

IL COSTRUTTO ITERAZIONE



Il costrutto iterazione

Consente di ripetere una operazione elementare o un blocco di operazioni.

Tipologie di iterazione

- Iterazione definita
- Iterazione condizionata

Tipologie di iterazione

- **Iterazione definita:** si conosce a priori quante volte l'azione elementare (o il blocco di istruzioni) verrà ripetuta. L'iterazione termina quando si è raggiunto il numero prefissato di ripetizioni.
- **Iterazione condizionata:** NON si conosce a priori quante volte l'azione elementare (o il blocco di istruzioni) verrà ripetuta. L'iterazione termina quando si verifica una certa condizione.
 - pre-condizionata: La condizione viene controllata prima delle istruzioni che quindi, potrebbero non essere mai eseguite.
 - post-condizionata: La condizione viene controllata dopo le istruzioni, che quindi vengono eseguite almeno una volta.

L'istruzione for

Nel linguaggio C il costrutto iterazione definito si implementa tramite l'istruzione **for**, che ha la seguente sintassi:

Viene eseguita solo una volta all'inizio del ciclo

Se la condizione è vera si entra nel ciclo, altrimenti si esce

Viene eseguita ad ogni iterazione e solo se condizione è vera

for (istruzione 1; condizione; istruzione 2)
istruzione o blocco;

ATTENZIONE: Non mettere mai il ; dopo le (), altrimenti l'istruzione o il blocco non saranno eseguiti.

L'istruzione for

i è una variabile contatore. Deve essere precedentemente dichiarata e viene qui inizializzata a 0

i<5 è una condizione, la cui verità viene controllata ad ogni iterazione. Se la condizione è vera si procede con le istruzioni, altrimenti si esce dal ciclo.

i++ è una istruzione di incremento del contatore. Si può scrivere anche i=i+1

```
for (i=0; i<5; i++)  
    istruzione o blocco;
```

La condizione

La condizione può essere:

- ❖ Una variabile singola \rightarrow come if aritmetico,
- ❖ Una proposizione elementare, costituita dal confronto, mediante operatori relazionali, tra due variabili o due espressioni,
- ❖ Una proposizione articolata, costituita dalla connessione di proposizioni elementari mediante connettivi logici.

Esempio

Visualizzare i primi 5 numeri naturali.

```
printf ("1, 2, 3, 4, 5");
```

Sembra più semplice ma
... se dovessimo
visualizzare i primi 50000
numeri naturali?

oppure

```
for (i=1; i<6; i++)  
cout << i;
```

Esempio

Visualizzare i primi 3 numeri naturali.

Eseguiamo il codice

```
for (i=1; i<4; i++)  
    cout << i;
```

i	condizione	a video
1	V	1
2	V	2
3	V	3
4	F	

L'istruzione for

Con lo standard C99, la variabile `i` può essere dichiarata anche dentro il `for`.

```
for (int i=0; i<5; i++)  
    cout >> i;
```

In questo caso `i` non può essere usata al di fuori del `for`.

```
for (int i=0; i<5; i++)  
    cout << "CIAO";  
cout >> i;
```

```
int i;  
for (i=0; i<5; i++)  
    cout << "CIAO";  
cout << i;
```

L'istruzione for

```
for (istruzione 1; condizione; istruzione 2)  
    istruzione o blocco;
```

Ciascuno dei termini presenti tra () può teoricamente essere omesso e sostituito da costrutti analoghi.

```
istruzione1
```

```
for (; condizione; istruzione 2)  
    istruzione o blocco;
```

```
int i=0;
```

```
for (; i<5; i++)  
    cout<<i;
```

L'istruzione for

```
for (istruzione 1; condizione; istruzione 2)  
    istruzione o blocco;
```

Ciascuno dei termini presenti tra () può teoricamente essere omesso e sostituito da costrutti analoghi.

```
for (istruzione1; condizione;)  
    {istruzione 2;  
    istruzione o blocco;}
```

```
for (i=0; i<5;)  
    {i++;  
    cout<<i;}
```

L'istruzione for

```
for (istruzione 1; condizione; istruzione 2)  
    istruzione o blocco;
```

Ciascuno dei termini presenti tra () può teoricamente essere omesso e sostituito da costrutti analoghi.

```
for (istruzione1;; istruzione 2)  
    {istruzione o blocco;  
      condizione  
      break;}
```

```
for (i=0;; i++)  
    { cout<<i;  
      if (i>=5)  
        break;}
```

L'istruzione for

for (**istruzione 1**; **condizione**; **istruzione 2**)
istruzione o blocco;

Ciascuno dei termini presenti tra () può teoricamente essere omesso e sostituito da costrutti analoghi.

for (;;)

E' un ciclo infinito perché la condizione è sempre verificata

ATTENZIONE: Anche quando si omette un termine tra () è indispensabile mettere il ; per soddisfare la sintassi del for.

L'istruzione for

for (**istruzione 1**; **condizione**; **istruzione 2**)
istruzione o blocco;

Ciascuno dei termini presenti tra () può teoricamente
essere omesso e sostituito da costrutti analoghi ...

