Fondamenti di Informatica e Laboratorio T-AB
Ingegneria Elettronica e Telecomunicazioni

Lab 04

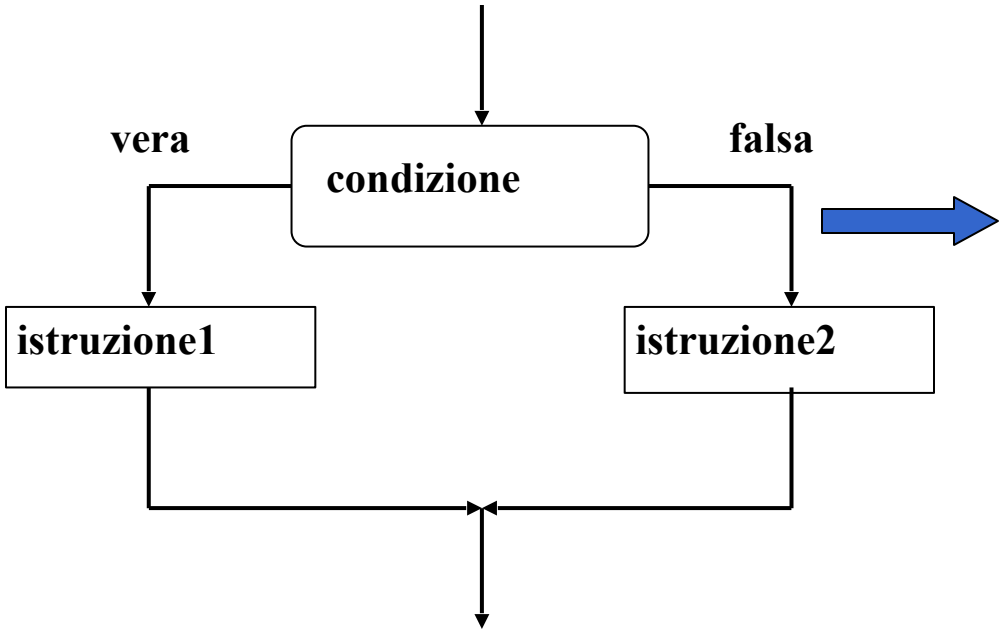
Istruzioni
Funzioni semplici

Lab 04

Istruzioni

ISTRUZIONE DI SCELTA SEMPLICE

```
<scelta> ::=  if (<cond>) <istruzione1>
                [ else <istruzione2> ]
```



La parte **`else`** è *opzionale*: se omessa, in caso di condizione falsa si passa subito all'istruzione che segue l'**`if`**.

ISTRUZIONI IF ANNIDATE

- Occorre attenzione *ad associare le parti else all' if corretto*

Regola semantica:
l'**else** è sempre associato all'**if** più interno

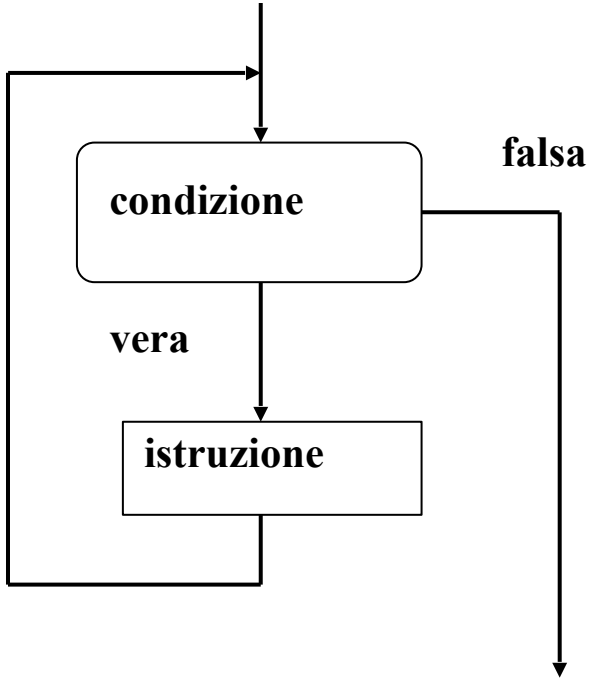
```
if (n > 0)
   if (a>b) n = a;
   else n = b; /* riferito a if(a>b) */
```

