E16 – Esercizi sulla realizzazione di classi

Esercizio 1 (esercizio 7.1 del libro di testo). Nella classe `Studente` definita di seguito indicare le variabili d'istanza, le variabili di classe e le variabili locali. Indicare inoltre i costruttori, i metodi di istanza e i metodi di classe, e per ciascun metodo dire se è di accesso o di modifica.

```
public class Studente{
	private String nome, cognome;
	private int matricola;
	private static int ultimaMatricola = 0;

	public Studente(String nome, String cognome){
		this.nome = nome;
		this.cognome = cognome;
		Studente.ultimaMatricola++;
		this.matricola = Studente.ultimaMatricola;
	}

	public String getNome(){
		return this.nome;
	}

	public String getCognome(){
		return this.cognome;
	}

	public int getMatricola(){
		return this.matricola;
	}

	public String toString(){
		return "Nome = " + this.getNome() +
		", cognome = " + this.getCognome() +
		", matricola = " + this.getMatricola();
	}

	public static int getProssimaMatricola(){
		return Studente.ultimaMatricola+1;
	}

	public void setNome(String n){
		this.nome = n;
	}

	public void setCognome(String n){
		this.cognome = n;
	}
}
```

Esercizio 2 (esercizio 7.2 del libro di testo). Dire che tipo di oggetti modella la classe `Studente` definita nell'Esercizio 1, e spiegare cosa fa ciascuno dei metodi definiti nella classe. Inoltre, dopo aver editato e compilato la classe `Studente`, si realizzi una classe di test di nome `TestStudente`, il cui metodo `main` effettui le seguenti operazioni:

- fa inserire all'utente nome e cognome di due studenti e crea i corrispondenti oggetti `Studente`;

- visualizza all'utente una descrizione completa dei due studenti inseriti, utilizzando il metodo `toString` della classe `Studente`;
- visualizza all'utente il numero di matricola dell'eventuale prossimo studente;
- chiede all'utente di ridefinire i dati (nome e cognome) del primo studente inserito, ed aggiorna i dati del corrispondente oggetto `Studente`;
- visualizza all'utente una descrizione completa del primo studente, al fine di mostrare come i dati dell'oggetto sono effettivamente cambiati.

Esercizio 3 (esercizio 7.3 del libro di testo). Definire una classe di nome `EquazioneDiPrimoGrado` i cui oggetti rappresentano equazioni di primo grado nella forma *ax + b = 0*. La classe deve avere i seguenti costruttori e metodi:

```
/* costruttore: crea un oggetto della classe
con i coefficienti specificati
*/
public EquazioneDiPrimoGrado(double a, double b)
/* calcola e restituisce la soluzione dell'equazione
*/
public double soluzione()
/* reimposta i coefficienti dell'equazione
*/
public void cambiaCoefficienti(double a, double b)
/* restituisce una descrizione dell'equazione
nella forma ax+b=0
*/
public String toString()
```

Scrivere inoltre una classe di test per la classe `EquazioneDiPrimoGrado`.

Esercizio 4 (esercizio 7.6 del libro di testo). Realizzare una classe di nome `ContoCorrente`, le cui istanze modellano conti correnti bancari gestiti da una generica banca. Un oggetto `ContoCorrente` deve avere i seguenti attributi:

- nome dell'intestatario del conto (una stringa),
- cognome dell'intestatario del conto (una stringa),
- numero di conto (un intero),
- ammontare del conto in euro (un numero reale).

La classe `ContoCorrente` ha inoltre un attributo statico, di nome `massimoScoperto`, che indica (in valore assoluto) il massimo valore di scoperto consentito per ogni conto corrente (cioè di quanto al più l'ammontare di un qualunque conto corrente può essere negativo). La classe deve avere i due costruttori seguenti:

```
/* crea un oggetto ContoCorrente con intestatario e numero di conto specificati;
l'ammontare iniziale viene fissato a 0 dal costruttore */
public ContoCorrente(String nome, String cognome, int num)

/* crea un oggetto ContoCorrente con intestatario, numero di conto e ammontare
iniziale specificati */
public ContoCorrente(String nome, String cognome, int num, double ammontare)
```

La classe deve inoltre avere i seguenti metodi di istanza:

```
/* aggiunge al conto corrente il valore specificato (che potrebbe anche essere
negativo) */
public void aggiungi(double valore)

/* restituisce true se l'ammontare corrente risulta inferiore allo scoperto
massimo consentito, e restituisce false altrimenti
*/
public boolean isScoperto()

/* restituisce una descrizione completa del conto corrente sotto forma di
stringa */
public String toString()
```

Infine, la classe ha il seguente metodo statico:

```
/* reimposta il valore del massimo scoperto con il nuovo valore specificato */
public static void setMassimoScoperto (double valore)
```

Dopo aver editato e compilato la classe `ContoCorrente`, realizzare una classe di test che permetta di verificarne il corretto funzionamento.

