

La Programmazione strutturata

Nella Programmazione strutturata è ammesso solo l'utilizzo di strutture di controllo *one-in/one-out*

Non è ammesso 'entrare' nella struttura o 'uscire' dalla stessa da punti diversi da quelli di *in* e di *out*

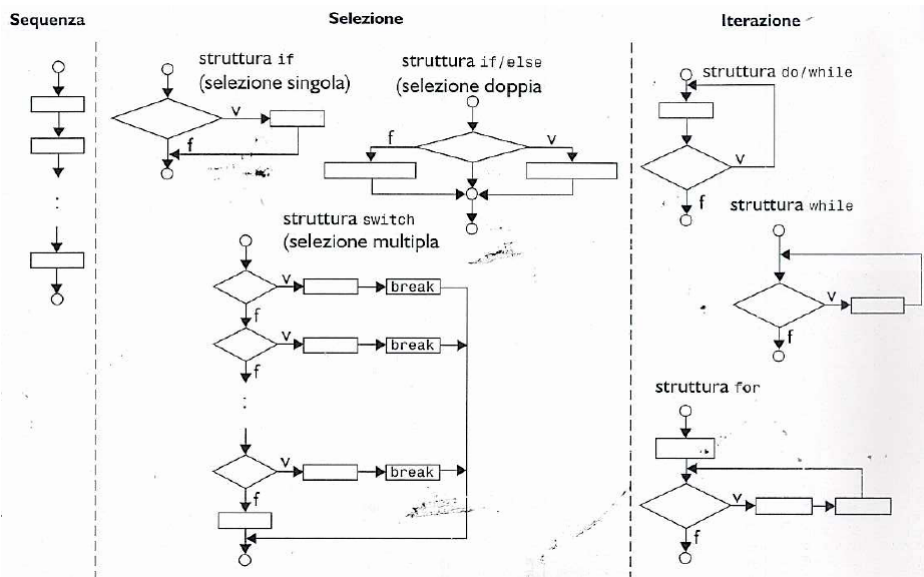
Una struttura di controllo può seguirne un'altra collegando il punto di *out* della prima a quello di *in* della seconda; in tal caso le strutture sono dette *'in sequenza'*

Una struttura di controllo può essere contenuta totalmente in un'altra ed in tal caso le strutture sono dette *'inestate'*

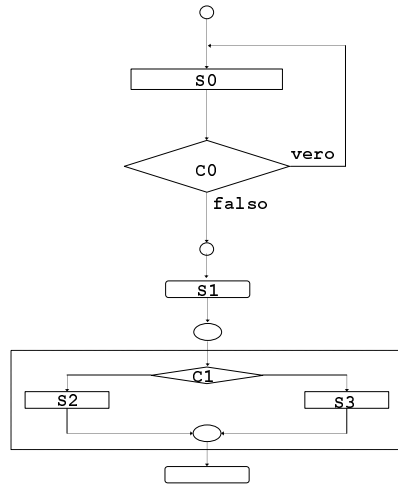
Le strutture di controllo non devono mai "intersecarsi"

Le strutture di controllo sono i "mattoncini" con cui costruire i programmi

Riepilogo strutture di controllo



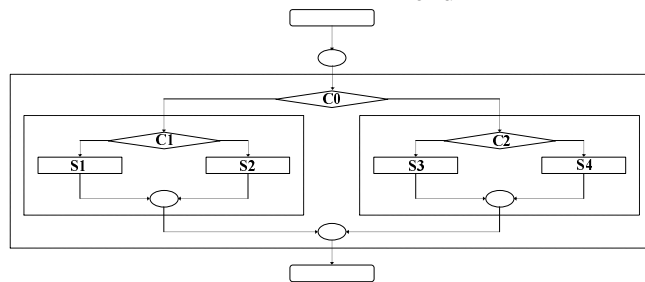
**Esempio:
Strutture in sequenza**



```
do
  S0
while (C0);
S1
if (C1) then
  S2
else
  S3
endif
```

**Esempio:
Strutture innestate**

```
if (C0) then
  if (C1) then S1
  else S2
endif
else
  if (C2) then S3
  else S4
endif
endif
```



