

Titolo modulo : Function C per problemi applicativi di base
[07-C]

Sviluppo di function in C per il calcolo del massimo, del minimo e per la ricerca, su insiemi di dati forniti da tastiera

Argomenti trattati:

- ✓ programmi C per il massimo, il minimo e il massimo/minimo
- ✓ programma C per la ricerca sequenziale

Prerequisiti richiesti: AP-05-03-C, AP-06-03-T, AP-06-04-T

problema:

calcolo dell'elemento massimo di un insieme di dati

```
#include <stdio.h>
void main()
/* massimo di un insieme di dati
   immessi da tastiera */
{
    int n,i;
    float numero_letto, massimo;
    printf("inserire il numero dei numeri (>1): ");
    scanf("%d",&n);
    printf("inserire il primo numero: ");
    scanf("%d",&numero_letto);
    massimo = numero_letto;
    for (i = 2; i <= n; i++)
    {
        printf("inserire %d-simo numero: ",i);
        scanf("%f",&numero_letto);
        if(numero_letto > massimo)
            massimo = numero_letto ;
    }
    printf("massimo dell'insieme di %d termini:%f\n",n,massimo);
}
```

```
main {
float numero_letto, massimo;
int n, i;
read (n,numero_letto) ;
massimo = numero_letto;
for (i=2, i<=n , i++) {
    read (numero_letto) ;
    if (numero_letto > massimo) {
        massimo = numero_letto ;
    }
}
printf (massimo) ;
}
```

problema:

calcolo dell'elemento minimo di un insieme di dati **immessi da tastiera**

```
#include <stdio.h>
void main()
/* minimo di un insieme di dati
   immessi da tastiera */
{
    int n,i;
    float numero_letto, minimo;
    printf("inserire il numero dei numeri (>1): ");
    scanf("%d",&n);
    printf("inserire il primo numero: ");
    scanf("%d",&numero_letto);
    minimo = numero_letto;
    for (i = 2; i <= n; i++)
    {
        printf("inserire %d-simo numero: ",i);
        scanf("%f",&numero_letto);
        if(numero_letto < minimo)
            minimo = numero_letto ;
    }
    printf("minimo dell'insieme di %d termini:%f\n",n,minimo);
}
```

problema:

calcolo dell'elemento massimo e dell'elemento minimo
immessi da tastiera

```
#include <stdio.h>
void main()
/* massimo e minimo di un insieme di
dati immessi da tastiera */
{
    int n,i;
    float numero_letto, max, min;
    printf("inserire il numero dei numeri (>1): ");
    scanf("%d",&n);
    printf("inserire il primo numero: ");
    scanf("%d",&numero_letto);
    max = numero_letto ; min = numero_letto ;
    for (i = 2; i <= n; i++)
    {
        printf("inserire %d-simo numero: ",i);
        scanf("%f",&numero_letto);
        if(numero_letto > max)
            max = numero_letto ;
        else if(numero_letto < min)
            min = numero_letto ;
    }
    printf("max e min dell'insieme di %d termini:%f %f\n",n,max,min);
}
```

```
main {
float numero_letto, max, min;
int n, i;
    read (n,numero_letto) ;
    max = numero_letto ;
    min = numero_letto ;
    for (i=2, i<=n, i++) {
        read (numero_letto);
        if (numero_letto > max) {
            max = numero_letto
        }
        else if (numero_letto < min) {
            min = numero_letto ;
        }
    }
    printf (max, min) ;
}
```

problema:

ricerca di una chiave in un insieme di dati **immessi da tastiera**

```
#include <stdio.h>
void main()
/* ricerca sequenziale di una chiave
in un insieme di
dati immessi da tastiera */
{
    int n,i=0,esito_ricerca=0;
    char chiave, dato_letto;
    printf("inserire il numero dei caratteri (>1): ");
    scanf("%d",&n);
    printf("inserire la chiave (1 car): ");
    fflush(stdin);
    scanf("%c",&chiave);
    do {
        printf("inserire %d-simo carattere:",i);
        fflush(stdin);
        scanf("%c",&dato_letto);
        i++;
        if(dato_letto == chiave)
            esito_ricerca = 1 ;
    }
    while (! (esito_ricerca || i==n));
    printf("\nesito ricerca della chiave %c nell'insieme di %d termini:%d \n",
        chiave,n,esito_ricerca);
}
```

```
main {
    char chiave,dato_letto;
    int n, i ;
    logical esito_ricerca ;
    read (chiave,n) ;
    i = 0 ;
    esito_ricerca = false ;
    do {
        read (dato_letto) ;
        i = i+1 ;
        if (chiave == dato_letto) {
            esito_ricerca = true ;
        }
    }
    while (! (esito_ricerca || i == n) ) ;
    print (esito_ricerca) ;
}
```

