STRUTTURE DI CONTROLLO

Dott. Franco Liberati


Argomenti
01
Controllo condizionale IF
Selezione doppia IF THEN ELSE
Annidamento IF THEN ELSE
02
Ripetizione WHILE
Ripetizione forzata DO WHILE
03
Iterazione FOR


Strutture di controllo

Strutture di controllo

- Un programma è formato da una sequenza di istruzioni eseguite una dopo l'altra
  - La CPU non ha una visione di insieme del programma: "vede" solo l'istruzione che deve eseguire e il contenuto dei registri
  - La CPU esegue una istruzione alla volta nell'ordine riportato in memoria

- Con **i salti** si **controlla l'ordine di esecuzione** perché rimandano il proseguimento di un programma all'indirizzo specificato
  - Se il **salto è condizionato** l'istruzione che è eseguita dopo un salto è quella che si trova all'indirizzo specificato come destinazione del salto se la condizione è vera; altrimenti si prosegue normalmente con l'istruzione successiva
  - Se il **salto è incondizionato** si prosegue all'indirizzo specificato senza valutare la verità di una condizione

Controllo condizionale
IF

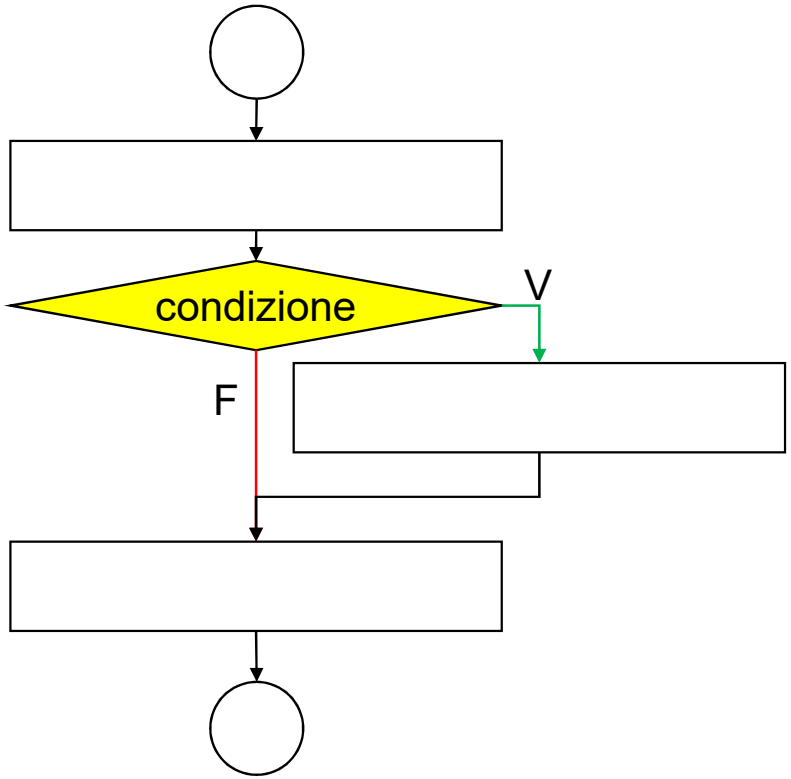
Controllo condizionale

IF (condizione) THEN istruzioni

Il controllo condizionale, ottenuto con l'impiego di una istruzione di salto condizionato, esegue un trasferimento del controllo del codice se una condizione è vera; oppure procede in modalità sequenziale, se la condizione è falsa

In altre parole la struttura di selezione è usata per prendere una decisione ovvero scegliere una alternativa

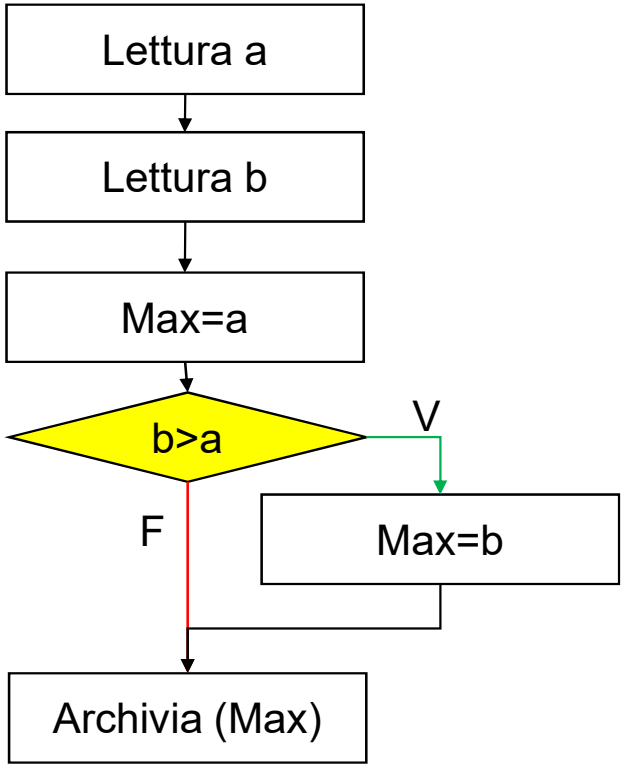
Formalmente il controllo condizionale è IF (Condizione) THEN {istruzioni}



