in chilometri la lunghezza della maratona
*/
#include <stdio.h>
main()
{
int miglia, yard;
float km;
miglia = 26;
yard = 385;
km = 1.609*(miglia + yard/1760.0);
printf (“\nUna maratona è lunga %f chilometri. \n”, km);
}
```


Tipi di dato e conversione di tipo

- int miglia, yard; indica che miglia e yard assumono valori interi
- float km; indica che km assume valori reali (6 cifre significative)
- La costante 1760.0 è di tipo *double* (16 cifre significative)
- yard/1760.0 è una operazione tra un *int* e un *double*
  - il risultato è un *double* OK!
- yard/1760 sarebbe stata un’ operazione tra interi
  - il risultato è un intero (nel nostro programma, sarebbe stato 0) NO!
- km = 1.609*(miglia + yard/1760.0);
  - il risultato di 1.609*(miglia + yard/1760.0) è un *double*
  - l’assegnamento di un valore di tipo *double* a km converte il valore in float