Il controllo del programma in C

DD Cap.3 pp.91-130

Un esempio per iniziare…

```
/* calcolo di 8!*/
 #include <stdio.h>
 #define C 8
 int main()
 {
   int i=1; int fatt=1;
   while (i<=C)
   {
     fatt*=i;
     i++;
   }
 printf("8!=%d", fatt);
 return 0;
 }
```

inizializzazione

calcolo

conclusione

prima di proseguire, parliamo ancora della printf()...

La maggior parte degli algoritmi ha:
1. una parte di inizializzazione
2. una parte di calcolo
3. una parte conclusiva

Altri cenni su printf()

printf("8!=%d", fatt);

Questa printf() ha due argomenti, una stringa tra apici ed una lista di variabili.

- Nella stringa, %d indica che i dati da stampare sono interi decimali;
- fatt è la variabile il cui contenuto deve essere stampato esattamente dove si trova il %d nella stringa;

Alcuni caratteri di conversione

%d intero decimale	%c carattere
%f reale (notaz. decimale)	%s stringa
%e reale (notaz. esponenz.)	%l davanti a d o l

Esercizi (printf)

- Scrivere un programma che prenda in input due interi che rappresentano le lunghezze di cateti di un triangolo rettangolo e restituisca in output la lunghezza dell'ipotenusa
- Scrivere un programma che stampi il proprio nome e cognome: a. su una riga; b. su due righe; c. in un riquadro.
- Scrivere un programma che stampi una tabella delle potenze graficamente ordinata:

Intero	Quadrato	Cubo	Quarta	Quinta
1	1	1	1	1
2	4	8	16	32
…	…	...		

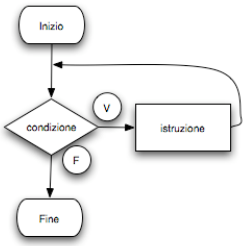
- Scrivere un programma che stampi la seguente figura, dove le dimensioni siano date in input:

```
*****
 *****
*****
 *****
*****
```

il ciclo while

- Il ciclo while:
  - viene testata la condizione per la ripetizione del ciclo; se essa è verificata il corpo del ciclo viene eseguito, altrimenti il controllo va all'istruzione successiva al ciclo while.
- Formato:

```
while(condizione)
  {
    istruzione;
  }
```



Osserviamo…

… il ciclo while dell'esempio:

```
while (i<=C)
    {
      fatt*=i;
      i++;
    }
```

Alcuni programmatori amano compattare il codice:

```
while (++i<=C)
fatt*=i;
```

(ma questa volta i parte da 0)

Altri costrutti per dire la stessa cosa:

```
for (i=1; <=C; i++)
fatt*=i;
```

```
do
{
  fatt*=i;
  i++;
}while (i<=C);
```

il ciclo for (1) p.94

nome della var. di controllo: i

parola chiave for

niente ; qui, altrimenti il ciclo non esegue istruzioni

```
for (i=1; i<=C; i++)
```

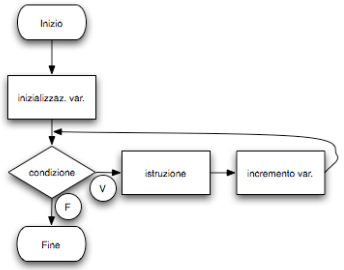
incremento della var. di controllo

valore iniziale della var. di controllo

condizione di conti-nuazione ciclo

Formato generale del ciclo for:

```
for (inizializzazione; test della con-
  dizione per la continuazione del
  ciclo; incremento)
    istruzione;
```



il ciclo for (2)

- Espressioni aritmetiche:
  - L'inizializzazione, la condizione di continuazione del ciclo e l'incremento possono contenere espressioni aritmetiche.

    Esempio: `x=2; y=10;`

    ```
    for ( j = x; j <= 4 * x * y; j += y / x )
    ```

    è equivalente a:

    ```
    for ( j = 2; j <= 80; j += 5 )
    ```
- Note sul ciclo for:
  - L'"incremento" può essere negativo (decremento)
  - Se la cond. di continuazione del ciclo è inizialmente falsa il corpo del ciclo for non viene eseguito, e il controllo procede con l'istruzione seguente al ciclo for
  - La variabile di controllo è spesso usata all'interno del corpo, ma non è necessario
  - I cicli for e while sono assolutamente equivalenti

Esempio

Problema: eseguire la moltiplicazione 8x6 simulandola con l'addizione (8+8+…+8).