Controllo condizionale

IF (esempio:calcolo massimo due operandi)

Calcolo del massimo di due operandi

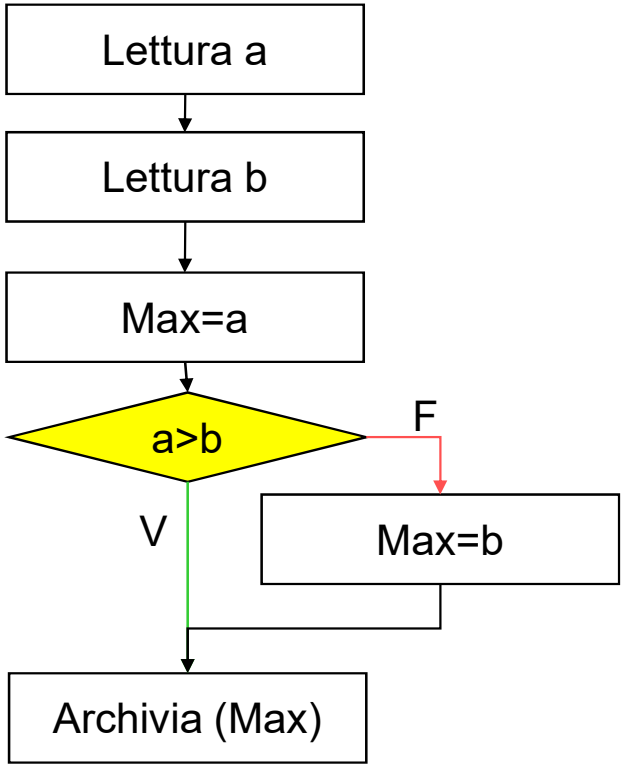



```
            .text
            .globl main
main:
            lw $t0,val_a        #lettura val_a
            lw $t1,val_b        #lettura val_b
            move $t2,$t0        #Max=a
            bgt $t1,$t0, DO_IF  #Se b>a salta e aggiorna Max
            j OUT_IF            #salta a fine
DO_IF:
            move $t2,$t1        #Max=b
OUT_IF:
            sw  $t2,massimo     #archiviazione del massimo
            li $v0,10           #terminazione del programma
            syscall

            .data
            val_a:.word 45
            val_b:.word 55
            massimo:.word 0
```

Controllo condizionale

IF (esempio:calcolo massimo due operandi-variante)

Calcolo del massimo di due operandi

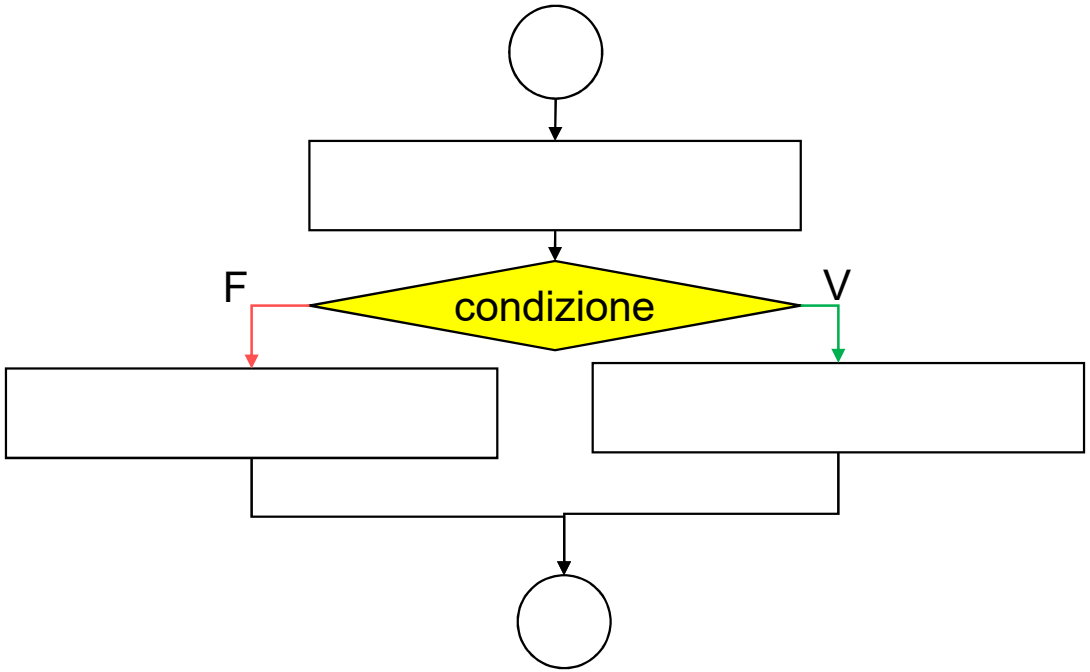



```
            .text
            .globl main
main:
            lw $t0,val_a        #lettura val_a
            lw $t1,val_b        #lettura val_b
            move $t2,$t0        #Max=a
            bgt $t0,$t1, NO_IF  #Se a>b salta e non aggiorna il Max
            move $t2,$t1        #Max=b
NO_IF:
            sw  $t2,massimo     #archiviazione del massimo
            li $v0,10           #terminazione del programma
            syscall

            .data
            val_a:.word 45
            val_b:.word 55
            massimo:.word 0
```