Istruzioni NON strutturate

Istruzione di salto incondizionato **goto**

`goto <label>`

dove `label` è una etichetta che individua una istruzione nel codice

L'istruzione `goto` provoca il trasferimento incondizionato del flusso di controllo del programma all'istruzione identificata da `label`, saltando tutte le altre istruzioni comprese tra quella di `goto` e quella con la `label` indicata

E' la principale fonte di **non strutturazione** dei programmi: il suo uso rende i **programmi poco comprensibili, poco testabili e poco manutenibili**

NON DEVE MAI ESSERE USATA

lo si può evitare usando opportunamente le strutture di controllo

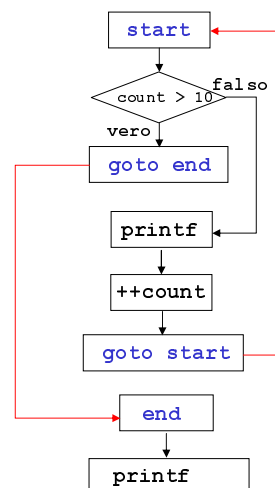
Esempio di programma con goto

```
/* Using goto */
#include <stdio.h>
main ()
{
    int count = 1;

    start:          /* label */
        if (count > 10)
            goto end;

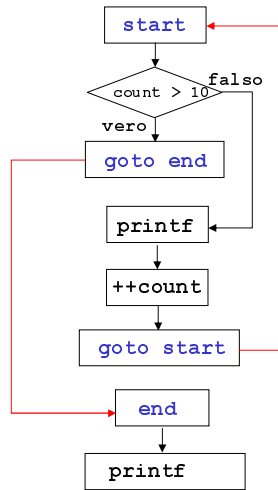
        printf("%d ", count);
        ++count;
        goto start;

    end:           /* label */
        printf('%d\n', count);
}
```



Notare come `start:`, `end:` e `goto` sono usate generando strutture NON 1-in/1-out

Esempio di programma con goto



Il flusso di controllo non è strutturato:
presenza di strutture NON one-in/one-out

La struttura if ha due punti di uscita su
due diverse istruzioni

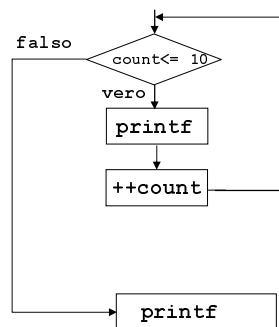
l'istruzione "goto start" implica un ciclo
(che però è di difficile individuazione), e
l'istruzione "goto end" rappresenta
l'uscita da tale ciclo

... il tutto per implementare un ciclo a
condizione iniziale !!!

Lo stesso programma in forma strutturata

```
#include <stdio.h>
main()
{
    int count = 1;

    while(count <= 10)
    {
        printf("%d ", count);
        ++count;
    }
    printf('%d\n', count);
}
```



Altre istruzioni NON strutturate

Istruzioni **break** e **continue**

sono usate all'interno di strutture cicliche per alterarne il flusso di controllo

break provoca l'uscita immediata dal ciclo

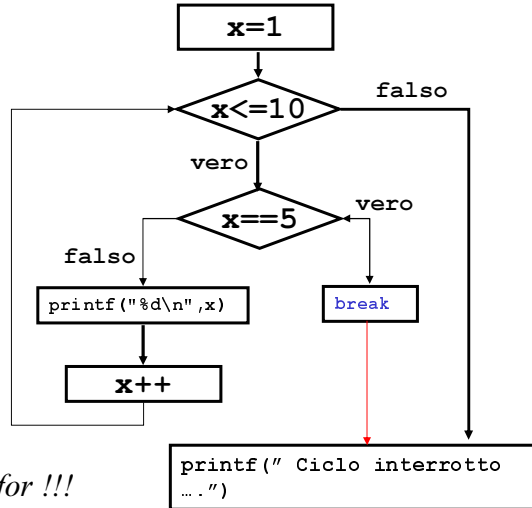
continue fa riportare il flusso di controllo immediatamente all'inizio del ciclo senza che le istruzioni che la seguono siano eseguite