... ma è bene lasciare tutto dentro il for ...

... e soprattutto è bene evitare i cicli infiniti!

L'istruzione for

```
char a;  
for (a='A'; a<='Z'; a++)  
    cout<<a;
```

```
float n;  
for (n=0; n<=1; n=n+0,1)  
    cout<<a;
```

L'istruzione for

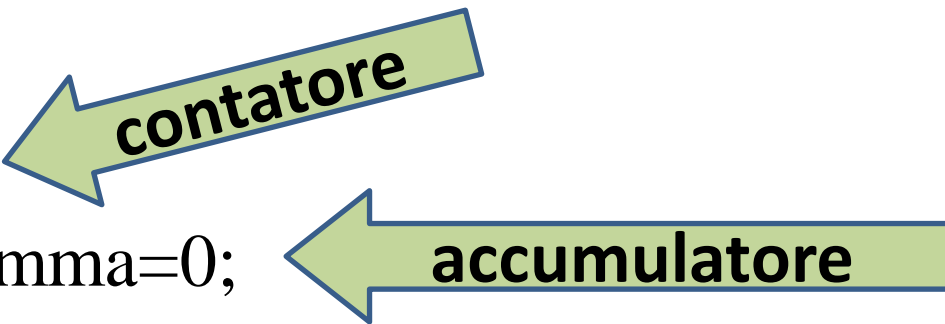
```
int n;  
for (n=10; n>=0; n--)  
    cout<<n;
```

```
int n;  
for (n=0; n<=10; n=n+2)  
    cout<<n;
```


Esempio

Eseguire la somma dei primi 50 numeri naturali.

```
int n;
int somma=0;
for (n=1; n<=50; n++)
    somma=somma+n;
cout<<somma;
```



The diagram consists of two green arrows with black outlines. The first arrow, labeled 'contatore', points from the right towards the variable 'n' in the first line of code. The second arrow, labeled 'accumulatore', points from the right towards the variable 'somma' in the second line of code.

Esempio

Stampare i numeri pari fino a n, con n acquisito da tastiera.

```
int n, i;  
  
cin>>n;  
for (i=0; i<=n; i++)  
    if (! (i%2))  
        cout<<i;
```

Esempio

Stampare i numeri pari fino a n, con n acquisito da tastiera.

```
int n,i;
```

```
cin>>n;
```

```
for (i=0; i<=n; i=i+2)
```

```
    cout<<i;
```

Esempio

Stampare la seguente figura:

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

```
int i,j;
```

```
for (i=0; i<3; i++)  
    { for(j=0; j<3; j++)  
        cout<<"*";  
      cout<<endl; }
```

L'istruzione while

Nel linguaggio C il costrutto iterazione pre-condizionato si implementa tramite l'istruzione **while**, che ha la seguente sintassi:

Il controllo della condizione viene effettuato prima dell'inizio del ciclo.
Se la condizione è vera si entra nel ciclo, altrimenti si esce

```
while (condizione)  
    istruzione o blocco;
```

ATTENZIONE: Non mettere mai il ; dopo le (), altrimenti l'istruzione o il blocco non saranno eseguiti.

L'istruzione while

$i < 20$ è una condizione, la cui verità viene controllata ad ogni iterazione. Se la condizione è vera si procede con le istruzioni, altrimenti si esce dal ciclo.

```
while ( $i < 20$ )  
    istruzione o blocco;
```

ATTENZIONE: In fase di scrittura del codice si deve sempre controllare che la condizione prima o poi venga soddisfatta, per evitare di cadere in cicli infiniti.

La condizione

La condizione può essere:

- ❖ Una variabile singola \rightarrow come if aritmetico,
- ❖ Una proposizione elementare, costituita dal confronto, mediante operatori relazionali, tra due variabili o due espressioni,
- ❖ Una proposizione articolata, costituita dalla connessione di proposizioni elementari mediante connettivi logici.

Esempio

Simulare un registratore di cassa che calcoli l'importo finale di una spesa. Il costo dei singoli prodotti è inserito da tastiera e termina quando l'importo totale supera una cifra prefissata TOT.

```
float prezzo;  
float somma=0;  
while(somma<TOT)  
    { cin>>prezzo;  
      somma=somma+prezzo; }
```


Esempio

Eseguiamo il codice

```
float prezzo;  
float somma=0;  
while(somma<TOT)  
    { cin>>prezzo;  
      somma=somma+prezzo; }
```

TOT

500

somma

0

100

350

380

507

prezzo

100

250

30

127

L'istruzione while

E' possibile simulare una while con un for:

```
while (condizione)  
    istruzione o blocco;
```

```
for (;condizione;)  
    istruzione o blocco;
```

L'istruzione while

Viceversa:

```
for (istruzione 1; condizione; istruzione 2)  
    istruzione o blocco;
```

```
istruzione 1;  
while (condizione)  
    {istruzione o blocco;  
    istruzione 2;} 
```

L'istruzione do while

Nel linguaggio C il costrutto iterazione post-condizionato si implementa tramite l'istruzione **do while**, che ha la seguente sintassi:

Il controllo della condizione è effettuato dopo l'inizio del ciclo.
L'istruzione viene eseguita almeno una volta e poi viene controllata la condizione.

do

istruzione o blocco;

while (**condizione**);

ATTENZIONE: Non mettere il ; dopo il do.
E' un errore di sintassi.

