Laboratorio 2 – Esercizi

Elementi di Informatica e Programmazione

Lab 2 – Es 0

Affrontare e risolvere tutti gli esercizi dei laboratori precedenti che non sono ancora stati affrontati e risolti!

Non ha molto senso tentare di risolvere gli esercizi di questo laboratorio senza aver fatto quelli precedenti...

Lab 2 – Es 1

Progettare il programma **twoNumbers.py** che chieda all'utente di inserire due numeri e ne visualizzi:

- la somma
- il prodotto
- il valore medio
- il valore massimo
- il valore minimo
- il valore assoluto della differenza

Lab 2 – Es 2

Progettare il programma **oddNumber.py** che:

- chieda all'utente di fornire un numero dispari (ovviamente intero...)
- se il numero fornito è pari, chieda di nuovo all'utente di fornire un numero dispari, dove aver visualizzato una segnalazione d'errore
- se il numero fornito (al primo o secondo tentativo) è dispari, scriva un messaggio di congratulazioni all'utente, riportando anche il numero fornito (*es. "Bravo! Il numero 25 è dispari"*)
- se, invece, neanche il secondo numero fornito è dispari, scriva un messaggio di disappunto

Attenzione: se il primo numero fornito è dispari, il programma **NON** deve chiedere un secondo numero. 

Lab 2 – Es 3 (continua)

Progettare il programma **isLeapYear.py** che segnali all'utente se il numero intero positivo che ha introdotto corrisponde a un anno bisestile oppure no.

- Ricordare che un anno è **bisestile** se è divisibile per 4. Fanno **eccezione** gli anni divisibili per 100, che non sono bisestili, e gli anni divisibili per 400, che invece sono bisestili (pur essendo divisibili anche per 100): tali eccezioni esistono però solo dopo l'adozione del calendario gregoriano, che avvenne nel **1582**.
- Il programma NON dovrà accettare il dato in ingresso (visualizzando, quindi, un messaggio d'errore) quando è un numero intero non positivo.

Lab 2 – Es 3 (continua)

Il programma dovrà essere **collaudato** con i seguenti valori:

- un valore negativo (segnalazione d'errore)
- il valore zero (segnalazione d'errore)
- l'anno 1582 (che non è bisestile, non essendo divisibile per 4)
- anni maggiori di 1582
  - un anno divisibile per 400, quindi bisestile (es. 2000)
  - un anno divisibile per 100 ma non per 400, quindi non bisestile (es. 1900)
  - un anno divisibile per 4 ma non per 100, quindi bisestile (es. 2016)
  - un anno non divisibile per 4, quindi non bisestile (es. 2017)
- anni minori di 1582
  - un anno divisibile per 400, quindi bisestile (es. 1200)
  - un anno divisibile per 100 ma non per 400, comunque bisestile (es. 1400)
  - un anno divisibile per 4 ma non per 100, quindi bisestile (es. 1580)
  - un anno non divisibile per 4, quindi non bisestile (es. 1581)

Lab 2 – Es 3

Risolvere l'esercizio in due modi:

- senza utilizzare gli operatori booleani (and, or e not)

- utilizzando gli operatori booleani

Prima di scrivere codice, ragioniamo con i diagrammi di flusso…

Lab 2 – Es 4 (continua)

Progettare il programma **checkCircles.py** che

- **chieda all'utente** di inserire la descrizione di due circonferenze nel piano cartesiano:
  - coordinata x del dentro della prima circonferenza
  - coordinata y del centro della prima circonferenza
  - raggio della prima circonferenza
  - coordinata x del dentro della seconda circonferenza
  - coordinata y del centro della seconda circonferenza
  - raggio della seconda circonferenza

Lab 2 – Es 4

Progettare il programma **checkCircles.py** che

- visualizzi uno dei messaggi seguenti:
  - Le due circonferenze sono coincidenti
  - Le due circonferenze sono tangenti esternamente
  - Le due circonferenze sono tangenti internamente
  - Le due circonferenze sono secanti
  - Le due circonferenze sono concentriche (ma non coincidenti)
  - Le due circonferenze sono una interna all'altra, ma non concentriche
  - Le due circonferenze sono (reciprocamente) esterne

  Per dubbi 'matematici':
  http://www.ripmat.it/mate/f/fj/fje.html

Lab 2 – Es 5 (continua)

- Progettare il programma **romanNumeral.py** che converta un numero intero positivo nel corrispondente valore espresso nel sistema di numerazione romano.
- I numeri romani sono costituiti da sequenze di sei diverse possibili cifre, aventi il valore seguente:
- I = 1
- V = 5
- X = 10
- L = 50
- C = 100
- D = 500
- M = 1000

Lab 2 – Es 5

Queste sono le regole di conversione:

- si possono rappresentare soltanto i numeri minori di 4000
- (come nel sistema di numerazione decimale posizionale) le migliaia, le centinaia, le decine e le unità vengono espresse separatamente e rappresentate in questo ordine
- i numeri da 1 a 9 sono rappresentati, in ordine crescente, da I, II, III, IV, V, VI, VII, VIII, IX; come si può notare, una lettera I che precede la lettera V o X rappresenta un'unità che viene **sottratta** dal valore del numero; inoltre, non si possono avere più di tre I consecutive
- le decine sono gestite come le unità, tranne per il fatto che, al posto delle lettere I, V e X, si usano rispettivamente le lettere X, L e C
- le centinaia sono gestite come le unità, tranne per il fatto che, al posto delle lettere I, V e X, si usano rispettivamente le lettere C, D e M
- Il programma deve chiedere all'utente un numero intero positivo minore di 4000 e visualizzare il corrispondente numero romano. Esempio: 1978 diventa MCMLXXVIII.