

Titolo unità didattica: Costrutti di controllo

[04]

Titolo modulo : Costrutti di selezione

[01-T]

Caratteristiche generali dei costrutti di selezione

Argomenti trattati:

- ✓ costrutto di selezione a due vie (**if-then-else**)
- ✓ costrutto di selezione **if-then**
- ✓ predicati
- ✓ costrutti di selezione nidificati
- ✓ costrutto di selezione a più vie

Prerequisiti richiesti: P1-02-*-T

un **costrutto di selezione** denota la **scelta** (**selezione**) tra due sequenze di istruzioni, in dipendenza del valore di una **condizione** (**predicato**)

un **costrutto di selezione** (a due vie) deve specificare

- ✓ una **condizione** (predicato) che può assumere valore vero o valore falso (*tertium non datur*)
- ✓ **due** sequenze di istruzioni

denota la scelta tra due sequenze computazionali

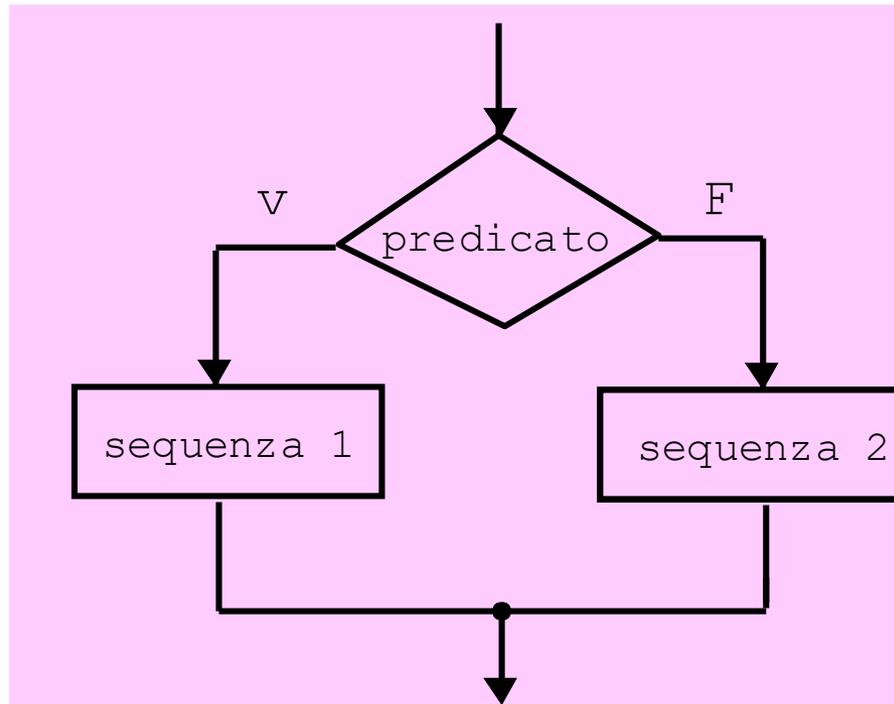
costrutto di selezione **if-then-else**

```
if (<predicato>
    {<blocco del then>}
else
    {<blocco dell' else>}
```

<blocco del then> , <blocco dell' else>
sono due sequenze di istruzioni

costrutto di selezione if-then-else

```
if (<predicato>)  
    {<sequenza 1>}  
else  
    {<sequenza 2>}
```



Esempio:

algoritmo per il calcolo del **valore assoluto** (o **modulo**) di un numero x

il valore assoluto si indica in Matematica con $|x|$ ed è definito da:

$$|x| = \begin{cases} x & \text{se } x \geq 0 \\ -x & \text{se } x < 0 \end{cases}$$

Esempio:

algoritmo per il calcolo del **valore assoluto**
(o **modulo**) di un numero x

dato di input: il numero x (variabile **x**)

dato di output: il numero $|x|$
(variabile **val_assoluto**)

dato di input: dal dispositivo di input

dato di output: sul dispositivo di output

Esempio:

algoritmo per il calcolo del valore assoluto di un numero

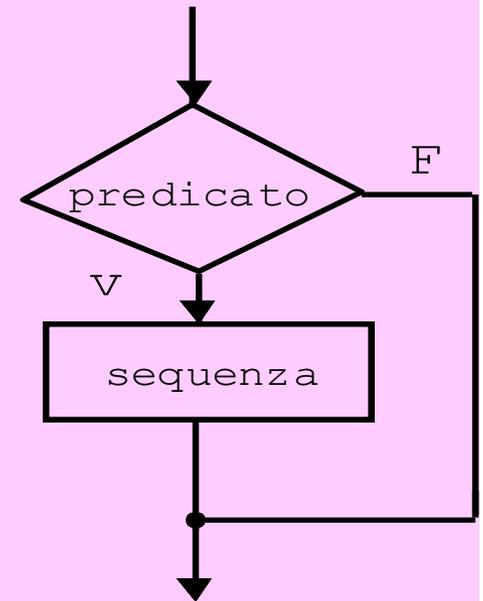
```
{  
float x, val_assoluto;  
read (x) ;  
if (x >= 0)  
    {val_assoluto = x ; }  
else  
    { val_assoluto = -x ; }  
printf (val_assoluto);  
}
```

```
{  
float x, val_assoluto;  
read (x) ;  
val_assoluto = x ;  
if (x < 0)  
    { val_assoluto = -x ; }  
printf (val_assoluto);  
}
```