Selezione doppia

IF (condizione) THEN istruzioni ELSE istruzioni

La struttura di selezione doppia permette di specificare azioni differenti quando la condizione è vera e quando è falsa; cioè è usata per scegliere tra opzioni alternative. In altre parole una selezione doppia esegue un'azione se una condizione è vera e ne esegue un'altra se l'espressione è falsa



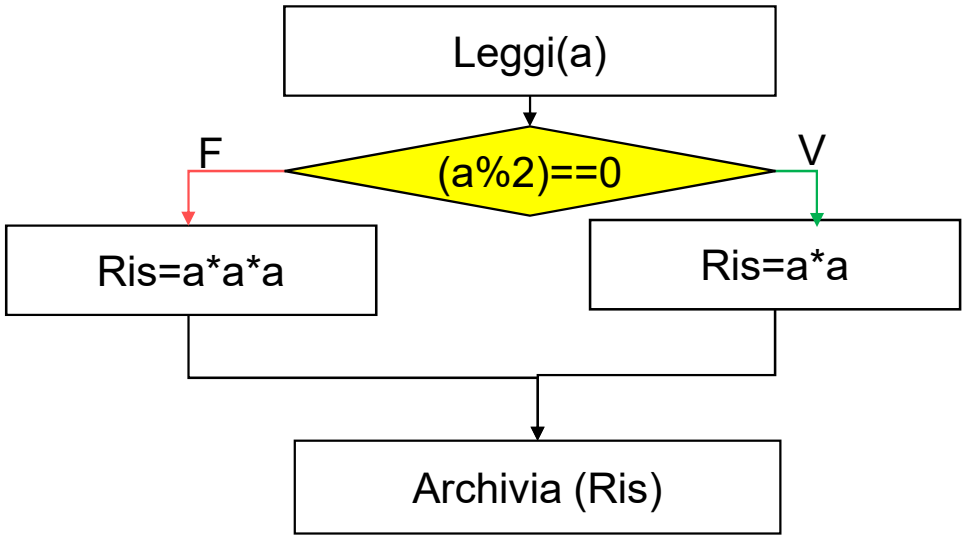
Il suo compito è pertanto la selezione tra due azioni differenti

Formalmente la selezione doppia è
IF (Condizione) THEN {istruzioni}
ELSE{istruzioni}

Selezione doppia

IF THEN ELSE (esempio)

Calcolo del quadrato se l'operando è un numero pari, del cubo se è un numero dispari

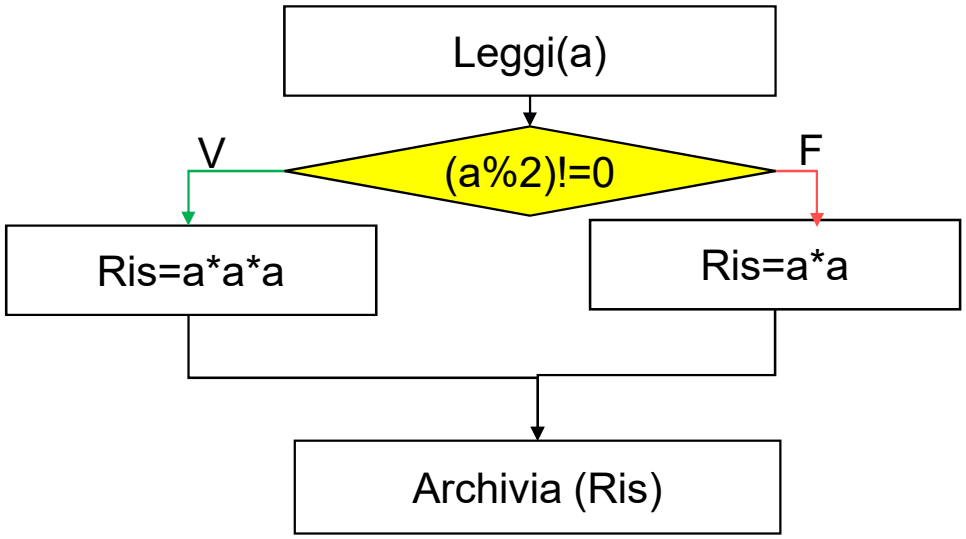



```
            .text
            .globl main
main:
            lw $t0,val_a            #lettura val_a
            rem $t1,$t0,2           #Calcolo della parità
            beqz $t1,THEN           #Se pari salto al ramo then…
            j ELSE                  #...altrimenti salto al ramo else
THEN:
            mul $t2,$t0,$t0         #calcolo quadrato
            j END_IF
ELSE:
            mul $t2,$t0,$t0          #calcolo cubo
            mul $t2,$t2,$t0         #
END_IF:
            sw  $t2,risultato       #archiviazione del massimo
            li $v0,10     #terminazione del programma
            syscall
            .data
            val_a:.word 45
            risultato:.word 0
```