L'istruzione do while

0 è una condizione (come if aritmetico), la cui verità viene controllata al termine della prima iterazione. Se la condizione è vera si procede con le altre iterazioni, altrimenti si esce dal ciclo.

```
do  
    cout<<"CIAO";  
while (0);
```

ATTENZIONE: In fase di scrittura del codice si deve sempre controllare che la condizione prima o poi venga soddisfatta, per evitare di cadere in cicli infiniti.

La condizione

La condizione può essere:

- ❖ Una variabile singola \rightarrow come if aritmetico,
- ❖ Una proposizione elementare, costituita dal confronto, mediante operatori relazionali, tra due variabili o due espressioni,
- ❖ Una proposizione articolata, costituita dalla connessione di proposizioni elementari mediante connettivi logici.

Esempio

Eseguiamo il codice

a video

do

```
    cout<<"CIAO";
```

```
while (0);
```



CIAO

Esempio

Inserire una sequenza di numeri interi. Terminare l'inserimento premendo 0.

```
int a;  
do  
    cin>>a;  
while(a);
```


Esempio

Eseguiamo il codice

```
int a;  
do  
    cin>>a;  
while(a);
```

a

20

-31

7

-1

0

Esempio

Eseguiamo il codice

a

```
int a;  
do  
    cin>>a;  
while(a);
```



0

L'istruzione do while

E' possibile simulare un do while con un for:

do

 {istruzione o blocco;}

while (**condizione**)

istruzione o blocco;

for (;**condizione**;

 istruzione o blocco;

L'istruzione do while

Viceversa:

```
for (istruzione 1; condizione; istruzione 2)  
    istruzione o blocco;
```

```
    istruzione 1;  
do  
    if(!condizione)  
        break;  
    {istruzione o blocco;  
    istruzione 2;}  
while (condizione)
```

L'istruzione do while

E' possibile simulare un do while con un while:

do

{istruzione o blocco;}

while (**condizione**)

istruzione o blocco;

while (**condizione**)

istruzione o blocco;

L'istruzione do while

Viceversa:

```
while (condizione)  
    istruzione o blocco;
```

```
do  
    if(!condizione)  
        break;  
    {istruzione o blocco;}  
while (condizione)
```

Esempio

Inserire una sequenza di numeri interi. Terminare l'inserimento premendo 0. Calcolare la somma e la media della sequenza di numeri.

```
int a, somma, conta;
```

```
float media;
```



```
somma=conta=0;
```

```
media=(float)somma/conta;
```

```
do
```



```
    { cin>>a;
```

```
      somma=somma+a;
```

```
      conta++;
```

```
    }
```

```
while(a);
```

Esempio

Eseguiamo il codice

```
int a, somma, conta;  
float media;  
somma=conta=0;  
do  
    {cin>>a;  
    somma=somma+a;  
    conta++;  
    }  
while(a);
```

a	somma	conta
1	1	1
3	4	2
0	4	3
	media	1,3

☺ media=(float)somma/conta;☺

Esempio

Aggiustiamo ed eseguiamo il codice:

```
int a, somma, conta;
```

```
float media;
```

```
somma=conta=0;
```

```
do
```

```
    {cin>>a;
```

```
      somma=somma+a;
```

```
      conta++;
```

```
    }
```

```
while(a);
```

```
media=(float)somma/(conta-1);
```

a

1

3

0

somma

1

4

4

conta

1

2

3

media

2

Esempio

Inserire una sequenza di numeri interi. Terminare l'inserimento premendo 0. Calcolare la somma e la media dei numeri maggiori di 5.

```
int a, somma=0, conta=0;
```



```
float media;
```

```
media=(float)somma/conta;
```

```
do
```



```
    { cin>>a;
```

```
    if(a>5)
```

```
        { somma=somma+a;
```

```
          conta++;}
```

```
    }
```

```
while(a);
```

Esempio

Eseguiamo il codice

```
int a, somma, conta;  
float media;  
somma=conta=0;  
do  
    {cin>>a;  
    if(a>5)  
        {somma=somma+a;  
        conta++;}  
    }  
while(a);
```

a	somma	conta
-1	0	0
3	0	0
0	0	0
	media	NaN

☺ media=(float)somma/conta;☺

Esempio

Aggiustiamo ed eseguiamo il codice:

```
int a, somma, conta;
```

```
float media;
```

```
somma=conta=0;
```

```
do
```

```
    {cin>>a;
```

```
    if(a>5)
```

```
        {somma=somma+a;
```

```
        conta++;}
```

```
    }while(a);
```

```
if(!conta)
```

```
    media=(float)somma/conta;
```

a

somma

conta

1

0

0

3

0

0

0

0

0

media

