

Fondamenti di Informatica e Laboratorio T-AB e Fondamenti di Informatica T1  
Ingegneria Elettronica e Telecomunicazioni e  
Ingegneria dell'Automazione  
a.a. 2010/2011

---

# Lab 07

## Funzioni semplici

# Esercizio 1 - Funzioni

---

Codificare in C la funzione

```
int max(int x, int y)
```

che restituisce il massimo valore tra due interi.

Codificare in C la funzione

```
int max3(int x, int y, int z)
```

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 e ne stampa il massimo.

# Esercizio 1 – Funzioni - Soluzione

---

```
int max(int a, int b)
{ if (a>b) return a;
  else return b;
}
```

```
int max3(int a, int b, int c)
{ int max_di_due;
  max_di_due = max(a,b);
  return max(max_di_due,c);
}
```

# Esercizio 1 – Funzioni - Varianti

---

```
int max3(int a, int b, int c)
{
    if max(a,b) > c)
        return max(a,b) ;
    else
        return c;
}
```

```
int max3(int a, int b, int c)
{
    return max(max(a,b) ,c) ;
}
```

# Esercizio 1 – Funzioni - Soluzione

---

Un possibile main

```
int main()
{int MAX;
  int v1, v2, v3;
  printf("Inserisci tre interi");
  scanf("%d, %d, %d", &v1,&v2,&v3);
  MAX = max3(v1,v2,v3);
  printf("Massimo valore inserito: %d",MAX);
  return 0;
}
```

NOTA: Prima di chiamare una funzione è necessario definirla. Nel file sorgente quindi prima del **main** e' necessario definire la funzione **max3** e prima di **max3** bisogna definire **max**

# Esercizio 2 - 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 2 – Funzioni - Soluzione

---

```
int somma(int n)
{ int k, s=0;
  for (k=1; k<=n; k++)
    s = s + k;
  return s;
}
```

```
int somma2(int n)
{int i, s2 = 0;
  for (i=1; i<=n; i++)
    s2 = s2 + somma(i);
  return s2;
}
```

NOTA: Nel file sorgente prima del **main** e' necessario definire la funzione **somma2** e prima di **somma2** bisogna definire **somma**

# Esercizio 2 – Funzioni - Soluzione

---

Un possibile main

```
int main() {  
    int N, S;  
  
    printf("Inserisci un intero");  
    scanf("%d", &N);  
  
    S = somma2(N);  
    printf("La somma vale %d", S);  
  
    return 0;  
}
```



# 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<sup>y</sup>** usando come operazione primitiva il prodotto.

# Esercizio 3 – Funzioni - Soluzione

---

```
int potenza(int x,int y)
{ int i, P=1; /* P: accumulatore di prod.*/
  for(i=1; i<=y; i++)
    P = P * x;
  return P;
}
```

```
int somma_potenze(int a, int n)
{ int i, s=0;
  for(i=1; i<=n; i++)
    s = s + potenza(a,i);
  return s;
}
```

# Esercizio 3 – Funzioni - Soluzione

---

Un possibile main

```
int main() {  
    int N1,N2,SP;  
  
    printf("Inserisci due interi");  
    scanf("%d,%d", &N1,&N2);  
  
    SP = somma_potenze(N1,N2);  
    printf("La somma delle potenze vale %d",SP);  
  
    return 0;  
}
```

# 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 4 – Funzioni - Soluzione

---

```
float square(float x)
{
    return x*x;
}
```

```
float cube(float x)
{
    return x*x*x;
}
```

# Esercizio 5 - 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 € = 0.98 \$

1 € = 1936.27 £

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

# Esercizio 5 – Funzioni - Soluzione

---

```
float euro_to_dollari(float money)
{
    return money*0.98;
}
```

```
float euro_to_lire(float money)
{
    return money*1936.27;
}
```

# 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 6 – Funzioni - Soluzione

---

```
int min_to_sec(int a)
{
    return a*60;
}
```

```
int ore_to_sec(int a)
{
    return a*60*60;
}
```

# 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 7 – Funzioni - Soluzione

---

```
float ipotenusa(float a, float b)
{
    return sqrt(a*a + b*b);
}
```

# 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 8 – Funzioni - Soluzione

---

```
float perimetro(float a, float b, float c)
{
    return a + b + c;
}
```

```
float area(float a, float b, float c)
{
    float p;
    float area;

    p = perimetro(a, b, c) / 2;
    area = sqrt(    p * (p-a) * (p-b) * (p-c)    );
    return 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  $N$ .

# Esercizio 9 – Funzioni - Soluzione

---

```
int primo(int x) {
    int i, resto;

    if ((x == 1) || (x == 2))
        return 1;
    else {
        i= 2;
        do
        {
            resto = x % i;
            i++;
        }
        while ((resto != 0) && (i<x));

        return (resto != 0);
    }
}
```