Selezione doppia

IF THEN ELSE (esempio-variante con condizione negata)

Calcolo del quadrato se l'operando è un numero pari, del cubo se è un numero dispari

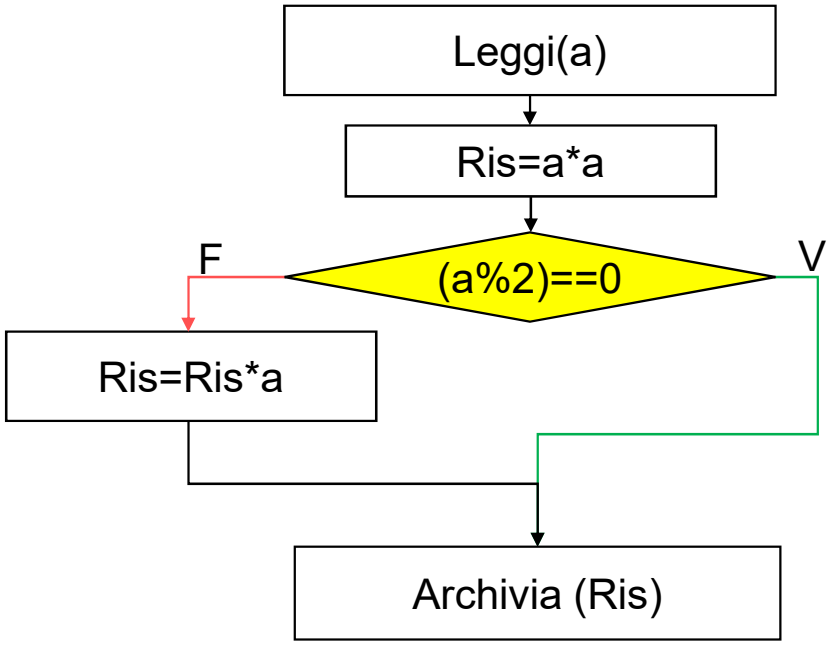



```
        .text
        .globl main
main:
        lw $t0,val_a        #lettura val_a
        rem $t1,$t0,2       #Calcolo della parità
        bnez $t1,ELSE       #Se pari salto al ramo then…
        mul $t2,$t0,$t0     #calcolo quadrato
        j END_IF
ELSE:
        mul $t2,$t0,$t0      #calcolo cubo
        mul $t2,$t2,$t0      #
END_IF:
        sw  $t2,risultato    #archiviazione del massimo
        li $v0,10    #terminazione del programma
        syscall
        .data
        val_a:.word 45
        risultato:.word 0
```

Selezione doppia

IF THEN ELSE (esempio-variante anticipo ramo then)

Calcolo del quadrato se l'operando è un numero pari, del cubo se è un numero dispari



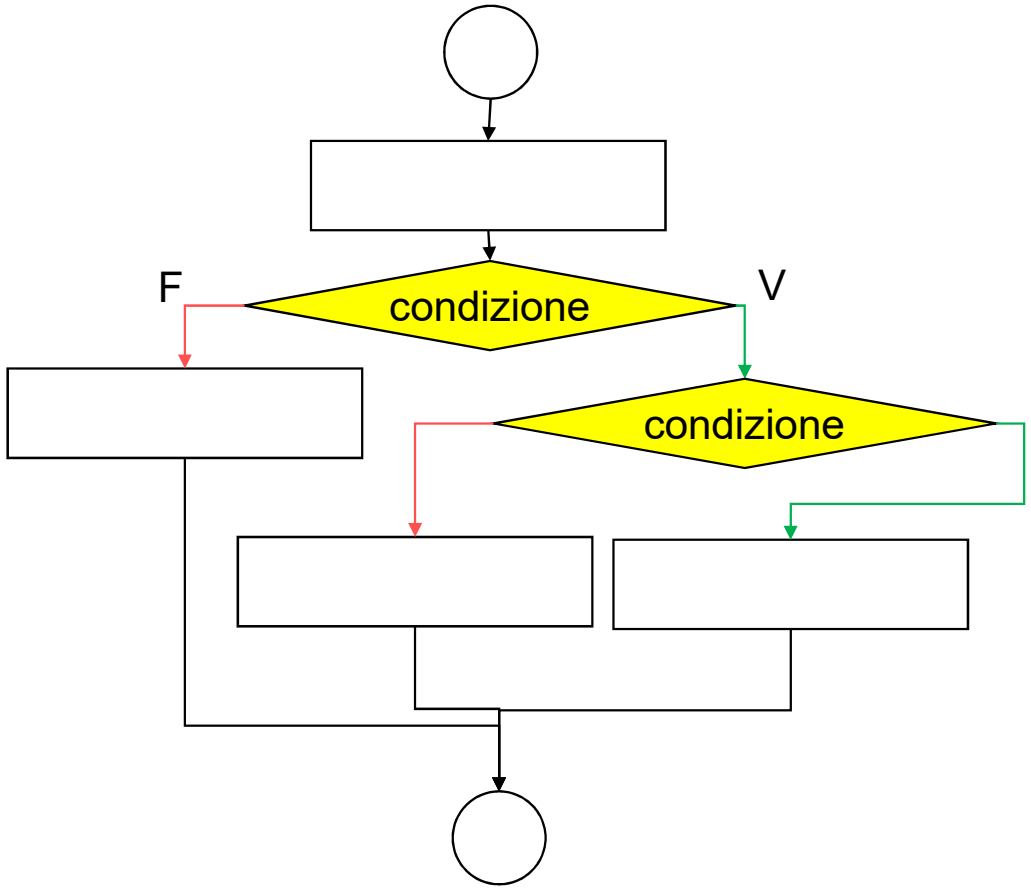

```
            .text
            .globl main
main:
            lw $t0,val_a          #lettura val_a
            rem $t1,$t0,2         #Calcolo della parità
            mul $t2,$t0,$t0       #calcolo quadrato
            beqz $t1,END_IF       #Se pari finisco
            mul $t2,$t0,$t0        #calcolo cubo
            mul $t2,$t2,$t0       #
END_IF:
            sw  $t2,risultato     #archiviazione del massimo
            li $v0,10    #terminazione del programma
            syscall


            .data
            val_a:.word 45
            risultato:.word 0
```

