Introduzione al C

Corso di Fondamenti di Informatica

Ingegneria Clinica

Esercitazione 7

DIPARTIMENTO DI INFORMATICA
E SISTEMISTICA ANTONIO RUBERTI

SAPIENZA
UNIVERSITÀ DI ROMA

Raffaele Nicolussi

Esercizio 1

- Si scriva una funzione in linguaggio C che riceve in ingresso un numero intero A letto da tastiera, lo riduca ad un valore compreso tra 0 e 127 mediante **sottrazione ripetuta di un adeguato numero di volte del valore 128** e lo restituisca come carattere ASCII. La funzione chiamante lo stampi sul video

Soluzione 1

```
…
        printf ("Inserisci un numero: ");
        scanf("%d",&a);
        while (a >= 128) {
                a -= 128;
…
```

Esercizio 2

- Si scriva un programma C che legga da tastiera un intero N>=0 e successivamente chieda all'utente di inserire N numeri interi stampando, alla fine, a video il maggiore tra essi, la loro media e la radice quadrata della somma.
- Si noti che per effettuare la radice quadrata esiste la funzione double sqrt (double) definita nel file di header: <math.h>

Soluzione 2

```
…
for (i=0; i<N; i++) {
            printf ("Inserisci un numero: ");
            scanf("%d",&a);
            if (max < a) {
                  max = a;
 {
            somma += a;
 {
       r = somma / 5;
       printf ("Il valore massimo inserito è: %d\n", max);
       printf ("La radice quadrata della somma è: %.3f\n",
      sqrt(somma));
       printf ("La media è: %.3f\n", r);
…
```

Esercizio 3

- Chiedere all'utente quanti tentativi vuole effettuare per risolvere il gioco
- Generare un numero a caso e chiedere all'utente un numero fino a quando non é uguale a quello generato casualmente o non si è superato il numero max di tentativi
- Dire ogni volta se il numero immesso é > o < di quello iniziale.

Generazione di un numero casuale in C

- Come faccio a generare un numero casuale compreso tra 1 e 100?

- Per generare un numero casuale, bisogna richiamare la funzione rand, dichiarata in stdlib.h
- Per assegnare a una variabile (i) un numero casuale da 1 a 100 con rand bisogna scrivere:
  i = rand() % 100 + 1;

- Prima di richiamare la funzione rand, è opportuno richiamare la funzione di randomizzazione **srand** (dichiarata in stdlib.h), che accetta come argomento un seme (del tipo unsigned int), che “insemina” la funzione rand
- In questo modo si induce la funzione a generare una diversa sequenza di numeri casuali ad ogni esecuzione del programma, a patto che sia fornito ogni volta un seme diverso.
- Si può fornire come argomento a srand l’orario in quell’istante, letto dall’orologio interno del computer da parte della funzione time, dichiarata in **time.h.**

```
#include <stdlib.h>
#include <time.h>
…
int i;
…
srand(time(NULL));
i = rand() % 100 + 1;
…
```

Esercizio 3 bis

- Chiedere all'utente quanti tentativi vuole effettuare per risolvere il gioco
- **Chiedere all'utente un numero MIN e MAX entro i quali il computer sceglierà il suo numero casuale**
- Generare un numero a caso tra MIN e MAX e chiedere all'utente un numero fino a quando non é uguale a quello generato casualmente o non si è superato il numero massimo di tentativi
- Dire ogni volta se il numero immesso é > o < di quello iniziale.
- Stampare il numero di tentativi occorsi per risolvere il gioco.
- Tenere traccia del punteggio migliore ottenuto tra diverse partite

Soluzione 3

```
…
    int a, t, r;
    r = rand();
    a = -1;

   while (a != r) {
      printf ("Inserisci un numero: ");
      scanf("%d",&a);
      if (a != r) {
          if (a<r) {
         printf ("Inserisci un numero maggiore.\n");
       } else {
      printf ("Inserisci un numero minore.\n");
                }
      }
```

Esercizio 4: Fibonacci

- Letti in input due numeri floating point *x* ed *y* e un intero *n*, calcola l'*n*-esimo valore della successione di Fibonacci, definita come segue:
  $U_0 = x$
  $U_1 = y$
  $U_{n+2} = U_{n+1} + U_n$

Soluzione 4

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

int main(void) {
  float x,y,z;
  int i,n;

  printf("Digita il primo ed il secondo valore della successione: ");
  scanf("%f %f", &x, &y);
  printf("Quale termine della successione vuoi calcolare? ");
  scanf("%d", &n);
  if (n==0) {
                z=x;
              } else {
```

Soluzione 4

```
if (n==1) {
      z=y;
    } else {
      for(i=2; i<=n; i++) {
        z = y+x;
        x = y;
        y = z;
      }
    }
  }
  printf("Il termine %d della successione e' %f.\n", n, z);
  return(1);
}
```

Esercizio 5

- Legge in input un intero *n* ed un floating point *x*>0 e calcola la somma delle potenze di *x*, da 0 ad *n*.

Soluzione 5

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

float potenza(int n, float x) {
  int i;
  float r=1;

  for (i=0; i<n; i++) {
    r=x*r;
  }
  return(r);
}
```

Soluzione 5

```
int main(void) {
  float somma, y;
  int m, j;

  printf("Digita un intero: ");
  scanf("%d", &m);
  printf("Digita un numero: ");
  scanf("%f", &y);
  somma=0;
  j=0;
  while (j<=m) {
    somma = somma + potenza(j,y);
    j=j+1;
  }
  printf("La somma delle potenze di %3.2f da 0 a %d e' %4.2f.\n", y, m, somma);
  return(1);
}
```

Esercizio 6

- Trasformare (se non già fatto) la generazione dei numeri di Fibonacci in una funzione
- *int numero = fibonacci (x);*
- Che genera l'x termine della serie di **fibonacci**
- Scrivere la funzione **sommaNfibonacci (n)** che somma i primi n termini della serie di **fibonacci** richiamando la funzione **fibonacci**