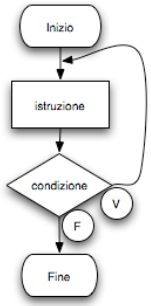
somma=0
considera un contatore i
i=1
finchè (i non è > di 6)
  somma=somma+8
  i=i+1
stampa somma

```
/* calcolo di 8x6*/
#include <stdio.h>
#define C1 6
#define C2 8

int main()
{
  int somma=0; int i;
  for(i=1; i<=C1; i++)
    somma+=C2;
  printf("6x8=%d", somma);
  return 0;
}
```

il ciclo do…while p.108

- Il ciclo do...while:
  - è simile al ciclo while
  - la condizione per la ripetizione del ciclo viene testata dopo che il corpo del ciclo è stato eseguito una prima volta, quindi tale corpo viene eseguito sempre almeno una volta.
- Formato:

```
do
  {
    istruzione;
  } while(condizione);
```



Esempio

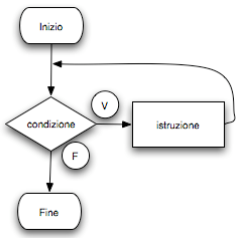
Problema: eseguire la moltiplicazione 8x6 simulandola con l'addizione (8+8+…+8).

somma=0
considera un contatore i
i=1
finchè (i non è > di 6)
  somma=somma+8
  i=i+1
stampa somma

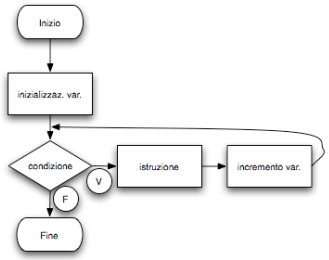
```
/* calcolo di 8x6*/
#include <stdio.h>
#define C1 6
#define C2 8

int main()
{
  int somma=0; int i=1;
  do
  {
    somma+=C2;
    i++;
  }
  while (i<=C1);
  printf("6x8=%d", somma);
  return 0;
}
```

I tre cicli

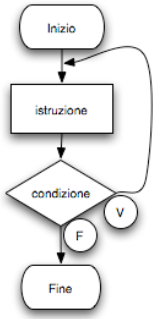
ciclo while



ciclo for



ciclo do…while



Lettura dell'input (1)

- Fin ora abbiamo sempre usato numeri fissi (costanti).
- Che succede se vogliamo inserire un valore da tastiera?
- istruzione scanf()
  - come printf() ha due argomenti, una stringa tra apici ed una lista di variabili
  - nella stringa compaiono i caratteri di controllo che specificano il tipo di input
  - nella lista compaiono le variabili in cui memorizziamo i valori
  - ogni nome di variabile è preceduto da & (operatore di indirizzo) che è, al momento, difficile da capire. Per ora ricordiamo di metterlo sempre prima delle variabili da stampare.

Lettura dell'input (2)

Problema: Sommare due interi dati in input

leggi il primo intero, a
leggi il secondo intero, b
esegui a+b
stampa la somma

```
/* somma di 2 interi*/
#include <stdio.h>
int main()
{
   int a; int b; int s;
   scanf("%d",&a);
   scanf("%d",&b);
   s=a+b;
   printf("%d+%d=%d", a, b, s);
}
```

Meglio:
printf("%d+%d=%d", a, b, a+b);
e si evita la variabile s

Esercizi (cicli)

Diagramma a blocchi, pseudocodice e traduzione in 3 programmi C, ciascuno con un ciclo diverso:

- Prendere in input un intero *n* e sommare tutti i numeri da 1 a *n*
- Prendere in input 2 interi *a* e *b* e calcolare axb simulando con la somma, *a+a+…+a* (*b* volte)
- Prendere in input un intero *n* e stampare un quadrato di "A" di dimensione *nxn*
- Prendere in input 20 interi e calcolarne la media (senza usare 20 variabili!). E se invece di 20 ci fosse una variabile?
- Prendere in input 20 interi compresi tra 1 e 3 e dire quanti 1, quanti 2 e quanti 3 ci sono nella sequenza (come sopra).