Annidamento IF THEN ELSE

La struttura a selezione singola e quella a selezione doppia possono essere usate contiguamente creando quello che si chiama **annidamento**.

Questa struttura, ad esempio, è impiegata quando bisogna effettuare test su casi multipli.

Non c'è uno schema definito perché l'annidamento è variabile ed è caratteristico dell'algoritmo in cui si concretizza.



Annidamento IF THEN ELSE

IF THEN ELSE (esempio-calcolo anno bisestile)

Calcolo se un operando corrisponde ad anno bisestile

```
BEGIN
lettura anno
if(anno%400==0) {bisestile=1;}
  else
  {
   if(anno%100==0){bisestile=0;}
      else
     {
       if(anno%4==0) {bisestile=1;}
          else {bisestile=0;}
      }
  }
END
```

```
            .text
            .globl main
main:
            lh $t0,anno
            rem $t1,$t0,400
            beqz $t1, THEN1
            j ELSE1
THEN1:      li $t3,1
            j END_IF
ELSE1:
            rem $t1,$t0,100
            beqz $t1, THEN2
            j ELSE2
THEN2:
            li $t3,0
            j END_IF
ELSE2:
            rem $t1,$t0,4
            beqz $t1, THEN3
            j ELSE3
```

```
THEN3:
            li $t3,1
            j END_IF
ELSE3:
             li $t3,0

END_IF:
            sb  $t3,bisestile
            li $v0,10
            syscall

            .data
            anno:.half 2024
            bisestile: .byte -1
```

Annidamento IF THEN ELSE

IF THEN ELSE (esercizio proposto per casa)

Calcolo se un operando corrisponde ad anno bisestile

OTTIMIZZARE IL CODICE

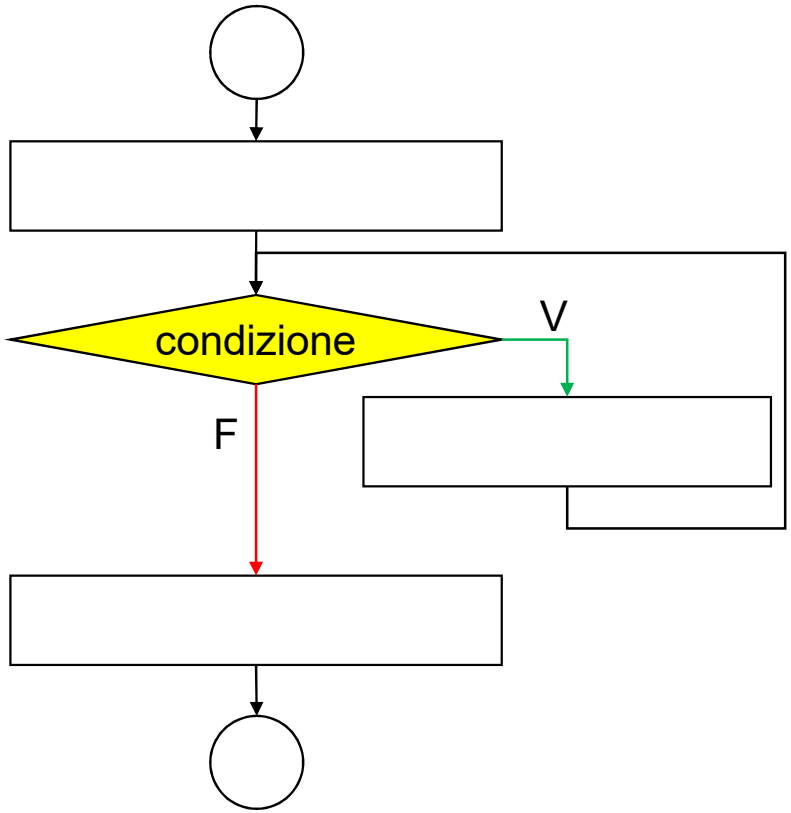
```
BEGIN
lettura anno
if(anno%400==0) {bisestile=1;}
  else
  {
   if(anno%100==0){bisestile=0;}
      else
     {
       if(anno%4==0) {bisestile=1;}
          else {bisestile=0;}
      }
  }
END
```

