Terza esercitazione

21/10/2022

Esercizio 1.

Si scriva una funzione che riceva in ingresso **tre** numeri **interi** x, y e z ne restituisca il **minimo**. Si crei poi una funzione di test che prenda in input **due** dei tre numeri dall'utente, x e y, e testi la tripletta (x, y, w) con $w \in [1, \max\{x, y\}]$.

<u>Nota</u>*: per testare la tripletta si intende testare ogni possibile tripla di valori ottenuta fissando x ed y, e variando w da 1 a $\max\{x, y\}$.*

Esempio di output:

```
> Inserisci il numero (x): 1
> Inserisci il numero (y): 3
> tripletta (1, 3, 1) - minimo -> 1
> tripletta (1, 3, 2) - minimo -> 1
> tripletta (1, 3, 3) - minimo -> 1
```

Esercizio 2.

Si scriva una funzione che, ricevuti in ingresso le **coordinate** $p_1 = (x_1, y_1)$ ed $p_2 = (x_2, y_2)$ di due punti del piano cartesiano, restituisca la loro **distanza euclidea** $d(p_1, p_2) = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$.

*<u>Nota</u>: la funzione radice quadrata (**sqrt**) in C viene fornita dalla libreria **math.h**, che va quindi importata come segue:*

```c
#include<stdio.h> // input output
#include<math.h>  // sqrt

double distanza(double x1, double y1, double x2, double y2){
  // codice vostro
}
```

Esempio di output:

```
> Inserire le coordinate (x1,y1) e (x2,y2) per cui si vuole calcolare la distanza:
> x1: 2
> y1: 5.5
> x2: 3
> y2: -1.6
> distanza tra (2,5.5) e (3,-1.6) = 7.17008
```

<u>Nota</u>: compilare il codice con il comando -lm per indicare al compilatore che nel programma c'è bisogno della libreria math.h; esempio:

```
> gcc esercizio_2.c -o esercizio_2.x -lm
```

Esercizio 3.

Si scriva una funzione che riceva in ingresso **due numeri double** x ed y e restituisca 1 se e solo se i due numeri sono **uguali a meno di un fattore** ϵ, ovvero $|x - y| < \epsilon$, e 0 in caso contrario. Il valore di ϵ deve essere un **parametro della funzione**.

Esempio di output:

```
> Inserire epsilon (e): 0.000001
> Inserire x: 0.001
> Inserire y: 1.001
> I due numeri sono diversi con epsilon 0.000001
```

Esercizio 4.

Dato un numero $n \in N, n > 0$ vogliamo stabilire se sia **primo** oppure no. Si implementi una funzione che testi la verità del predicato *"n è un numero primo"*.

Si crei inoltre un programma che testi la funzione per i valori di input che vanno da 1 a 100 (incluso). Il programma deve stabilire il **numero di numeri primi trovati** nell'intervallo.

Esempio di output:

```
> Inserire il numero (n): 7
> 7 è un numero primo.
> Numero di numeri primi trovati da 1 a 100: [scopritelo voi 😉]
```

Esercizio 5.

Scrivere una funzione che, preso in input un numero intero in base binaria, lo converta a base decimale.

Hint: l'algoritmo per passare da base binaria a base decimale, dato un numero le cui cifre sono $c_n, c_{n-1}, \dots, c_1, c_0$ espresso in base 2 è: $c_0 * 2^0 + c_1 * 2^1 + \cdots + c_{n-1} * 2^{n-1} + c_n * 2^n$

Esempio concreto: $1101_2 = 1 * 2^0 + 0 * 2^1 + 1 * 2^2 + 1 * 2^3 = 13_{10}$

Esempio di output:

```
> Inserire il numero in base 2: 1001
> L'equivalente in base 10 è 9
```

<u>Nota</u>: compilare il codice con il comando -lm per indicare al compilatore che nel programma c'è bisogno della libreria math.h; esempio:

```
> gcc esercizio_5.c -o esercizio_5.x -lm
```

Esercizio 6.

Si implementi una funzione che calcola la seguente successione fino al termine n-esimo:

$$a_n = \begin{cases} \frac{1}{2}, & \text{se } n = 1 \\ \frac{a_{n-1}+1}{2}, & \text{altrimenti} \end{cases}$$

Verificare che il limite della successione è 1.

Esercizio 7.

Si implementi una funzione che calcola la seguente successione fino al termine n-esimo:

$$a_n = \begin{cases} p, & \text{se } n = 1 \\ \frac{1}{2}\left(a_{n-1} + \frac{p}{a_{n-1}}\right), & \text{se } n > 1 \end{cases}$$

Verificare che il limite della successione è $\sqrt{p}$.

<u>Nota</u>: compilare il codice con il comando -lm per indicare al compilatore che nel programma c'è bisogno della libreria math.h; esempio:

```
> gcc esercizio_7.c -o esercizio_7.x -lm
```

Esercizio 8.

Si consideri la seguente serie:

$$s_n = \sum_{k=0}^{n} \frac{1}{(2k+1)^2}$$

Si verifichi che per $n \to \infty$ la serie tenda a $\frac{\pi^2}{8}$.

Esercizio 9.

Implementare una funzione che calcoli il termine n-esimo della serie

$$s_n = \sum_{i=1}^{n} \frac{i * k^{i+1}}{3^i}$$

Dove k è un parametro della funzione. Provare a capire per quali valori di k la serie converge.

Hint: Nella serie è presente un numero che può dare un indizio.