Valore sentinella (1)

Problema: far inserire ad uno studente i propri voti e stamparne la media

- Dobbiamo eseguire un ciclo, ma l'utente non deve preventivamente inserire il numero di esami. Come facciamo a sapere quando ha finito?
- In un ciclo, se ci sono dei valori che la variabile contatore non assume mai, uno di tali valori può essere usato per segnalare che si è arrivati alla fine della serie di valori da inserire.
- Qui, i voti vanno da 18 a 30, quindi 0 può essere usato come valore sentinella

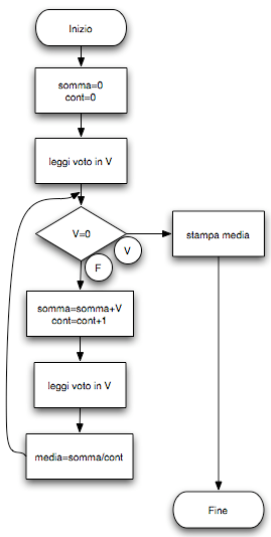
Input: 28 27 30 22 0

Attenzione: non si può sempre dire!

Valore sentinella (2)

Problema: far inserire ad uno studente i propri voti e stamparne la media

```
somma=0; cont=0;
leggi il primo voto in V
finché (V è diverso da 0)
  somma=somma+V
  cont=cont+1
  leggi il prossimo voto in V
media=somma/cont
stampa media
```



l'istruzione break p.110

- L'istruzione break causa l'uscita immediata da un ciclo (oppure da uno switch, che vedremo tra poco)
- L'esecuzione del programma continua con la prima istruzione dopo la struttura interrotta
- L'istruzione break nei cicli si può sempre evitare scrivendo opportunamente la condizione di uscita dal ciclo, e può confondere la comprensione, quindi…
  E' MEGLIO EVITARLA!

Esempio:
```
for (i=1; i<=10; i++)
{
  if (i==5) break;
  printf("%d ", i);
}
```

Output: 1 2 3 4

l'istruzione continue p.110

- L'istruzione continue fa in modo che vengano ignorate le istruzioni ad essa successive nel corpo di un ciclo, SOLO PER L'ITERAZIONE CORRENTE.
- L'esecuzione del programma continua con:
  - while e do…while
    viene immediatamente eseguito il test di entrata nel ciclo e, se verificato, si rientra nel ciclo
  - for
    viene eseguita l'espressione di incremento e poi eseguito il test di entrata; se verificato si rientra nel ciclo
- L'istruzione continue si può sempre evitare scrivendo opportunamente la condizione di uscita dal ciclo, e può confondere la comprensione, quindi…
  E' MEGLIO EVITARLA!

Esempio:
```
for (i=1; i<=10; i++)
{
  if (i==5) continue;
  printf("%d ", i);
}
```

Output: 1 2 3 4 6 7 8 9 10

L'istruzione di selezione multipla switch (1) p.102

L'istruzione switch è utile quando una variabile o un'espressione viene testata per tutti i valori che essa può assumere e, a seconda del valore, sono intraprese azioni diverse.
Il formato prevede una serie di case ed un caso di default opzionale:

```
switch(valore)
{case '1':
   azione1;
 case '2':
   azione2;
 ...
 default:
   azione_default;
}
```

L'istruzione di selezione multipla switch (2)

```
scanf("%d", &scelta);
switch(scelta)
{
  case 1:
    printf("hai scelto 1");
    break;
  case 2:
    printf("hai scelto 2");
    break;
  case 3:
  case 4:
    printf("hai scelto 3 o 4");
    break;
  default:
    printf("hai scelto un valore >3");
    break;
}
```

non servono {} anche per istruzioni multiple

possibilità di case multipli

necessario, altrimenti esegue tutti i case dopo quello in cui entra

Esercizi (switch)

- Prendere in input un intero e stampare una stringa che dica se è positivo o negativo, pari o dispari.
- Prendere in input un intero da 1 a 10 e stampare il suo valore a lettere (3: tre)
- Prendere in input un intero da 1 a 12 e stampare il mese corrispondente (3: marzo)
- Prendere in input 3 valori, che rappresentano i coefficienti di un'equazione di secondo grado, e stampare i risultati dell'equazione.
- Prendere in input un intero tra 1 e 99 e stamparlo come numero romano.