Se vogliamo cambiare questa semantica, dobbiamo inserire un blocco

```
if (n > 0)
   { if (a>b) n = a; }
else n = b; /* riferito a if(n>0) */
```

ISTRUZIONE `while`

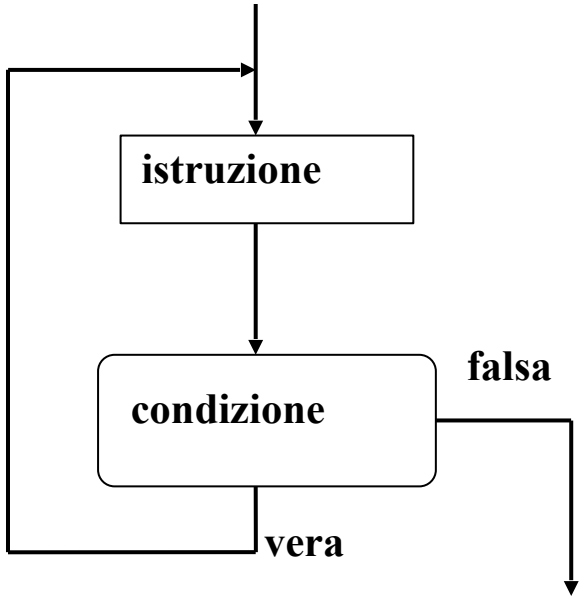
```
<while> ::=
    while(<condizione>) <istruzione>
```



- L'istruzione viene ripetuta *per tutto il tempo in cui la condizione rimane vera.*
- Se la condizione è **falsa**, l'iterazione **non viene eseguita *neppure una volta***.

ISTRUZIONE `do..while`

```
<do-while> ::=
  do <istruzione>  while(<condizione>);
```



È una variante della precedente: la condizione viene verificata *dopo* aver eseguito l'istruzione.

Se la condizione è falsa, l'iterazione **viene comunque eseguita *almeno una volta***.

ISTRUZIONE `for`

- È una evoluzione dell'istruzione **`while`** che mira a eliminare alcune frequenti sorgenti di errore:
  - mancanza delle *inizializzazioni delle variabili*
  - mancanza della *fase di modifica del ciclo* (rischio di ciclo senza fine)

```
<for> ::=
 for( <espr-iniz>;<cond>;<espr-modifica>) <istruzione>
```

Esercizio 1 – Cicli

Si realizzi un programma che, partendo da una base **`a`** ed un limite **`n`** **(inseriti da tastiera)**, calcoli la seguente funzione:

$$\sum_{i=0}^{n} a^i$$

Provate qualche test con questi valori di input:
a = 2 , n = 2 somma = 7
a = 4 , n = 5 somma = 1365
a = 5 , n = 3 somma = 156
a = 3 , n = 8 somma = 9841
a = 12 , n = 2 somma = 157

Realizzare il programma in due modi diversi:
1. Utilizzando due cicli (uno per la sommatoria, ed uno per la potenza)
2. Utilizzando un ciclo solo (è possibile usare una condizione)

La soluzione sarà mostrata con cicli for e una variante con while

Esercizio 2 – if innestati

Stampa di caratteri in ordine alfabetico

- Realizzare un programma che legge da input tre caratteri, e li stampa in ordine alfabetico. Si assuma di inserire 3 caratteri diversi.
- Utilizzando l'istruzione if
  - Per determinare il secondo carattere, devo per forza utilizzare degli if innestati

Lab 04

Funzioni Semplici

ESEMPIO

```
int max (int x, int y ){
    if (x>y) return x;
          else return y;
}
```

- Il simbolo **max** denota il nome della funzione
- Le variabili intere **x** e **y** sono i parametri della funzione
- Il valore restituito è un intero **int**

```
int main(){
      int m;
      m = max(2,13);
      return 0;
}
```

Esercizio 1 - Funzioni

Codificare in C la funzione

```
int max(int a, int b)
```

che restituisce il massimo valore tra due interi.