Ripetizione
WHILE

Ripetizione

WHILE (condizione) { istruzioni}

Nei programmi è essenziale la struttura di ripetizione WHILE poiché permette ripetere un'azione finché una condizione rimane vera
Quando la condizione diventa falsa si 'esce dal ciclo', ovvero non si ripete più il blocco iterativo e si procede eseguendo la prima istruzione dopo la struttura di iterazione.



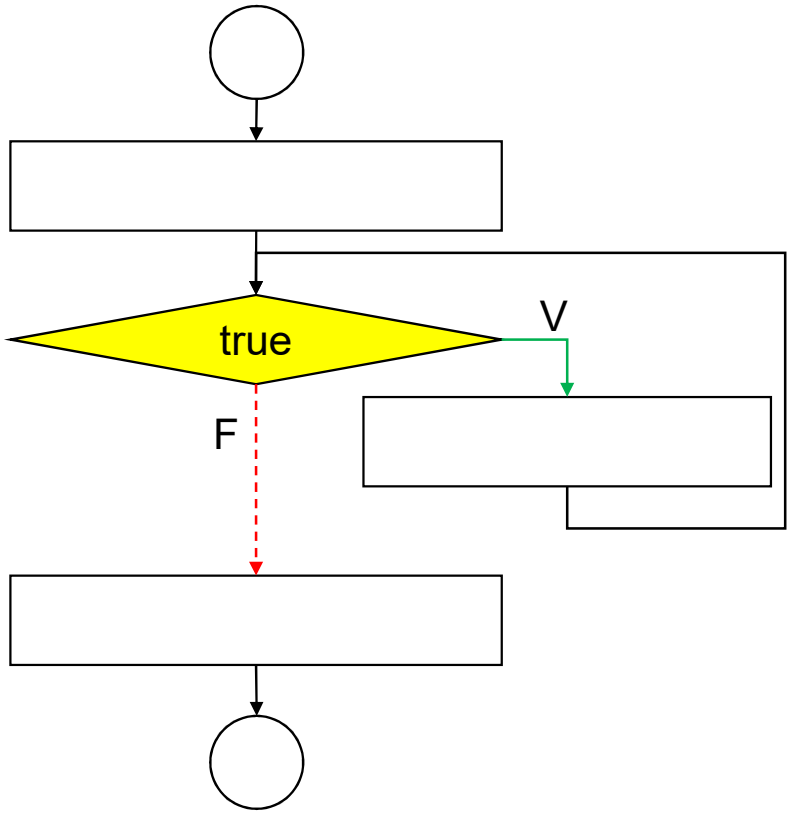
Formalmente una struttura di ripetizione è così descritta:

WHILE (condizione) {istruzioni}

Ripetizione

WHILE (condizione) { istruzioni}

L'impiego di questo costrutto può comportare un **ciclo infinito**, ovvero la condizione di continuazione non diventa mai falsa e quindi il programma ripete incessantemente il blocco delle istruzioni.
Un ciclo infinito è una condizione da scongiurare eccetto in alcuni (rari) casi



Ripetizione

WHILE (esempio: contatore di numeri)

Sommatoria di valori immessi da tastiera con terminazione quando il valore immesso è zero

```
BEGIN
totale=0
leggi x
while{x!=0)
{
        totale=totale+x;
        leggi x
}
END
```

```
.text
.globl main
main:
        li $t0,0            # inizializza il registro che ospiterà totale
        li $v0,5            # Servizio di lettura intero
        syscall             # Chiamata del servizio
        move $t1,$v0        # spostamento del valore letto da tastiera
WHILE:
        bnez $t1,DO_WHILE   #Esegue il CICLO WHILE se
                             #la condizione è vera
        j EXIT_WHILE
DO_WHILE:
        add $t0,$t0,$t1     # sommatoria
        li $v0,5            # Servizio di lettura intero
        syscall             # Chiamata del servizio
        move $t1,$v0        # spostamento del valore letto da tastiera
        j WHILE             # SALTO CICLO WHILE
EXIT_WHILE:
        sw $t0,totale       # Salva risultato in totale
        li $v0,10
        syscall
.data
        totale: .word 0
```

Ripetizione

WHILE (esempio: contatore di numeri - variante)