esercizi

realizzare le seguenti function C

```
int maxI(int a, int b)
float maxF(float a, float b)
char maxC(char a, char b)
int minI(int a, int b)
float minF(float a, float b)
char minC(char a, char b)
```

che determinano il max o il min tra due dati

```
int e_pari(int n)
int e_dispari(int n)
```

che determinano se un intero è pari o è dispari
(consiglio: usare `n%2`)

```
#include <stdio.h>
int maxI(int a, int b);
void main()
{
    int x,y;
    printf("inserire i valori dei 2 numeri (int): ");
    scanf("%d %d",&x,&y);
    printf("il piu' grande tra %d e %d : %d\n",x,y,maxI(x,y));
}
/* massimo tra due numeri interi */
int maxI(int a, int b)
{
    if (a > b)
        return a;
    else
        return b;
}
```

```
/* determina se un numero intero e' pari */
int epari(int n)
{
    if (n%2 == 0)
        return 1;
    else
        return 0;
}
```

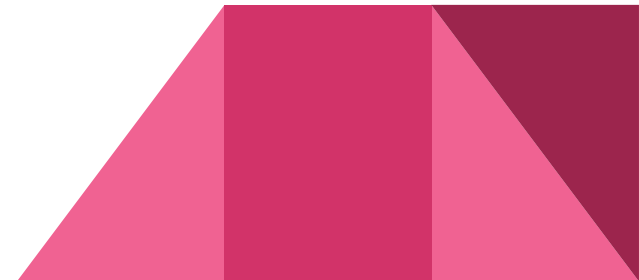
```
/* determina se un numero intero e' pari */
int epari(int n)
{
    return !(n%2);
}

/* determina se un numero intero e' dispari */
int edispari(int n)
{
    return n%2;
}
```


Esercizi

Esercizio 1

Eleva al cubo una variabile n (data in input) usando una chiamata per valore.



```
1 #include <stdio.h>
2
3 int cubopervalore(int x);
4
5 int main(void){
6     int num, cubo;
7     printf("Inserisci valore\n");
8     scanf("%d", &num);
9
10    cubo = cubopervalore(num);
11
12    printf("valore al cubo = %d\n", cubo);
13
14    return 0;
15 }
16
17 int cubopervalore(int x){
18     return x * x * x;
19 }
```

chiamata per valore.

Inserisci valore

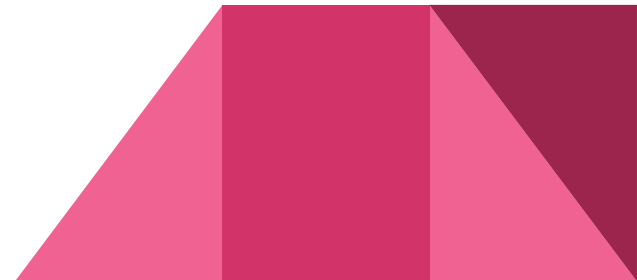
3

valore al cubo = 27

Program ended with exit code: 0

Esercizio 2

Eleva al cubo una variabile n (data in input) usando una chiamata per riferimento.



```
1 #include <stdio.h>
2
3 void cuboperindirizzo(int *);
4
5 int main(void){
6     int num;
7     printf("Inserisci valore\n");
8     scanf("%d", &num);
9
10    cuboperindirizzo(&num);
11
12    printf("valore al cubo = %d\n", num);
13
14    return 0;
15 }
16
17 void cuboperindirizzo(int *x){
18     *x = *x * *x * *x;
19 }
```

chiamata per riferimento.

