Istruzione if/else

Espressioni booleane

Espressioni booleane

- Possono assumere solo due valori:
  - VERO (TRUE, per il C qualunque valore diverso da 0)
  - FALSO (FALSE, il valore 0)

Operatori relazionali

Permettono di confrontare fra di loro due valori:

== uguale
!= diverso
> maggiore
< minore
>= maggiore o uguale
<= minore o uguale

Se il confronto è vero il risultato è 1 altrimenti 0

Operatori booleani

- **&& (and logico), || (or logico), ! (not logico)**
  - **! falso → vero, ! vero → falso**
  - **Attenzione non è necessariamente vero che !!x sia uguale a x**
    - Ad esempio !!2 → !(!2) .→ !0 → 1 (però sia 1 che 2 sono 'veri')
    - Ovviamente !!0 → 0

AND	FALSE	TRUE
FALSE	FALSE	FALSE
TRUE	FALSE	TRUE

OR	FALSE	TRUE
FALSE	FALSE	TRUE
TRUE	TRUE	TRUE

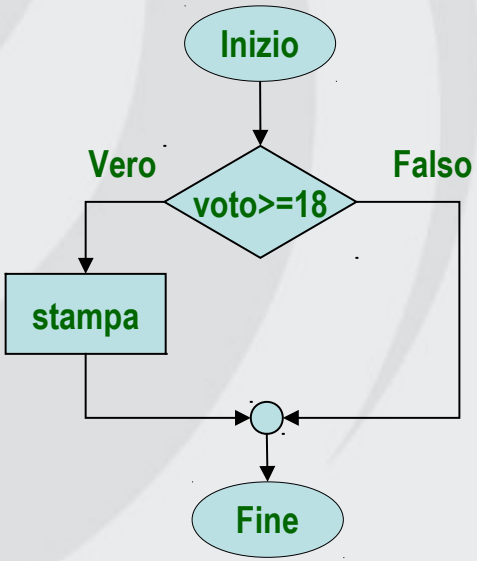
- **La valutazione delle espressioni booleane termina non appena il risultato è certo:**
  - **`espr1 && espr2` se espr1 risulta falsa espr2 non viene valutata**
  - **`espr1 || espr2` se espr1 risulta vera espr2 non viene valutata**
  - **3<2 && var++ 3<2 è falsa quindi a non viene incrementata**

Operatori booleani

- Attenzione a non confondere && e || con & e |
  - 4 || 2 → 1
  - 4 | 2 → 6 (comunque interpretato vero)
  - 4 && 2 → 1
  - 4 & 2 → 0 (falso)
  - (3<2) & var++ → 0 (falso) ma var viene comunque incrementato

Selezione binaria: istruzione if

- Sintassi:
  - `if(espressione_booleana) istruzione;`
- È usata per scegliere fra due alternative:
  - Se il voto è maggiore o uguale a 18 allora hai superato l'esame
- In C:
  - if(voto>=18) printf("Esame superato\n");



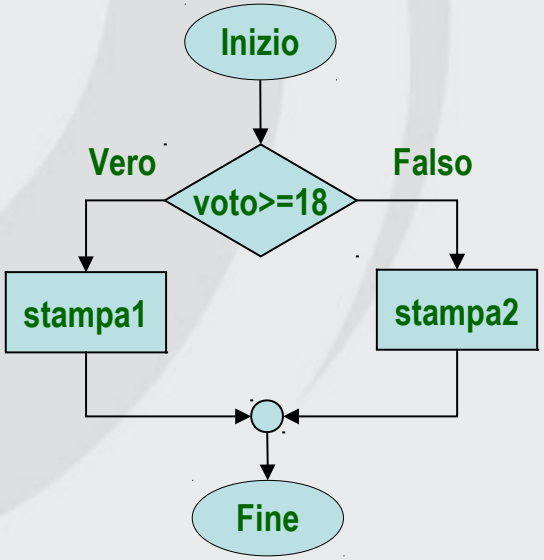
L'istruzione if è il primo esempio di istruzione (o costrutto) di controllo (del flusso)

Selezione binaria: istruzione if/else

- **Sintassi generale:**

```
if(espressione_booleana) istruzione1;
else istruzione2;
```

```
if(voto>=18)
   printf("Esame superato\n");
else
   printf("Esame non superato\n");
```



Attenzione ad un errore frequente:

`if(a=0) ...;`

L'istruzione è sintatticamente corretta e dà sempre come risultato falso (0)

Normalmente in realtà si voleva scrivere

`if(a==0) ...;`

Selezione binaria: istruzione if/else

- Attenzione ad un errore frequente:

```
if(voto=30)
    printf("Complimenti!\n");
```

- L'istruzione è sintatticamente corretta e dà sempre come risultato vero (30)
  - Normalmente in realtà si voleva scrivere

```
if(voto==30)
```

- Comunque con l'opzione di compilazione -Wall viene segnalata una anomalia

Operatore ?