Sommatoria di valori immessi da tastiera con terminazione quando il valore immesso è zero

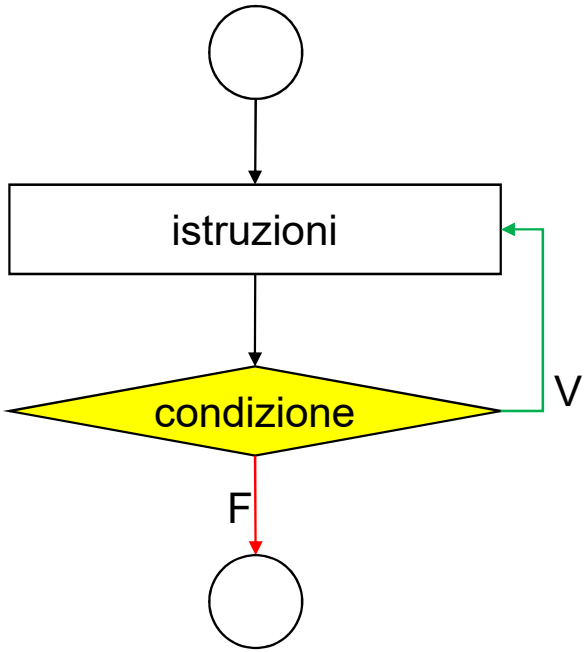
```
BEGIN
totale=0
leggi x
while{x!=0)
{
        totale=totale+x;
        leggi x
}
END
```

```
.text
.globl main
main:
        li $t0,0            # inizializza il registro che ospiterà totale
        li $v0,5            # Servizio di lettura intero
        syscall             # Chiamata del servizio
        move $t1,$v0        # spostamento del valore letto da tastiera
WHILE:
        beqz $t1,END_WHILE  #Esce dal CICLO WHILE se
                            #la condizione è falsa
        add $t0,$t0,$t1     # sommatoria
        li $v0,5            # Servizio di lettura intero
        syscall             # Chiamata del servizio
        move $t1,$v0        # spostamento del valore letto da tastiera
        j WHILE             # SALTO CICLO WHILE
END_WHILE:
        sw $t0,totale       # Salva risultato in totale
        li $v0,10
        syscall
.data
totale: .word 0
```

Ripetizione forzata

DO { istruzioni} WHILE (condizione)

In questo caso si verifica la condizione di continuazione dopo l'esecuzione del corpo del ciclo. Pertanto, il blocco delle istruzioni è sempre eseguito almeno una volta; e quando la condizione non è verificata l'esecuzione prosegue normalmente



Formalmente una struttura di ripetizione forzata è così descritta:

DO {istruzioni} WHILE (condizione)

Ripetizione forzata

DO WHILE (esempio)

Indovina un numero magico compreso tra 0 e 5

```
Begin
System_Random(Magic;0,5);
do
{
Insert(operando);
} while (operando!=Magic)
End
```

```
.text
.globl main
main:
        li $v0,42   #Richiesta servizio di generazione di un numero
                    #causale
        li $a0,1    #Identificatore del numero casuale
        li $a1,5    #Estremo superiore: range[0;5]
        syscall     #Attivazione del servizio
        move $s0,$a0 #In $s0 è copiato il numero aleatorio (Magic)
DO:
        li $v0,5    #Richiesta servizio lettura di un intero da tastiera
        syscall     #Attivazione del servizio
        move $t0,$v0 #Spostamento operando inserito dall'utente
        bne $t0,$s0,DO      #Ripetizione del ciclo se il valore
                            #immesso dall'utente è diverso
                            #dal numero aleatorio
        li $v0,10
        syscall
```