Inserisci valore

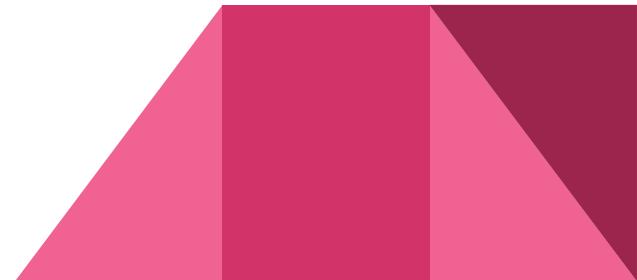
3

valore al cubo = 27

Program ended with exit code: 0

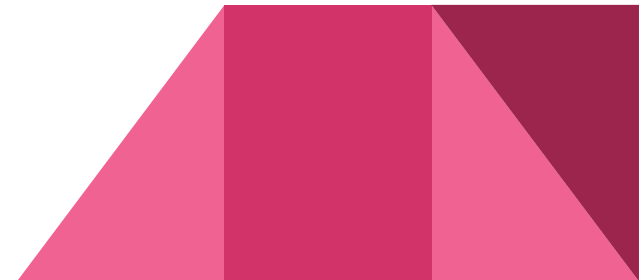
Esercizio 3

Scrivere un programma che data una sequenza di numeri inseriti dall'utente visualizzi quanti di loro sono negativi e quanti sono positivi.



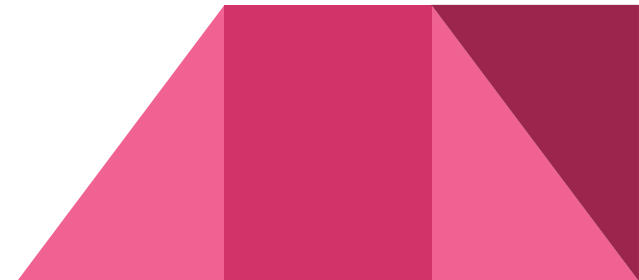
Esercizio 4

Scrivere un programma che, data una sequenza di caratteri, visualizzi quanti corrispondono a delle vocali.



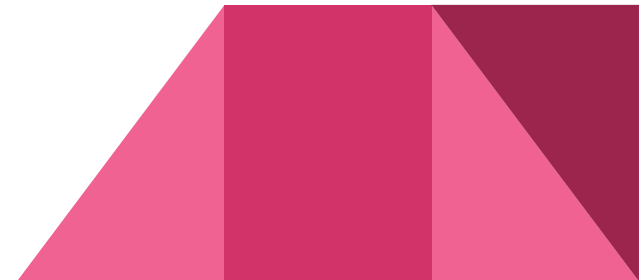
Esercizio 5

Scrivere un programma in cui l'utente deve indovinare, in due fasi, con un massimo di 10 tentativi, un codice segreto. Il codice segreto è un intero definito dal programmatore con valore tra 0 e 100. Se l'utente indovina il codice nella prima fase, allora nella seconda fase l'utente deve indovinare un secondo codice segreto composto da una vocale (controllare che l'utente non inserisca una consonante).



Esercizio 6

Scrivere un programma per simulare l'inserimento di un PIN per il telefonino. Nella prima fase viene chiesto all'utente di inserire un codice di lunghezza 5. Nella seconda fase l'utente inserisce il codice e ha al massimo 3 tentativi per indovinarlo.



Esercizi su funzioni e procedure

Esercizio 1 dal manuale pag 36

Dato il seguente programma costruire due funzioni: una che restituisca la distanza tra due punti e l'altra per la traslazione di un punto. Suddividere il programma su più file: file per il main, il file header ed un file con le funzioni.

```
#include <stdio.h>

int main () {
    float x1,y1,x2,y2,d,t1,t2,a=2,b=10;
    printf ("Inserire le coordinate del primo punto (float) : ");
    scanf("%f%f",&x1,&y1);
    printf ("Inserire le coordinate del secondo punto (float) : ");
    scanf("%f%f",&x2,&y2);
    t1 = x1 + a;
    t1 = y1 + b;
    d = sqrt(x1*x2+y1*y2);
    printf ("la distanza dei punti è:%f\n",d);
    return 0;
}
```