- L'operatore ? è l'unico operatore che coinvolge 3 espressioni
- Sintassi:
  <espressione booleana> "?" <espressione 1> ":" <espressione 2>
- Viene valutata l'espressione booleana, se risulta vera il valore finale è il risultato della prima espressione, altrimenti della seconda

```
min = a<b ? a : b; // min assume il minimo fra a e b
```

Calcolo del massimo

```
int val1, val2, max;
…
if(val1>val2) {
  max = val1;
} else {
  max = val2;
}

max = val2;
if(val1 > max) max = val1;



max = val1 > val2 ? val1 : val2;
```

If annidati

```
char classe;
if ( voto >= 30 )
   classe='A';
else
   if ( voto >= 27 )
      classe='B';
   else
      if ( voto >= 24 )
         classe='C';
      else
         if ( voto >= 18 )
            classe='D';
         else
            classe='F';
```

la parte **else** viene eseguita quando la condizione **if** non è soddisfatta

Se la condizione è soddisfatta, il resto dell'istruzione viene saltato

Espressioni booleane complesse

- Un modo alternativo di scrivere istruzioni annidate
  - Si evita una eccessiva indentazione
  - In genere si preferisce questa convenzione

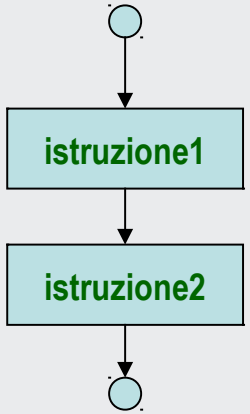
```
char classe;
if ( voto >= 30 )
   classe='A';
else if ( voto >= 27 )
   classe='B';
else if ( voto >= 24 )
   classe='C';
else if ( voto >= 18 )
   classe='D';
else
   classe='F';
```

Dal punto di vista dell'esecuzione non cambia nulla

Istruzione composta

- Molto spesso una unica istruzione non è sufficiente
- Se necessario allora si usa la “istruzione composta”
  - Non è altro che una serie di istruzioni racchiuse tra parentesi graffe
  - Viene interpretata dal compilatore come una singola istruzione
  - Attenzione non deve essere terminata dal “;” come le istruzioni semplici

```
{
    istruzione1;
    istruzione2;
    …
}
```



Calcolo del massimo e del minimo

```
int a, b, max, min;

scanf("%d %d", &a, &b);

if(a>b) {
     max = a;
     min = b;
} else {
     max = b;
     min = a;
}
```

- È usata anche la notazione alternativa:

```
if(a>b)
{
     max = a;
     min = b;
}
else
{
     max = b;
     min = a;
}
```

Calcolo del massimo e del minimo

- Consiglio in fase di editing:
  - Se si usano delle parentesi (qualunque tipo di parentesi) scrivere immediatamente la coppia di parentesi e poi inserire all'interno il testo necessario
  - Diminuisce in maniera significativa la probabilità di dimenticare una parentesi
- Notare che trovare eventuali errori è molto difficile se il codice precedente è scritto senza utilizzare la "indentazione"

```
if(a>b){max=a;min=b;}else{max=b;min=a;}
```

Calcolo della radice quadrata

sqrt.c (~) - gedit

Open Save Undo

sqrt.c

```c
#include <stdio.h> // contiene la definizione di printf e scanf
#include <math.h>  // contiene la definizione di sqrt

int main(int argc, char **argv)
{
    int n;
    printf("Il programma calcola la radice quadrata di un numero intero\n");
    printf("Scrivi un numero: ");
    if( scanf("%d", &n) == 1) {
        double quadrato = n;
        /*
         * sqrt richiede un parametro double e
         * e restituisce un valore double
         */
        double radice = sqrt(quadrato);
        // %f permette la stampa di un varore double
        printf("La radice quadrata di %d e` %f\n", n, radice);
        return 0;
    } else {
        printf("Il dato introdotto non e` corretto\n");
        return 1;
    }
}
```

C Tab Width: 4 Ln 1, Col 64 INS

Compilazione di sqrt.c

```
user@ubuntu: ~
$ gcc -Wall sqrt.c
/tmp/cc7KPLkx.o: In function `main':
sqrt.c:(.text+0x68): undefined reference to `sqrt'
collect2: ld returned 1 exit status
$
$
$ gcc -Wall sqrt.c -lm
$
```

Non trova il codice di sqrt

Viene aggiunta la libreria “m” (math)

if annidati (massimo di 3 valori)

```
int a, b, c, max;
printf("Scrivi 3 valori: ");
scanf("%d %d %d");
if(a>b) {  // il massimo è a o c
    if(a>c) {
            max = a;
    } else {
            max = c;
    }
} else {  // il massimo è b o c
    if(b>c) {
            max = b;
    } else {
            max = c;
    }
}
```

Come esercizio calcolare contemporaneamente anche il minimo
Il codice è corretto anche se vi sono valori uguali?

Altri esempi

- Equazione di secondo grado
- Anno bisestile