Programmazione

Appello del 24/01/2012

Esercizio 1 (8 punti)

a. (4 punti) Scrivere l'output del seguente programma Java

```
public class Main {
public static void main(String[] args) {
        int [] A = {55,83,21,16,43,10,56};
        stampa(A);
        enigma(A);
        stampa(A);
    }

public static void stampa (int [] A ){
        for (int x:A) {
            System.out.print (x + " ");
        }
        System.out.println();
    }

public static void enigma (int A[]){
      int i=A.length-2;
      do {
        A[i+1]=A[i]+1;
        i--;
      } while (i>=0);
    }
}
```

b. (4 punti) Riscrivere il metodo enigma in modo equivalente (ossia in modo che QUALSIASI sia l'array A lo modifichi nello stesso modo) facendo uso esclusivamente del costrutto iterativo **for**

ESERCIZIO FACOLTATIVO:
Riscrivere il metodo enigma in modo equivalente facendo uso della ricorsione, senza utilizzare alcun costrutto iterativo (né for, né while, né do-while). Se necessario, si possono utilizzare anche altri metodi statici di appoggio, a patto che non usino alcun costrutto iterativo.

Esercizio 2 (8 punti)

Scrivere un metodo **public static boolean verifica (int [][] A, int [][] B, int [][] C)** che presi in input tre array bidimensionali di numeri interi restituisce ***true*** se e solo se valogono tutte le seguenti condizioni:

- A, B e C sono tutte e tre matrici rettangolari
- A, B e C hanno le stesse dimensioni
- A+B = C

Attenzione:

- Per svolgere il compito si hanno a disposizione **90** minuti (**50** minuti per chi deve svolgere solo una parte).
- Scrivere **subito** nome, cognome, matricola e numero del compito su OGNI FOGLIO.
- Durante la prova scritta non è possibile abbandonare l'aula.
- Non è ammesso **per nessun motivo** comunicare in qualsiasi modo con altre persone
- **Non** è possibile consultare appunti, libri, dispense o qualsiasi altro materiale.
- Qualsiasi strumento elettronico di calcolo o comunicazione (telefoni cellulari, calcolatrici, palmari, computer, etc…) deve essere **completamente disattivato** e **depositato in vista sulla cattedra**

SECONDA PARTE

Esercizio 3 (8 punti)

Si implementino in Java le classi **Articolo** e **Legge**.
La *classe* **Articolo** ha i seguenti attributi:

- **numero** (un intero)
- **testo** (una stringa)

ed i seguenti metodi di istanza:

- *costruttore* che crea un oggetto della *classe* **Articolo** assegnando numero e testo.
- metodi "*get*" per tutti gli attributi, cioè metodi che restituiscono i valori di ciascun attributo;
- metodo "toString"

La *classe* **Legge** ha i seguente attributi:

- **titolo** (una Stringa)
- **articoli** (un arrayList di Articolo)

ed i seguenti metodi di istanza:

- *costruttore* che crea un oggetto della *classe* **Legge** prendendo in input il titolo della legge
- metodo **public void *addArticolo (Articolo a),*** che aggiunge l'articolo a all'arrayList articoli, **solo se** nella legge non è già presente un articolo con lo stesso numero.
- metodo **public String getTitolo ()** che restituisce il titolo della legge.
- metodo **public int getNArticoli ()** che restituisce il numero di articoli della legge.
- metodo **public Articolo getArticolo (int k)** che restituisce il k-esimo articolo della legge, ovvero l'articolo avente k come numero (non necessariamente quello in posizione k). Se un tale articolo non esiste, viene restituito null.
- metodo "toString", che sfrutta il metodo toString della classe Articolo

ESERCIZIO FACOLTATIVO:
Fare in modo che il metodo toString della classe Legge stampi gli articoli in ordine crescente di numero (si possono utilizzare, se lo si ritiene necessario, anche variabili di istanza ausiliarie).

Esercizio 4 (8 punti)

- Si scriva una classe astratta **Funzione** per rappresentare funzioni sul piano cartesiano
- La classe deve avere un metodo astratto **public abstract double calcola (double x);** che sarà implementato dalle classi che estendono **Funzione** con lo scopo di calcolare il valore della funzione nel punto **x**.
- La classe deve inoltre implementare un metodo **public boolean LEq (Funzione f, double x)** che ritorna ***true*** se e solo se il valore della funzione su cui è invocato calcolata nel punto **x** è minore o uguale al valore della funzione **f** calcolata sempre in **x** *[si sfrutti il metodo* **calcola***]*
  Ad esempio, se f e g sono due funzioni, f.LEq(g,3) deve ritornare true se e solo se f(3)≤g(3)
- *Si implementino infine le seguenti classi* ***concrete*** *che estendono* **Funzione***,* ognuna con le opportune variabili di istanza, ed in ognuna delle quali bisogna implementare un ***costruttore*** ed **il** metodo **calcola***:*
  - La classe **Retta**, il cui costruttore prende due double (m e q) tali che la funzione rappresentata è **f(x)=mx+q**
  - La classe **Parabola**, il cui costruttore prende tre double (a,b,c) tali che la funzione rappresentata è **$f(x)=ax^2+bx+c$**