Uso dell'istruzione **break** in una struttura **for**

```
#include <stdio.h>
main()
{
    int x;
    for (x = 1; x <= 10; x++)
    {
        if (x == 5)
            break; /* interrompe il ciclo se x == 5 */
        printf("%d ", x);
    }
    printf("\n Ciclo interrotto per x == %d\n", x);
}
```

Uso dell'istruzione break in una struttura for

```
#include <stdio.h>
main()
{
    int x;
    for (x = 1; x <= 10; x++)
    {
        if (x == 5)
            break;
        /* interrompe il ciclo se x == 5 */
        printf("%d ", x);
    }
    printf("\n Ciclo interrotto
per x == %d\n", x);
}
```



Due punti di uscita dal ciclo for !!!

Uso dell'istruzione continue in una struttura for

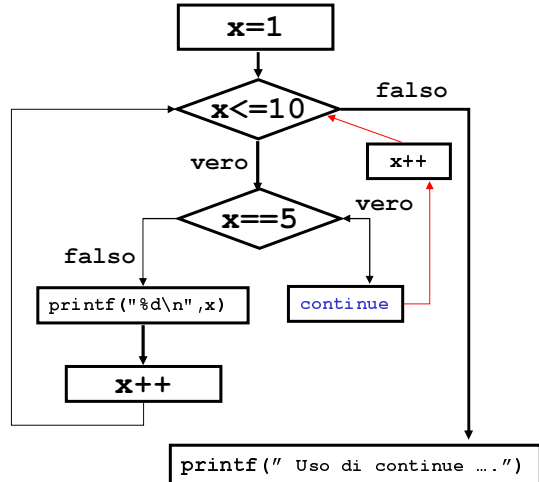
```
#include <stdio.h>
main()
{
    int x;
    for (x = 1; x <= 10; x++)
    {
        if (x == 5)
            continue; /* salta le successive istruzioni
tornando all'inizio del ciclo
se x == 5 */
        printf("%d ", x);
    }

    printf("\n Uso di continue per saltare la stampa
del valore 5\n");
}
```

Uso dell'istruzione continue in una struttura for

```
#include <stdio.h>
main()
{
    int x;
    for (x = 1; x <= 10; x++)
    {
        if (x == 5)
            continue;
        /* salta le successive
        istruzioni
        tornando all'inizio del ciclo
        se x == 5 */
        printf("%d ", x);
    }

    printf("\n Uso di continue
    per saltare la
    stampa del valore 5\n");
}
```

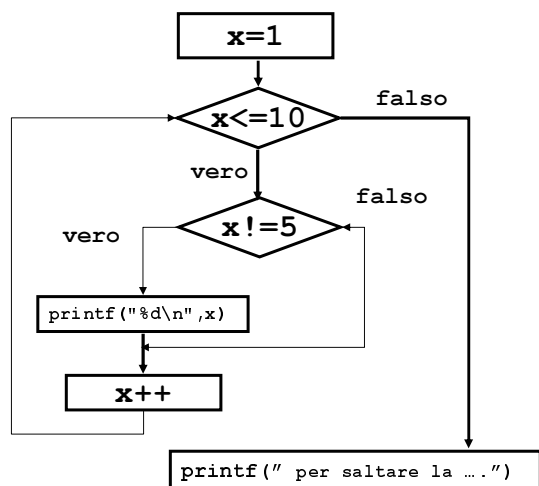


Due punti di ritorno al ciclo for !!!

Si può ottenere la stessa cosa senza continue strutturando opportunamente

```
#include <stdio.h>
main()
{
    int x;
    for (x = 1; x <= 10; x++)
    {
        if (x != 5)
            printf("%d ", x);
    }

    printf("\n per saltare la
    stampa del valore
    5\n");
}
```



Istruzioni NON strutturate

L'uso delle istruzioni NON Strutturate:

`goto <label>`

`continue`

`break`

va sempre evitato ed in particolare l'istruzione

`goto <label>`

NON DEVE MAI ESSERE USATA

E' sempre possibile costruire programmi strutturati usando opportunamente le strutture di controllo