Corso di laurea in Informatica AA 2007/08 Programmazione, M.A. Alberti

Compitino II - 17 dicembre 2007

Cognome **Nome**

Matricola

---

1 peso 5

Data la seguente dichiarazione con inizializzazione:

```
String[] musicisti =
      {"vivaldi", "scarlatti", "pergolesi", "tartini", "monteverdi", "verdi"};
```

1. Scrivere l'istruzione di selezione per riportare la stringa **`pergolesi`**.	
2. Scrivere un'espressione per riportare l'indice della prima occorrenza del carattere **`'i'`** in **`tartini`**.	
3. Scrivere un'espressione per riportare il carattere '**`e`**' nella stringa **`monteverdi`**.	
4. Estrarre la sottostringa **`verdi`** dall'elemento che lo contiene.	
5. Scrivere un'istruzione che controlli se la sottostringa ottenuta alla domanda 4. sia uguale all'ultimo elemento dell'array e dire quale e' il risultato.	
6. Dire qual è l'indice dell'ultimo elemento dell'array	
7. Scrivere il valore di **`musicisti.length`**.	
8. Dire qual è la posizione della stringa **`scarlatti`**	
9. Implementare un ciclo per scorrere l'array alla ricerca della stringa riferita dalla variabile **`musicista`** al fine di riportarne la posizione nell'array	

---

2 peso 3

Dire cosa succede durante la compilazione e l'esecuzione del codice seguente:

```
public class CosaSuccede {
     public static void main(String[] args) {
          final int MAX = 3;
          StringBuffer[] archivio = new StringBuffer[MAX];
          for (int i=0;  i<archivio.length; i++) {
               archivio[i].append("posizione " + i);
               System.out.println (archivio[i]); }
     }
}
```

compilazione:

esecuzione:

---

3 peso 3

Dato il codice seguente:

```
      String[] unArray = new String[MAX];
```

```
for (int i=0; i<unArray.length; i++)
     System.out.println(unArray[i]);
```

L'esecuzione del ciclo-for causa errore in compilazione?
Se la compilazione ha successo, in esecuzione cosa viene stampato?

---

4 peso 5
Scrivete il metodo statico **`somma_parziale`** che riceve come parametro un array di tipo **`int[ ]`** e riporta un array contenente la somma parziale degli elementi del parametro. Es.: dato il **`vettore = [2, 5, 3, 8, 5]`** il metodo **`somma_parziale(vettore)`** produce in output l'array **`[2, 7, 10, 18, 23]`**.

```
..................... somma_parziale (int[ ] vett) {




}
```

---

5 peso 5
Assumendo le seguenti dichiarazioni

```
final int MAX = 6, LIMITE = 25;          int num1, num2, num3;
```

Scrivere l'ouput dei seguenti spezzoni di codice:

	`num1 = 18; num2 = 25; num3 = 26;`

`if (num2 < num1)`
`  if (num3 < LIMITE)`
`  System.out.println ("ROSSO");`
`else`
`  System.out.println ("BLU");`
`System.out.println ("VERDE");`	

	`num1 = 12; num2 = 9; num3 = 89;`

`if (num2 < num1)`
`  if (num3 < LIMITE)`
`  System.out.println ("ROSSO");`
`else`
`  System.out.println ("BLU");`
`System.out.println ("VERDE");`	

`while (num1 <= LIMITE) {`
`  if (num3++ / num1 < MAX)`
`  System.out.println ("quadri");`
`  else System.out.println ("picche");`
`  num1 *= 2;`
`}`	

	`num1:  num2:  num3:` `num1 = 14;`

`while (num1 >= MAX) {`
`  if (num1-- % 2 == 0) continue;`
`  System.out.println (num1);`
`}`	

	`num1:  num2:  num3:`

---

6 peso 4
Dato il codice:

```
final int MAX = 5;
int[][] struttura = new int[MAX][];
for (int i = 0; i < struttura.length; i++) {
   struttura[i] = new int[MAX];
   for (int j = 0; j < struttura[i].length; j++) {
        struttura[i][j] = (i+1) * (j+1);   } }
```

Dire che tipo rappresenta **`struttura`**:
Dire come **`struttura`** viene inizializzata nel ciclo-for:

---

7 peso 3
Nella classe seguente **`Rettangolo`**, completare il costruttore **`Rettangolo(double, double)`**, i metodi **`uguale(Rettangolo p)`**, **`perimetro()`** e **`area()`.**

```
public class Rettangolo {
   private double alt, larg;

   public Rettangolo(double a, double b){

   }
   public boolean uguale(Rettangolo p) {

   }
}
```

```
   public double perimetro() {

   }
   public double area() {

   }
   public void trasforma
       (double n_alt, double n_larg){
      this.alt = n_alt;
      this.larg = n_larg;
   }
```

---

8 peso 2
La classe seguente effettua un test della classe **`Rettangolo`**. Dire se il codice è corretto e scrivere l'output:

```
public class TestRettagolo {
   public static void main(String[] args) {
      double a = 1.0, b = 2.0;
      Rettangolo unRettangolo;
      unRettangolo.trasforma(a, b);
      System.out.println (unRettangolo);
   }
}
```

---

9 peso 3
Definire una classe **`Quadrato`** che estende la classe **`Rettangolo`**. La classe ha un campo privato di tipo `double` individuato dall'identificatore `lato`. Definire un costruttore con un parametro che verrà assegnato al campo `lato`.

---

10 peso 2
Sovrascrivere per la classe **`Quadrato`** il metodo **`perimetro()`**:

Dire in generale qual'è la norma che deve essere rispettata nella sovrascrittura:

---

11 peso 8

Vogliamo ora che la classe **`Rettangolo`** implementi l'interfaccia **`Comparable`** fornendo un'implementazione del metodo astratto **`compareTo (Object o).`**
Implementate il metodo **`compareTo`** considerando come criterio di ordinamento l'area dei due rettangoli: il parametro indiretto e il parametro effettivo. Si ricorda che il metodo **`compareTo`** riporta il valore **`0`** se le due aree sono uguali; **`-1`** se l'area del parametro indiretto è minore dell'area del parametro e **`1`** in caso contrario.

---

12 peso 5

Consideriamo il seguente spezzone di codice:

```
Rettangolo[] lista = new Rettangolo[MAX];
…
```

Si chiede di inizializzare **`lista`** con oggetti alternativamente di classe **Rettangolo** o **Quadrato** in funzione della parità dell'indice che indica la posizione.
Quindi si consideri l'istruzione seguente e si dichiari in generale quale metodo **`perimetro()`** viene invocato:

```
for (int i=0; i<lista.length; i++)
            per = lista[i].perimetro();
```