

Introduzione al C

Corso di Fondamenti di Informatica
Ingegneria Clinica

Esercitazione 5

DIPARTIMENTO DI INFORMATICA
E SISTEMISTICA ANTONIO RUBERTI



SAPIENZA
UNIVERSITÀ DI ROMA

Raffaele Nicolussi

Esercizio 1

Scrivere un programma che, letti da stdin un intero positivo n e un numeri reale x , calcola lo sviluppo di Taylor di ordine n della funzione e^x , dato dalla seguente formula:

$$e^x = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \dots + \frac{x^n}{n!}$$

```
Calcolo dello sviluppo di Taylor di ordine n di
e(x)
```

```
Inserire x: 1.5
```

```
Inserire n: 6
```

```
4.477539
```

```
Premere un tasto per continuare . . .
```



Soluzione

```
float x,num=1,risultato=1,den=1; int i=1,n;
```

```
printf("Calcolo dello sviluppo di Taylor di ordine n di e(x)\n");
```

```
printf("Inserire x: ");
```

```
scanf("%f",&x);
```

```
printf("Inserire n: ");
```

```
scanf("%d",&n);
```

```
for (i = 1; i <= n; i++)
```

```
{
```


```
    num = num*x;
```

```
    den = den * i;
```

```
    risultato = risultato + (num/den);
```

```
}
```

```
printf("%f\n",risultato);
```



Esercizio 2

Leggere da standard input una sequenza di caratteri terminata da invio e determinare il numero di occorrenze delle vocali all'interno della sequenza

```
Inserire una sequenza di caratteri terminata da invio:  
Prova di esecuzione dell'esercizio numero 2  
Numero occorrenze lettera a: 1  
Numero occorrenze lettera e: 7  
Numero occorrenze lettera i: 4  
Numero occorrenze lettera o: 4  
Numero occorrenze lettera u: 2  
Premere un tasto per continuare . . .
```

Soluzione

```
#include <stdio.h>
```

```
#include <stdlib.h>
```

```
int main () {
```

```
    int cont_a=0, cont_e=0, cont_i=0, cont_o=0, cont_u=0;
```

```
    char carattere;
```

```
    printf("\nInserire una sequenza di caratteri terminata da invio: \n");
```

```
    scanf("%c", &carattere);
```

```
    while ( carattere != '\n' ) {
```

```
        switch (carattere ) {
```

```
            case 'a': cont_a = cont_a +1; break;
```

```
            case 'e': cont_e = cont_e +1; break;
```

```
            case 'i': cont_i = cont_i +1; break;
```

```
            case 'o': cont_o = cont_o +1; break;
```

```
            case 'u': cont_u = cont_u +1; break;
```

```
        }
```

```
        scanf("%c", &carattere);
```

```
    }
```

Soluzione

```
printf("Numero occorrenze lettera a: %d\n", cont_a);  
printf("Numero occorrenze lettera e: %d\n", cont_e);  
printf("Numero occorrenze lettera i: %d\n", cont_i);  
printf("Numero occorrenze lettera o: %d\n", cont_o);  
printf("Numero occorrenze lettera u: %d\n", cont_u);  
  
system("pause");  
return 0;  
}
```

Esercizio 2

Leggere da standard input un numero intero $N \geq 1$ e stampare la somma dei numeri da 1 a N .

```
Inserire n (>=1): 100
Risultato = 5050
Premere un tasto per continuare .
. .
```

Soluzione

```
int n, i = 1, somma = 0;
printf("\nInserire n(>=1): ");
scanf("%d", &n);

while ( n <= 0 ) {
    printf ("Inserire n (>=1): ");
    scanf("%d", &n);
}

while ( i <= n ) {
    somma = somma + i;
    i = i + 1;
}

printf("Risultato = %d\n", somma);
```


Esercizio 4

Leggere da input due numeri interi n ed m e stampare tutti i numeri pari compresi tra n ed m se n è minore di m , oppure compresi tra m ed n se m è minore di n

```
Inserire n ed m: 4 71
6  8  10  12  14  16  18  20  22  24  26  28  30
32  34  36  38  40  42  44  46  48  50  52  54  56
58  60  62  64  66  68  70
Premere un tasto per continuare . . .
```

