Corso di laurea in Comunicazione digitale — Programmazione I

AA 2009/10 — Prof. M.A. Alberti

III Appello - 8 giugno 2010

Cognome **Nome**

Matricola

1 (6 punti)

Data la gerarchia di classi contenuta nel file `Quadrati.java` completare il codice in modo che non ci siano errori durante la compilazione del file e in modo che il costruttore della classe `Quadrati` istanzi correttamente un poligono di 4 lati.

```
class Figure {
   int lati;

   Figure(int i){
      lati = i;
      System.out.println (“Figure di ” + lati + “ lati”)
  }
}

class Poligoni extends Figure {

   Poligoni (int i) { // costruttore di un poligono

      super(i);

      System.out.println ("Poligono di " + i + " lati");
  }
}

public class Quadrati extends Poligoni {

   Quadrati() { // costruttore di un quadrato

      super(4);

      System.out.println ("Il quadrato ha 4 lati uguali");
   }

   public static void main (String[] args) {
      Quadrati mioQuadrato = new Quadrati();
   }
}
```

I costruttori di `Poligoni` e `Quadrati` stampano un messaggio che segnala l'avvenuta costruzione dell'oggetto di tipo `Figure`. Quindi si scriva l'output che si ottiene all'esecuzione del metodo `main`, della classe `Quadrati`.

Figura di 4 lati
Poligono di 4 lati
Il quadrato ha 4 lati uguali

2 (5 punti)

Scrivere la classe `Sacchetti.java` che genera 100 numeri pseudo-casusali compresi tra 1 e 12 (estremi inclusi) e che conta le occorrenze dei numeri generati tra 1 e 4, quelli tra 5 e 8 e gli altri e scriva un rapporto finale. I contatori si chiamano `sacco_1`, `sacco_2` e `sacco_3`.

```
import java.util.Random;

public class Sacchetti{
   public static void main (String[] args){
      int sacco_1=0, sacco_2=0, sacco_3=0, generato;
      Random rand = new Random ();
      for (int i=0; i < 100; i++) {
         generato = Math.abs(rand.nextInt() % 12) + 1;
         System.out.println(generato);
         if (generato <= 4)
            sacco_1++;
         else if (generato <= 8)
           sacco_2++;
         else sacco_3++;
      }
      System.out.println ("sacchetto 1: " + sacco_1);
      System.out.println ("sacchetto 2: " + sacco_2);
      System.out.println ("sacchetto 3: " + sacco_3);
   }
}
```

3 (4 punti)

Stabilire qual'è l'effetto dell'esecuzione dei cicli:

`for (int i = 1; i < 50; i++) {`
`  if (i == 29) break;`
`  if (i % 4 != 0) continue;`
`  System.out.println (i);`
`}`
	4
	8
	12
	16
	20
	24
	28

`int i = 0, j;`
`while (true) { i++; j = i * 2;`
`  if (j == 12) break;`
`  if (i % 3 == 0) continue;`
`  System.out.println (i);`
`}`
`System.out.println ("i: " + i);`
`System.out.println ("j: " + j);`
	1
	2
	4
	5
	i: 6
	j: 12

4 (4 punti)

Assumendo le dichiarazioni seguenti:

```
final int MAX = 10;
```

```
final int MIN = 5;
int i, s = 0, valore;
```

Dire qual'è il valore delle variabili `i, s` e `valore` dopo aver eseguito l'istruzione:

```
for (i=2; valore > MIN && valore <= MAX; valore++, i++, s+=2*++i);
```

Calcolate tre cicli `for` per diversi valori della variabile `valore`, considerandoli indipendenti l'uno dall'altro.

valore	i	s	valore
4	2	0	4
8	8	36	11
20	2	36	20

Se i cicli fossero eseguiti sequenzialmente otterremmo valori diversi? **SI** NO

Per quale variabile? **la variabile `s` che nel terzo ciclo avrebbe valore 0; tutto il resto rimane invariato**

---

5 (5 punti)

Scrivere un metodo per rimuovere un elemento in posizione data da un array che verrà riportato all'ambiente chiamante modificato. Sia la posizione dell'elemento da rimuovere sia l'array sono passati come parametri. Per rimuovere l'elemento alla posizione indicata si deve procedere a eseguire lo spostamento di una posizione a sinistra di tutti i valori dell'array fino alla fine dello stesso. Si supponga cha l'array sia un array di interi.

```
int[] public static rimuovi(int i, int[] a) {
   for (int j = i; j < a.length-1;)
            a[j]=a[++j];
   a[a.length-1] = 0;
   // altrimenti rimane il valore prima dello slittamento
   return a;
}
```

E' ragionevole supporre che il metodo sia statico poiche' l'array viene apssato come parametro diretto e non come parametro indiretto (l'oggetto su cui chiamare il metodo)

---

6 (6 punti)

Le seguenti istruzioni, scritte in ordine sparso, rappresentano il corpo di un metodo che legge due numeri interi da input e li divide producendo a stampa il quoziente, ma rientra in un ciclo di lettura se si verifica il tentativo di dividere per 0. Ricomporre le istruzioni in ordine corretto. Si noti che potrebbero mancare parentesi graffe e/o punto e virgola. Mancano anche alcuni dichiarazioni ovvie e l'apertura dei canali di input e output che trascuriamo.

```
n1 = in.readInt("Un intero: ");
boolean notOk = true;
```

```
r = n1/n2;
try {
out.println(r);
while (notOk) {
n2 = in.readInt("Un secondo intero: ");
notOk = false;
catch (ArithmeticException e) {
out.println("Tentativo di divisione per 0");
```

Quello che si vuole ottenere e' un ciclo che consenta di recuperare l'errore di tentata divisione per 0. Quindi si vuole ottenere il funzionamento seguente:

```
Un intero: 5
Un secondo intero: 0
Tentativo di divisione per 0
Un intero: 5
Un secondo intero: 2
2.0
```

Ad esempio con il metodo:

```
    public static void main(String[] args) {
        ConsoleInputManager in = new ConsoleInputManager();
        ConsoleOutputManager out = new ConsoleOutputManager();
        int n1, n2;
        double r;
        boolean notOk = true;
        while (notOk) {
            try {
                n1 = in.readInt("Un intero: ");
                n2 = in.readInt("Un secondo intero: ");
                r = n1/n2;
                out.println(r);
                notOk = false;
            }
            catch (ArithmeticException e) {
                out.println("Tentativo di divisione per 0");
            }
        }
    }
```