Istruzioni per il controllo di ciclo - ciclo a condizione generica

Permette di ripetere l'esecuzione di un blocco di istruzioni finchè non viene verificata una condizione logica.

Sintassi

```
DO
  istruzione_1
     ...
  istruzione_k
     ...
  IF(condizione) EXIT
     ...
  istruzione_k+1
     ...
  istruzione_n
END DO
```

Problema

Leggere da input un insieme di numeri reali e calcolarne la media. Non si conosce in anticipo la quantità di valori da leggere, che comunque è limitata ad un massimo di 50; la lettura di un valore < 0 indica che l'insieme da leggere è terminato.

```
INTEGER :: cont
REAL :: x, somma, media
cont=0
somma=0.
DO
  WRITE(*,*) "Valore: "
  READ(*,*) x
  ! Condizione di uscita
  IF (x<0.) EXIT
  cont=cont+1
  somma=somma+x
  IF (cont>=50) EXIT
END DO
IF(cont>0)
  media=somma/cont
```

Note

Con questo costrutto, la verifica della condizione di uscita dal ciclo può essere posta in un punto qualunque del blocco. Inoltre, non è detto che sia unica.

Sono caratteristiche positive ?

Non è facile individuare nel ciclo le condizioni di uscita. Codice non facilmente comprensibile.

A tempo di esecuzione non è desumibile per quale condizione il ciclo è terminato e quindi quali istruzioni sono state eseguite. Incertezza sul punto di uscita e sulla condizione di uscita dal ciclo

Quindi...

È meglio avere un'unica condizione di uscita.

È meglio avere una condizione di uscita all'inizio o alla fine del costrutto.

Esempio: come ristrutturare un ciclo

```
cont=0
somma=0.

DO
  WRITE(*,*) "Valore: "
  READ(*,*) x
  ! 1a Condizione di uscita
  IF (x<0.) EXIT
  cont=cont+1
  somma=somma+x
  ! 2a Condizione di uscita
  IF (cont>=50) EXIT
END DO
```

```
cont=0
somma=0.

WRITE(*,*) "Valore: "
READ(*,*) x
DO
  IF (x<0.).OR.(cont>=10) EXIT
  cont=cont+1
  somma=somma+x
  WRITE(*,*) "Valore: "
  READ(*,*) x
END DO
```

Istruzioni per il controllo di ciclo - ciclo WHILE

Permette di ripetere l'esecuzione di un blocco di istruzioni finchè viene verificata una condizione logica valutata all'inizio del ciclo.

Sintassi

```
DO WHILE(condizione)
  istruzione_1
  istruzione_2
     ...
  istruzione_n
END DO
```

E' equivalente a:

```
DO
  IF (.NOT. condizione) EXIT
  istruzione_1
  istruzione_2
     ...
  istruzione_n
END DO
```

Esempio di utilizzo del WHILE

```
cont=0
somma=0.

WRITE(*,*) "Valore: "
READ(*,*) x
DO
  IF (x<0.).OR.(cont>=50) EXIT
  cont=cont+1
  somma=somma+x
  WRITE(*,*) "Valore: "
  READ(*,*) x
END DO
```

```
cont=0
somma=0.

WRITE(*,*) "Valore: "
READ(*,*) x
DO WHILE ((x>=0.).AND.(cont<50))
  cont=cont+1
  somma=somma+x
  WRITE(*,*) "Valore: "
  READ(*,*) x
END DO
```

Problema 1

Leggere da input un insieme di numeri reali >= 0 e determinare il valore massimo. Non si conosce in anticipo la quantità di valori da leggere; la lettura di un valore < 0 indica che l'insieme da leggere è terminato.