Soluzione 4

```
int n, m, i;

printf("\nInserire n ed m: ");
scanf("%d%d", &n, &m);

if ( n < m ) {
    i = n + 1;
    while ( i < m ) {
        if ( i%2 == 0 )
            printf ("%d ", i);
        i = i + 1;
    }
}
else {
    i = m + 1;
    while ( i < n ) {
        if ( i%2 == 0 )
            printf ("%d ", i);
        i = i + 1;
    }
}
```

Esercizio 4 bis

Leggere da input due numeri interi n ed m e stampare tutti i numeri dispari compresi tra n ed m se n è minore di m , oppure compresi tra m ed n se m è minore di n

Soluzione 4 bis

```
int n, m, i, max;
```

```
printf("\nInserire n ed m: ");  
scanf("%d%d", &n, &m);
```

```
if ( n < m ) {  
    max = m;  
    i = n + 1;
```

```
}
```

```
else {  
    max = n;  
    i = m + 1;
```

```
}
```

```
while ( i < max ) {  
    if ( i%2 != 0 )  
        printf ("%d ", i);  
    i = i + 1;
```

```
}
```

Esercizio 5

Scrivere un programma che legge da standard input un carattere c e due numeri interi b ed h , e stampa un rettangolo di base b ed altezza h di caratteri c

```
Inserire il carattere da stampare: &
```

```
Inserire base e altezza: 7 3
```

```
&&&&&&&
```

```
&&&&&&&
```

```
&&&&&&&
```

```
Premere un tasto per continuare . . .
```

Soluzione 5

```
int b, h, i, j;
char car;

printf("\nInserire il carattere da stampare: ");
scanf("%c", &car);
printf("\nInserire base e altezza: ");
scanf("%d%d", &b, &h);


printf("\n");
i = 1;
while ( i <= h) {
    j = 1;
    while ( j <= b ) {
        printf ("%c", car);
        j = j + 1;
    }
    printf("\n");
    i = i + 1;
}
```

Esercizio 3

Scrivere un programma che, letta da stdin una sequenza di caratteri terminata da invio, per ciascun carattere della sequenza esegue la seguente azione:

- se il carattere è una lettera minuscola stampa su stdout la corrispondente lettera maiuscola
- se il carattere è una lettera maiuscola stampa su stdout la corrispondente lettera minuscola
- in tutti gli altri casi stampa uno spazio

Nella rappresentazione ASCII

- le lettere maiuscole hanno codici compresi tra 65 ('A') e 90 ('Z')
 - le lettere minuscole hanno codici compresi tra 97 ('a') e 122 ('z')
- 

Esercizio 3

Inserire una sequenza di caratteri
terminata da invio:

```
abc12.Ief2g
```

```
ABC iEF G
```


```
Premere un tasto per continuare . . .
```


Soluzione

```
char carattere;
```

```
printf("\nInserire una sequenza di caratteri terminata da invio: \n");  
scanf("%c", &carattere);
```

```
while ( carattere != '\n' ) {  
    if ( carattere >= 65 && carattere <= 90 ) /* lettera maiuscola */  
        printf ("%c", carattere + 32);  
    else if ( carattere >= 97 && carattere <= 122 ) /* lettera  
minuscola */  
        printf ("%c", carattere -32);  
    else  
        printf ("%c", ' ');  
    scanf("%c", &carattere);  
}
```



Esercizio 4

Scrivere un programma che legge da stdin un carattere **c** e due interi **b** ed **h** e stampa—utilizzando un "ciclo for"—un rettangolo di **base b** ed **altezza h** di caratteri **c**, come specificato in figura

Se **c = 'A'**, **b = 5**, **h = 3**, il risultato sarà

```
AAAAA
AAAAA
AAAAA
```

*Usare una prima scanf per il carattere **c** ed una seconda per i due interi **b** ed **h***



Soluzione

```
int base , altezza , i , j ;
```

```
char car ;
```

```
printf( "Inserire un carattere:\n" ) ;
```

```
scanf( "%c" , &car ) ;
```

```
printf( "Inserire base rettangolo (intero >= 1):\n" ) ;
```

```
scanf( "%d" , &base ) ;
```

```
printf( "Inserire altezza rettangolo (intero >= 1):\n" ) ;
```

```
scanf( "%d" , &altezza ) ;
```

```
for ( i = 1 ; i <= altezza ; i = i+1 )
```

```
{
```

```
    for ( j = 1 ; j <= base ; j = j+1 )
```

```
        printf( "%c" , car ) ;
```

```
    printf( "\n" ) ;
```

```
}
```