Riscrivere il seguente programma utilizzando una procedura

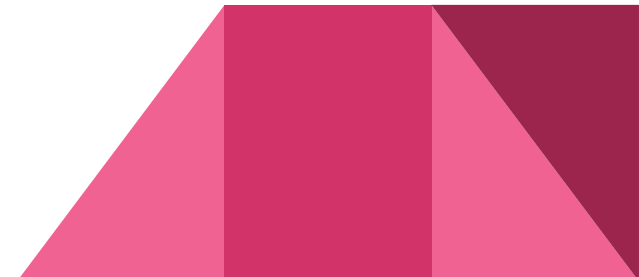
```
#include <stdio.h>

int f(int);

int main () {
    int i;
    int a, s = 0;
    for(i=0;i<3;i++)
    {
        printf("Dammi il %d numero\n", i+1);
        scanf("%d", &a);
        s += f(a);
    }
    printf("Il risultato e': %d", s);
    return 0;
}

int f(int a)
{
    static int cnt = 1;
    ++cnt;
    if (cnt % 2 == 0) return a+cnt;
}
```

Esercizio 4 dal manuale pag 38



Esercizio 7 dal manuale pag 39

Scrivere una funzione che calcoli la seguente formula

$$(x^3 + 3x + 5)/(8x + 1) \quad (3.1)$$

Scrivere successivamente un programma che calcoli i valori della funzione per $x = 0, 2, 4, 6, \dots, 20$.



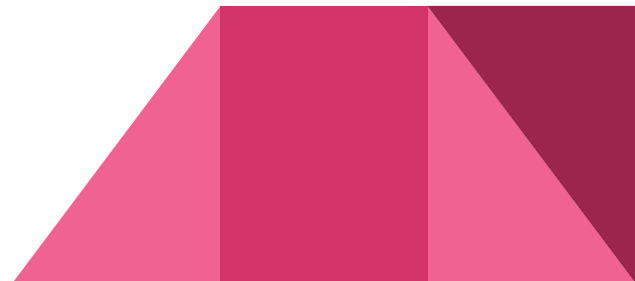
Esercizio 8 dal manuale pag 40

Scrivere una procedura che dati due valori x e y calcoli i valori delle seguenti funzioni matematiche

$$(x^3 + 3x + 5)/(8x + 1) \quad (3.2)$$

$$(y^4)/(1 + y) \quad (3.3)$$

Scrivere successivamente un programma che calcoli e visualizzi il risultato delle funzioni precedenti nei punti $(0, 0)$, $(1, 1)$, $(3, 3)$, $(5, 5)$, $\dots$, $(9, 9)$.

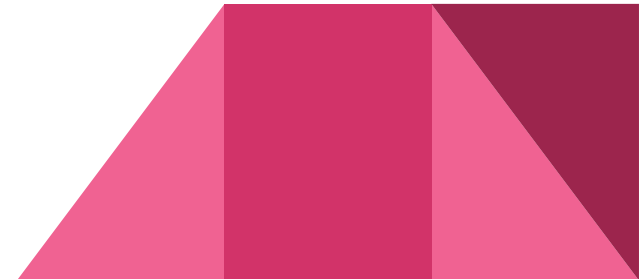


Esercizio 9 [✓]

Descrivere il significato della seguente procedura e trasformarla in funzione

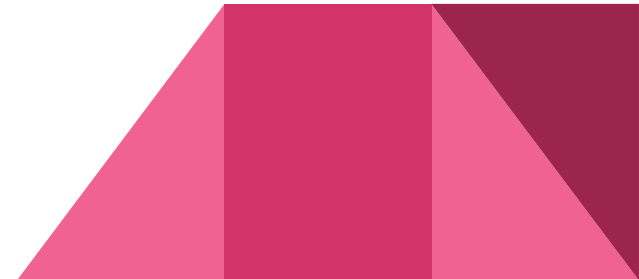
```
void f(int m, int n, int *p)
{
    int i; long p = 1;
    for(i = 1; i <= n; ++i) p *= m; }
```

Scrivere successivamente un programma che calcoli e visualizzi i suoi risultati in una tabella, usando i seguenti valori di **m** ed **n** : 0 – 0, 0 – 1, 0 – 2, 1 – 0, 1 – 1, 1 – 2, 2 – 0, 2 – 1, 2 – 2.