Problema 2

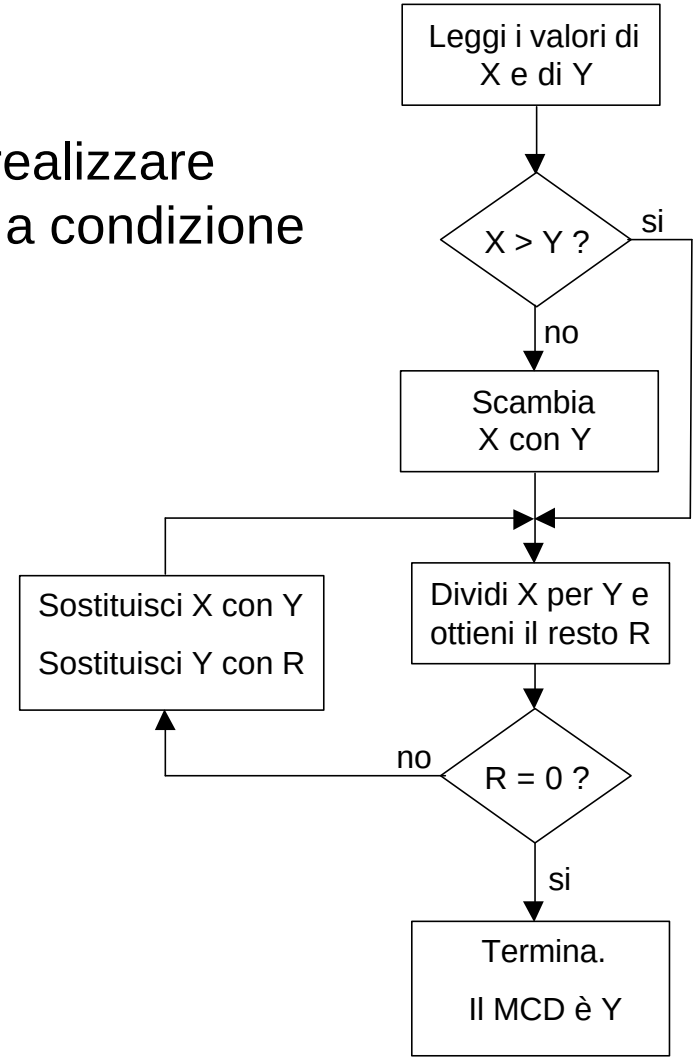
Nelle stesse ipotesi del problema 1, determinare il valore minimo dell'insieme dei valori letti.

```
PROGRAM Cercmax1

  INTEGER :: cont,maxnum
  REAL :: x, max

  PARAMETER(maxnum=10)

  cont=0
  max=0.

  WRITE(*,*) "Valore: "
  READ(*,*) x

  DO WHILE ((x>=0.).AND.(cont<maxnum))
    cont=cont+1
    IF (max < x) THEN
      max=x
    END IF

    WRITE(*,*) "Valore: "
    READ(*,*) x
  END DO

  WRITE(*,*) "Numero valori letti: ",cont
  WRITE(*,*)
  IF (cont>0) THEN
    WRITE(*,*) "Il valor massimo e': ",max
  END IF

END PROGRAM
```

```
PROGRAM Cercmax2

  INTEGER :: cont,maxnum
  REAL :: x, max

  PARAMETER(maxnum=10)

  cont=0
  max=0.

  DO
    WRITE(*,*) "Valore: "
    READ(*,*) x
    cont=cont+1

    IF (max < x) THEN
      max=x
    END IF

    IF ((x<0.).OR.(cont>=maxnum)) EXIT
  END DO

  WRITE(*,*) "Numero valori letti: ",cont
  WRITE(*,*)
  IF (cont>0) THEN
    WRITE(*,*) "Il valor massimo e': ",max
  END IF

END PROGRAM
```

```
PROGRAM Cercmin
  IMPLICIT NONE

  INTEGER :: cont,maxnum
  REAL :: x,min

  PARAMETER (maxnum=50)


  WRITE(*,*) "Valore: "
  READ(*,*) x

  IF (x>=0.) THEN
    cont=1
    min=x
    WRITE(*,*) "Valore: "
    READ(*,*) x

    DO WHILE((x>=0.).AND.(cont<maxnum))
      cont=cont+1
      IF(x<min) min=x
      WRITE(*,*) "Valore: "
      READ(*,*) x
      WRITE(*,*) x
    END DO

    WRITE(*,*)"Numero valori letti: ",cont
    WRITE(*,*)"Il valore minimo e': ",min
  ELSE
    WRITE(*,*)"Numero valori letti: 0"
  END IF

  WRITE(*,*) "Premi ENTER per terminare..."
  READ(*,*)
END PROGRAM
```

Un programma da realizzare utilizzando un ciclo a condizione



```
PROGRAM MCD
  ! Calcola il MCD tra due valori interi
  ! Utilizza l'algoritmo di Euclide

  IMPLICIT NONE

  INTEGER :: x,y,r,appo
  LOGICAL :: finito

  WRITE(*,*) "Dammi i valori: "
  READ(*,*) x,y
  WRITE(*,*)

  ! Scambia x ed y se y > x
  IF (y>x) THEN
    appo=x
    x=y
    y=appo
  END IF

  finito=.FALSE.
  DO
    r=MOD(x,y)
    ! Stampa di controllo
    WRITE(*,*) "x= ",x,"  y= ",y,"  r= ",r
    IF (r/=0) THEN
      x=y
      y=r
    ELSE
      finito=.TRUE.
    END IF

    ! Verifica la condizione di fine ciclo
    IF (finito) EXIT
  END DO

  WRITE(*,*) "Il MCD e' ",y
  WRITE(*,*) "Premi ENTER per terminare"
  READ(*,*)
END PROGRAM
```