Esercizio 5

Scrivere un programma C che legga da stdin una sequenza di numeri positivi la cui lunghezza non è nota a priori, terminata da un numero negativo. Per ogni numero letto il programma deve stampare su stdout la media di tutti i numeri letti fino a quel momento.

```
Inserisci un numero positivo (o negativo per terminare): 2.2
Media attuale (1 numero/i): 2.200000
Inserisci un numero positivo (o negativo per terminare): 3.3
Media attuale (2 numero/i): 2.750000
Inserisci un numero positivo (o negativo per terminare): 5.5
Media attuale (3 numero/i): 3.666667
Inserisci un numero positivo (o negativo per terminare): 0
Media attuale (4 numero/i): 2.750000
Inserisci un numero positivo (o negativo per terminare): -1
Premere un tasto per continuare . . .
```

Soluzione

```
int i = 1;
```

```
float media, num, somma = 0;
```

```
printf ("Inserisci un numero positivo (o negativo per terminare): ");
```

```
scanf("%f",&num);
```

```
while (num >= 0)
```

```
{
```

```
    somma = somma + num;
```

```
    media = somma/i;
```

```
    printf ("Media attuale (%d numero/i): %f\n", i, media);
```

```
    printf ("Inserisci un numero positivo (o negativo per terminare): ");
```

```
    scanf("%f",&num);
```

```
    i++;
```

```
}
```



Esercizio 6

Leggere da stdin una sequenza di **0** e **1** terminata da **2** (acquisire i valori uno per volta) e stampare la lunghezza della più lunga sottosequenza di soli **0** presente nella sequenza letta

Esempio: per la sequenza

0 0 1 **0 0 0** 1 1 1 1 0 0 2

la risposta cercata è **3**



Algoritmo Esercizio

Variabili utilizzate (tipo intero):

bit: valore letto, **contatore**: numero di 0 accumulati

lmax: massima lunghezza sottosequenza di 0

- **contatore=0**; **lmax=0** (inizializzazione)
- leggi un numero (valore registrato in **bit**)
- finché **bit** è diverso da 2
 - se **bit** è pari a 0:
 - incrementa **contatore**
 - se **contatore > lmax**: **lmax=contatore**
 - altrimenti
 - contatore=0**
 - leggi un altro numero
- stampa **lmax**

Soluzione

```
int bit, cont = 0, maxlung = 0;
```

```
printf("Inserisci una sequenza di 0 e 1 terminata da 2\n");
```

```
scanf("%d", &bit);
```

```
while ( bit!=2)
```

```
{
```

```
    if ( bit == 0)
```

```
    {
```

```
        cont = cont + 1;
```

```
        if (cont > maxlung)
```

```
            maxlung = cont;
```

```
    }
```

```
    else
```

```
        cont = 0;
```

```
        scanf("%d", &bit);
```

```
}
```

```
printf("La lunghezza della piu' lunga sottosequenza di 0 e' %d\n", maxlung);
```




Esercizio 7

Un intero $N > 1$ è detto **primo** se i suoi unici divisori sono **1** e **N**

Scrivere un programma che legge da stdin un intero e determina se è primo

Algoritmo (*inefficiente*): provare se tra i numeri compresi tra **2** e **N-1** c'è un divisore di **N**



Algoritmo(*inefficiente*) Esercizi

Variabili utilizzate (tipo intero):

numero: valore letto, **provadiv**: possibile divisore di **numero**, **trovatodiv**: diventa vero (1) se si trova un divisore di **numero**

- **provadiv=2**; **trovatodiv=0** (inizializzazione)
- leggi valore (registrato in **numero**)
- finché **provadiv** < **numero**
 - se **provadiv** divide **numero**: **trovato=1**
 - **provadiv=provadiv+1**
- se **trovato=1**: **numero** non è primo
altrimenti: **numero** è primo

Soluzione

```
int numero, provadiv = 2, trovatodiv = 0;
```

```
printf ("Inserire un numero intero maggiore di uno: \n");
```

```
scanf ("%d",&numero);
```

```
while (provadiv < numero)
```

```
{
```

```
    if ((numero % provadiv) == 0)
```

```
        trovatodiv = 1;
```

```
    provadiv = provadiv + 1;
```

```
}
```

```
if (trovatodiv==0)
```

```
    printf("Il numero %d e' un numero primo\n", numero);
```

```
else
```

```
    printf("\nIl numero %d non e' un numero primo\n", numero);
```