versione alternativa

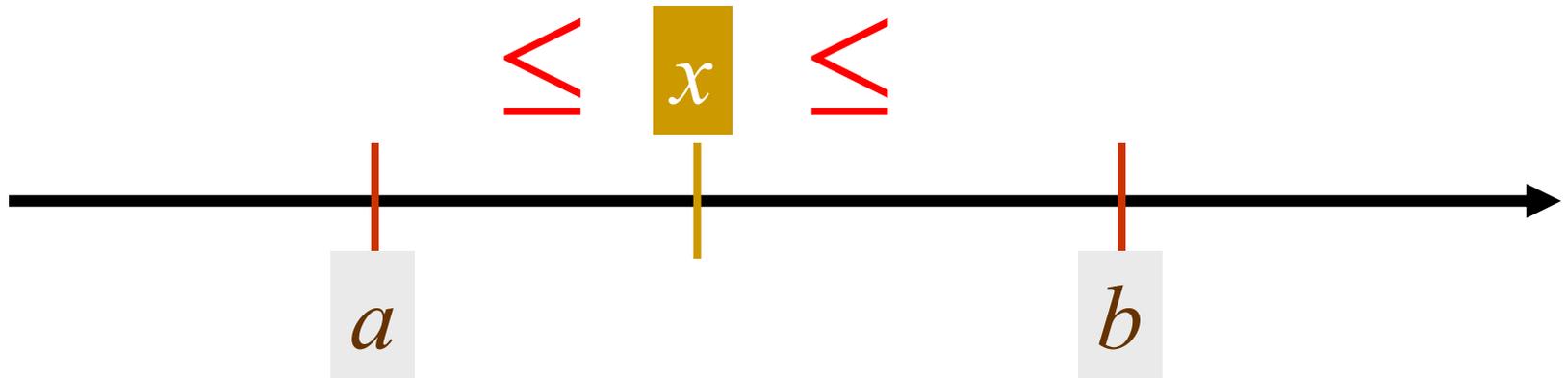
costrutto di selezione **if-then**

```
if <predicato>  
    {<blocco del then>}
```



Esempio:

algoritmo per determinare se un numero x
appartiene a un intervallo $[a,b]$



x deve essere maggiore o uguale ad a
e **contemporaneamente**
minore o uguale a b

Esempio:

algoritmo per determinare se un numero x appartiene a un intervallo $[a,b]$

dati di input: il numero x (variabile x), il numero a (variabile a), il numero b (variabile b)

dato di output: $true$ (1) oppure $false$ (0)
(variabile $appartiene$)

dati di input: dal dispositivo di input

dato di output: sul dispositivo di output

Esempio:

algoritmo per determinare se un numero x
appartiene a un intervallo $[a,b]$

```
{  
float x,a,b;  
logical appartiene;  
read (x,a,b);  
if (x >= a && x <= b )  
    { appartiene = true ; }  
    else  
    { appartiene = false ; }  
printf (x,a,b,appartiene) ;  
}
```

predicato

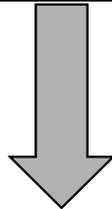
Esempio:

algoritmo per determinare se un numero x
appartiene a un intervallo $[a,b]$

```
{  
float x,a,b;  
int appartiene;  
read (x,a,b);  
  if (x >= a && x <= b )  
      { appartiene = 1 ; }  
  else  
      { appartiene = 0 ; }  
printf (x,a,b,appartiene) ;  
}
```

costrutti di selezione nidificati

```
if (<predicato 1>
  { <blocco del then 1> }
else { if (predicato 2>
  { <blocco del then 2> }
  else
  { <blocco dell'else 2> }
}
```



selezione a **tre** vie

Esempio: calcolare il
numero a :

$$a = \begin{cases} 1 & \text{se } x > 0 \\ 0 & \text{se } x = 0 \\ -1 & \text{se } x < 0 \end{cases}$$

```
if (x>0)
  { a = 1 ; }
else if (x==0)
  { a = 0 ; }
else
  { a = -1 ; }
```

```
if (x > 0)
    { a = 1; }
else if ( x == 0 )
    { a = 0; }
else
    { a = -1; }
```

sequenze di operazioni

se il valore di x è >0 : $x > 0$, $a = 1$

se il valore di x è $==0$: $x > 0$, $x == 0$, $a = 0$

se il valore di x è <0 : $x > 0$, $x == 0$, $a = -1$

Esempio:

algoritmo per determinare se un numero intero x è positivo, nullo o negativo

dato di input: il numero x (variabile x)

dato di output: nessuno: si visualizza (in alternativa): “il numero è positivo”, “il numero è nullo”, “il numero è negativo”

dato di input: dal dispositivo di input

output: sul dispositivo di output

Esempio:

algoritmo per determinare se un numero intero x è positivo, nullo o negativo

```
{  
int x;  
read (x);  
  if (x > 0)  
    { print (“il numero è positivo”); }  
  else if (x == 0)  
    { print (“il numero è nullo”); }  
  else  
    { print (“il numero è negativo”); }  
}
```

costrutti di selezione a **n** vie

```
switch (<variabile>) {  
  case <val_1> : <blocco del caso 1>  
  case <val_2> : <blocco del caso 2>  
  ..  
  case <val_n> : <blocco del caso n>  
  default: <blocco del caso default>  
}
```

se <**variabile**> ha valore <val_1> si eseguono (solo) le istruzioni del <blocco del caso 1>

se <**variabile**> ha valore <val_2> si eseguono (solo) le istruzioni del <blocco del caso 2>

costrutti di selezione a **n** vie

```
switch (<variabile>) {  
  case <val_1> : <blocco del caso 1>  
  case <val_2> : <blocco del caso 2>  
  ..  
  case <val_n> : <blocco del caso n>  
  default: <blocco del caso default>  
}
```

se <**variabile**> ha valore <val_**n**> si eseguono (solo) le istruzioni del <blocco del caso **n**>

se <**variabile**> ha un **qualsiasi altro** valore si eseguono le istruzioni del <blocco del **default**>

Esempio:

algoritmo per determinare se una città x (tra quattro città prefissate) è quella desiderata

dato di input: la città x (variabile x)

dato di output: nessuno: si visualizza (in alternativa): “risposta esatta”, un messaggio 1, un messaggio 2, un messaggio 3, un messaggio di errore

dato di input: dal dispositivo di input

output: sul dispositivo di output

```
{
char x;
printf (" dire quale tra Napoli (N), Caserta (C),
        Avellino (A), Benevento (B) è più a nord" );
read (x) ;
switch (x) {
    case 'N' : printf ("Napoli è la più a sud
                      delle 4");
    case 'C' : printf ("Caserta è più a sud
                      solo di Benevento");
    case 'A' : printf ("Avellino è più a nord
                      solo di Napoli");
    case 'B' : printf ("risposta esatta");
    default: printf ("errore nell' inserimento della
                      citta" );
}
}
```

una **sentenza** è una affermazione che può essere vera o falsa

sentenza **A**: Napoli è il capoluogo della Campania

sentenza **B**: $7 > 10$

sentenza **C**: la Terra è più grande di Urano

un **predicato** è una struttura logica che si ottiene connettendo sentenze

➤ negazione (**not**, **!**): \neg

➤ congiunzione (**and**, **&&**): \wedge

➤ disgiunzione (**or**, **||**): \vee

Esempi:

predicato $\neg A$: Napoli **non** è il capoluogo della Campania **falso**

predicato $\neg B$: $7 \leq 10$ **vero**

predicato $A \wedge B$: Napoli è il capoluogo della Campania **and** $7 > 10$ **falso**

predicato $B \vee C$: $7 > 10$ **or** la Terra è più grande di Urano **falso**

predicato $\neg B \wedge \neg C$: $7 \leq 10$ **and** la Terra **non** è più grande di Urano **vero**

convenzione per operazioni (funzioni) logiche:
tabella di verità

| p | $\neg p$ |
|----------|----------|
| v | f |
| f | v |

not

| p | q | $p \wedge q$ |
|----------|----------|--------------|
| v | v | v |
| v | f | f |
| f | v | f |
| f | f | f |

and

| p | q | $p \vee q$ |
|----------|----------|------------|
| v | v | v |
| v | f | v |
| f | v | v |
| f | f | f |

or

convenzione per operazioni (funzioni) logiche:
tabella di verità

| <i>p</i> | <i>q</i> | |
|----------|----------|----------|
| v | v | f |
| v | f | f |
| f | v | f |
| f | f | v |

nor

| <i>p</i> | <i>q</i> | |
|----------|----------|----------|
| v | v | f |
| v | f | v |
| f | v | v |
| f | f | v |

nand

| <i>p</i> | <i>q</i> | |
|----------|----------|----------|
| v | v | f |
| v | f | v |
| f | v | v |
| f | f | f |

xor

Esempi:

relazione di De Morgan:

$$p \wedge q = \neg(\neg p \vee \neg q)$$

i predicati

$$(x \geq a) \wedge (x \leq b)$$

$$\neg((x < a) \vee (x > b))$$

sono equivalenti

sono **veri** se e solo se x appartiene
all'intervallo $[a, b]$