Prova scritta dell'esame di Fondamenti di Informatica

Data: 22 marzo 2007 **TRACCIA C** **Durata della prova: 2 ore**

Cognome e Nome		*Corso*		*Matricola*	

Esercizio 1

Si consideri il seguente programma:

```
public class Esercizio1C{
  static void elabora(int[] v) {
    for (int i = 0; i < v.length; i++) {
      int s = 1;
      if (i % 2 == 1) {
        for (int j = i + 1; j < v.length; j++)
          s *= v[j];
      }else{
        for (int j = i - 1; j >= 0; j--)
          s *= v[j];
      }
      System.out.println(s);
    }
  }

  public static void main(String[] args) {
    int[] a = { 1, -1, 1, 1, 1, -1 };
    elabora(a);
  }
}
```

Si illustri il funzionamento del metodo *elabora* e, a titolo d'esempio, si mostri cosa viene stampato in output al termine dell'esecuzione del *main*.

Esercizio 2

Scrivere un metodo *verifica* che riceve un vettore **a** di **interi** (di dimensione pari) ed un intero **x**. Il metodo restituisce il valore **true** se per **ogni** elemento di **a** in posizione **dispari** il suo prodotto con l'elemento precedente è maggiore o uguale a **x** **e** la sua somma con l'elemento precedente è minore o uguale a **x**, **false** altrimenti. Ad esempio, se **a**=[7, 2, 3, 5, 8, 3, 9, 2] e **x**=15 il metodo restituirà **false**.

Esercizio 3

Si realizzi una classe Matrice per rappresentare matrici quadrate di double che contenga almeno i seguenti metodi:

1. un metodo *verificaSomme* che riceve una matrice quadrata di double **m** e restituisce **true** se si ottiene lo stesso risultato sommando gli elementi dell'ultima riga, gli elementi sulla prima colonna, e quelli sulla diagonale secondaria, restituisce **false** altrimenti;
2. un metodo *costruisciVettore* che riceve una matrice quadrata di double **m** ed un double **k** e restituisce un vettore di double **v**. Gli elementi di **v** sono calcolati considerando solo le colonne di **m** per cui la somma degli elementi è minore di **k**. In particolare, ogni elemento di **v** è pari alla somma tra gli elementi di una colonna che soddisfa la precedente condizione e gli elementi della diagonale principale di **m** (si veda l'esempio);
3. un metodo *sottomatrice* che riceve una matrice quadrata di double **m** e restituisce una sottomatrice **sm** ottenuta a partire da **m** (ignorando l'ultima riga se la dimensione è dispari). La sottomatrice **sm** deve contenere gli elementi di **m** i cui indici di riga e di colonna non sono entrambi pari né entrambi dispari (si veda l'esempio);
4. un metodo *main* che legge una matrice quadrata di double **m** e invochi opportunamente i metodi descritti nei punti precedenti.

Esempio:

m =

7.1	1.2	7.2	1.3	**6.3**
4.0	0.0	0.3	**2.3**	**2.1**
2.3	1.1	**3.1**	1.3	**3.2**
1.1	**5.1**	4.2	0.0	**4.1**
1.2	**2.0**	**5.2**	**1.8**	**5.5**

1. *verificaSomme(m)* → ***false***
2. *costruisciVettore(m, 15)* → *[**25.1** **22.4**]*

   (in particolare, la somma degli elementi della seconda colonna è minore di 15, e di conseguenza l'array risultante conterà in posizione 0 l'elemento 25.1 = 1.2+0.0+1.1+5.1+2.0+7.1+0.0+3.1+0.0+5.5
3. *sottomatrice(m)* → **sm =**

1.2	**1.3**
4.0	**0.3**
1.1	**1.3**
1.1	**4.2**
2.0	**1.8**