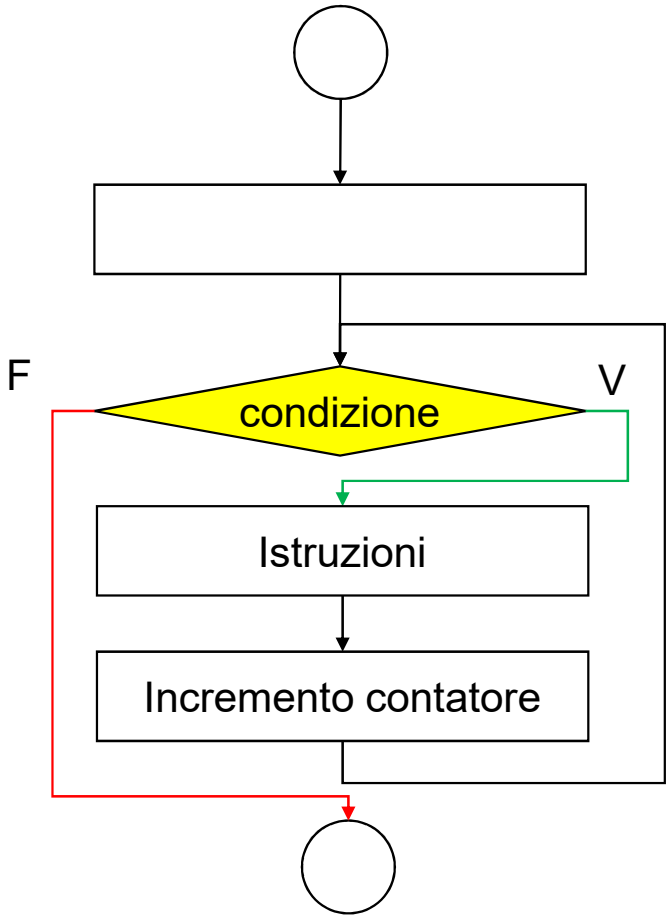

Iterazione
FOR

Iterazione

FOR

La struttura FOR gestisce una iterazione controllata da un contatore. Quindi il numero di iterazioni è strettamente correlato alla veridicità di una condizione in cui il contatore svolge un ruolo determinante

In altre parole il FOR esegue n volte un blocco di istruzioni



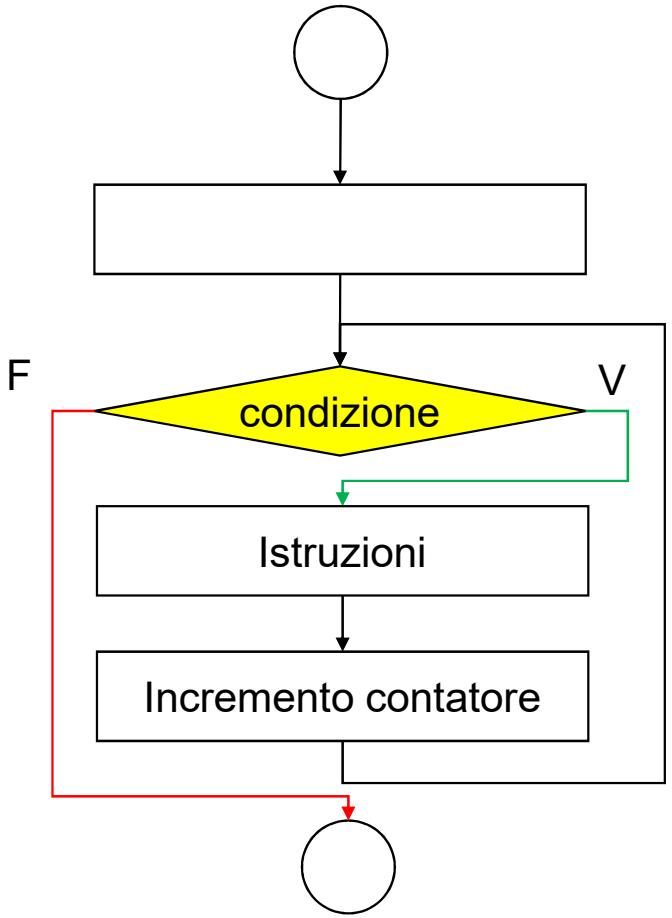
Iterazione

FOR

Formalmente una struttura di iterazione controllata da un contatore è così descritta:

FOR (inizializzazione *contatore*; condizione; incremento *contatore*) {istruzioni}

La struttura FOR inizializza il contatore di controllo ad un valore prestabilito (contatore=*init*). In seguito è analizzato il presupposto di continuazione del ciclo (condizione). Se la condizione è soddisfatta si esegue l'istruzione, o il blocco di istruzioni, associato. Il valore del contatore è poi manipolato (incremento) e si valuta nuovamente la condizione. L'iterazione termina quando la condizione non è verificata, cioè nel caso di fallimento del test di continuazione



Iterazione

FOR (esempio: media sei valori)

Media (per interi) di 6 numeri

```
BEGIN
tot=0;
for (cont=0;cont<6; cont=cont+1)
{
  Read(operando);
  tot=tot+operando;
}
media=tot/6;
END
```

```
.eqv CONT $t0 #Associazione della variabile contatore al registro $s0
.eqv LIMITE $t1 #Associazione della variabile limite al registro $s1
.eqv INCR $t2  #Associazione della variabile incremento al registro $s2

.text
.globl main
main:
            li CONT,0     #Inizializzazione del contatore
            li LIMITE,6   #Estremo superiore del numero di iterazioni da fare
            li INCR,1     #Assegnazione del passo di incremento
            li $t4,0      #Inizializzazione della variabile tot
FOR:        bgeCONT, LIMITE,END_FOR  #Analisi del contatore: il superamento del limite
                                     #conclude il FOR
            li $v0,5      #Richiesta del servizio di lettura di un intero da tastiera
            syscall       #Attivazione del servizio

            move $t3,$v0              #Copia dell'operando immesso dall'utente
            add $t4, $t4,$t3          #Calcolo tot=tot+val
            add CONT,CONT, INCR #Incremento contatore
            j FOR                     #Ripetizione del ciclo
END_FOR:
            div $t6,$t4,6  #Calcolo della media (arrotondamento all'intero superiore)
            sw $t6, media
            li $v0,10
            syscall
.data
media: .word 0
```

FINE