Codificare in C la funzione

```
int max3(int a, int b, int c)
```

che restituisce il massimo valore fra tre interi, sfruttando la funzione max definita precedentemente.

Definire un possibile main che prende in ingresso i tre valori dall'utente (input da tastiera) e ne stampa il massimo.

Esercizio 2 - Funzioni

Si progettino e si realizzino due funzioni così definite:

```
float euro_to_dollari(float money)
float euro_to_lire(float money)
```

ognuna delle quali converte un valore in euro nella moneta corrispondente. A tal fine si supponga che:
1 € = 1.07 $
1 € = 1936.27 £

Si progetti poi un programma che legge da input un valore, inteso come quantità di euro, e stampa la conversione in dollari ed in lire.

Esercizio 3 - Funzioni

Si scriva una funzione

```
int somma_potenze(int a,int n);
```

che dati **a** e **n** deve calcolare $\sum_{i=1}^{n} a^i$

A tal fine si scriva una funzione

```
int potenza(int x,int y);
```

che dati **x** e **y** deve calcolare x^y usando come operazione primitiva il prodotto.

Esercizio 4 - Funzioni

Creare una funzione **`float square(float x)`** . La funzione deve restituire il quadrato del parametro **`x`**.

Creare un'altra funzione, di nome **`float cube(float x)`**, che restituisce invece il cubo del valore **`x`**.

Progettare quindi e codificare un programma che legge un float da tastiera e restituisce il suo quadrato ed il suo cubo. Per calcolare il quadrato ed il cubo si devono utilizzare le due funzioni sopra definite.

Esercizio 5 - Funzioni

Si scriva una funzione

```
int somma2(int n);
```

che dato `n` deve calcolare $\sum_{i=1}^{n} \sum_{j=1}^{i} j$

A tal fine si sfrutti una funzione

```
int somma(int n);
```

che dato `n` deve calcolare $\sum_{k=1}^{n} k$

Esercizio 6 - Funzioni

Codificare in C la funzione
`int min_to_sec(int a)` che considera il parametro a come minuti e restituisce il numero di secondi corrispondente.

Codificare in C la funzione
`int ore_to_sec(int a)` che considera il parametro a come ammontare di ore, e restituisca il numero di secondi corrispondente. Si utilizzi la funzione definita precedentemente.

Definire un possibile main che prende in ingresso tre valori interi, rappresentanti ore, minuti e secondi della durata di un CD Audio. Il programma deve stampare il valore corrispondente in secondi.

Esercizio 7 - Funzioni

Codificare in C la funzione
`int ipotenusa(int a, int b)` che, dati i cateti **`a`** e **`b`** di un triangolo rettangolo, restituisce il valore dell'ipotenusa.

A tal scopo si utilizzi il Teorema di Pitagora:

$$Ipotenusa = \sqrt{a^2 + b^2}$$

Per calcolare la radice quadrata si utilizzi la funzione di libreria **`sqrt(x)`**. Per utilizzare quest'ultima si aggiunga l'istruzione
`#include <math.h>` in testa al file.

Definire un possibile main che legga da tastiera due valori che rappresentino i cateti di un triangolo rettangolo, e stampi il valore dell'ipotenusa.

Esercizio 8 - Funzioni

Codificare in C la funzione
`int perimetro(int a, int b, int c)` che, dati i lati a,b,c di un triangolo, ne calcola il perimetro.

Codificare in C la funzione
`float area(int a, int b, int c)`
che restituisce l'area di un triangolo i cui lati misurano a, b, c. A tal scopo si usi la formula di Erone:

$$Area = \sqrt{p(p-a)(p-b)(p-c)}$$

Dove p è la metà del perimetro. A tal scopo si includa l'header **`<math.h>`** e si utilizzi la funzione **`sqrt(x)`** .

Definire un possibile main che prende in ingresso i tre lati di un triangolo e stampa perimetro ed area.

Esercizio 9 - Funzioni

Codificare in C la funzione **`int primo(int x)`** che restituisce:
1 se x è un numero primo
0 altrimenti.
Si utilizzi a tal proposito l'operatore modulo (%).

Si progetti un programma che legge da tastiera un numero N, e stampa a video tutti i numeri primi compresi tra 0 e N.