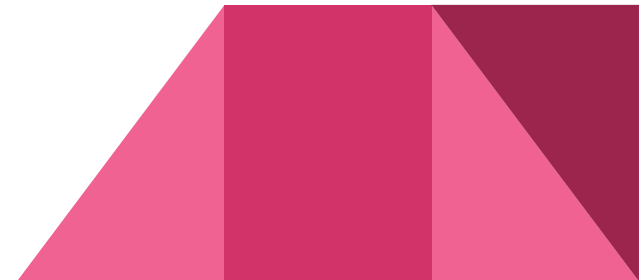
Esercizio 10 [✓]

Scrivere un programma che simuli un *salvadanaio*. L'utente può inserire e prelevare soldi. Visualizzare *salvadanaio vuoto* se non ci sono soldi. Il numero di operazioni di inserimento e prelievo sono decise dall'utente. L'inserimento e il prelievo devono avvenire con la chiamata a due procedure differenti.



Esercizio 11

Scrivere due funzioni che calcolino il massimo e il minimo di un insieme di valori inseriti dall'utente. Scrivere successivamente un programma che verifichi se la loro differenza è minore di un certo valore inserito.





I numeri casuali



Per la generazione di numeri pseudocasuali in C viene usata la funzione **rand()**. Essa genera solo numeri interi (tipo `int`) che appartengono all'intervallo `[0, RAND_MAX]` e che sono uniformemente distribuiti in tale intervallo. **RAND_MAX** è definito nella libreria **stdlib.h**. Un semplice utilizzo è il seguente

```
int nc;  
nc = rand();
```

Per generare a ogni esecuzione del programma una diversa sequenza di numeri pseudocasuali è necessario usare la funzione **srand()**

```
srand(nuovo_seed);
```



Ad esempio per una generazione automatica di un diverso seme si può usare la seguente chiamata a **srand**

```
srand((unsigned int)time(0));
```

dove **time** è una funzione della libreria **time.h**, che restituisce il valore dell'ora riportato dall'orologio interno del calcolatore.

Per avere dei numeri interi casuali uniformemente distribuiti in un qualsiasi intervallo $[A, B]$ viene usata l'operazione di modulo

```
int nc;
```

```
nc = A + rand()%(B + 1 - A);
```

Per generare numeri pseudocasuali di tipo reale uniformemente distribuiti nell'intervallo $[a, b]$ usiamo invece la seguente relazione

```
nc_f = a+(b-a) * (float)rand()/(float)RAND_MAX;
```

Esercizi su rand()

Esercizio 12

Simulare per dieci volte il lancio di un dado e visualizzare la somma dei risultati di tutti i lanci.

Esercizio 13 [✓]

Due giocatori si sfidano lanciando un “dado truccato”. Il dado ha dei valori interi nell’intervallo $[5, 15]$. A ogni turno vince il giocatore che ottiene un punteggio maggiore. In caso di parità il punto viene assegnato a entrambi. Simulare 10 sfide e visualizzare il giocatore che vince più volte.

Esercizio 14 [✓]

Scrivere una procedura che dato una ascissa x calcoli la seguente funzione matematica

$$(x^3 + 3x + 5)/(8x + 1) \quad (3.4)$$

Scrivere successivamente un programma che calcoli e visualizzi il risultato della funzione in 20 ascisse casuali scelte nell’intervallo $[0, 1]$.



Soluzione esercizio esercitazione

```
13 int main(void){
14
15     int somma1, somma_gauss;
16     int n;
17     printf("inserisci il numero di cui fare la sommatoria \n");
18     scanf("%d", &n);
19
20     somma1 = sommatoria(n);
21     printf("sommatoria = %d \n", somma1);
22     somma_gauss = gauss(n);
23
24     printf("sommatoria gauss = %d \n", somma_gauss);
25
26     if(somma1 == somma_gauss)
27         printf("Ben fatto!\n");
28     else
29         printf("Hai sbagliato qualcosa\n");|
30     return 0;
31 }
32
33 int sommatoria(int n){
34     int k, somma1 = 0;
35     for(k = 0; k <= n; k++)
36         somma1 += k;
37     return somma1;
38 }
39
40 int gauss(int n){
41     int somma2;
42     somma2 = n*(n+1)/2;
43     return somma2;
44 }
```