Soluzioni

Esercizio 1 - svolgimento. Segue una descrizione sintetica di tutti gli elementi che il testo richiede di inviduare.

Variabili di istanza: `nome` e `cognome` (di tipo `String`), `matricola` (di tipo `int`)

Variabili di classe: `ultimaMatricola` (di tipo `int`)

Variabili locali (inclusi i parametri):

- `nome` e `cognome` (parametri del costruttore)
- `n` (parametro del metodo `setNome`)
- `n` (parametro del metodo `setCognome`)

Costruttori e metodi:

- costruttore: `public Studente(String nome, String cognome)`
- metodi di istanza di accesso: `getNome`, `getCognome`, `getMatricola`, `toString`
- metodi di istanza di modifica: `setNome`, `setCognome`
- metodi di classe di accesso: `getProssimaMatricola`

Esercizio 2 - svolgimento. La classe `Studente` modella studenti universitari. Precisamente, ogni oggetto `Studente` rappresenta uno studente di uno stesso ateneo, descritto tramite nome, cognome e numero di matricola.

- Il costruttore permette di istaziare un oggetto Studente specificandone nome e cognome; il numero di matricola del nuovo studente viene assegnato automaticamente dal costruttore sulla base della matricola precedente (ultima assegnata).
- I metodi `getNome`, `getCognome` e `getMatricola`, restituiscono rispettivamente nome, cognome e matricola dell'oggetto `Studente` ricevente. Per contro, i metodi `setNome` e `setCognome`, permettono di cambiare nome e cognome di un oggetto `Studente`.
- Il metodo `toString` fornisce, sotto forma di unica stringa, una descrizione completa dello studente rappresentato dall'oggetto ricevente.

Esercizio 3 - svolgimento. Vengono di seguito riportati il codice della classe `EquazioneDiPrimoGrado` e il codice di una possibile classe di test.

```
public class EquazioneDiPrimoGrado{
      // attributi
      private double coeffA, coeffB;

      /* costruttore: crea un oggetto della classe
      con i coefficienti specificati
      */
      public EquazioneDiPrimoGrado(double a, double b){
```

```
            this.coeffA = a;
            this.coeffB = b;
      }

      /* calcola e restituisce la soluzione dell'equazione
      */
      public double soluzione(){
            // x = -b/a
            return -this.coeffB/this.coeffA;
      }

      /* reimposta i coefficienti dell'equazione
      */
      public void cambiaCoefficienti(double a, double b){
            this.coeffA = a;
            this.coeffB = b;
      }

      /* restituisce una descrizione dell'equazione
      nella forma ax+b=0
      */
      public String toString(){
            return (this.coeffA + "x + " + this.coeffB + " = 0");
      }
}


import fond.io.*;

public class TestEquazioneDiPrimoGrado{
      public static void main(String[] args){
            InputWindow in = new InputWindow();

            // test del costruttore e dei metodi toString e soluzione
            double a = in.readDouble("coefficiente a?");
            double b = in.readDouble("coefficiente b?");

            EquazioneDiPrimoGrado eq = new EquazioneDiPrimoGrado(a,b);

            OutputWindow out = new OutputWindow();
            out.writeln("Equazione: " + eq.toString());
            out.writeln("Soluzione = " + eq.soluzione());

            // test del metodo cambiaCoefficienti
            a = in.readDouble("Cambia il coefficiente a");
            b = in.readDouble("Cambia il coefficiente b");
            eq.cambiaCoefficienti(a,b);
            out.writeln("Equazione: " + eq.toString());
            out.writeln("Soluzione = " + eq.soluzione());
      }
}
```

Esercizio 4 - svolgimento. Vengono di seguito riportati il codice della classe `ContoCorrente` e il codice di una possibile classe di test.

```
public class ContoCorrente{
      // attributi
      private String nome, cognome;
      private int numConto;
      private double saldo;
      private static double massimoScoperto = 1000;
```

```
	/* crea un oggetto ContoCorrente con intestatario
	e numero di conto specificati; l'ammontare iniziale
	viene fissato a 0 dal costruttore
	*/
	public ContoCorrente(String n, String c, int num){
		// richiama l'altro costruttore
		this(n, c, num, 0);
	}

	/* crea un oggetto ContoCorrente con intestatario,
	numero di conto e ammontare iniziale specificati
	*/
	public ContoCorrente(String n, String c, int num, double amm){
		this.nome = n;
		this.cognome = c;
		this.numConto = num;
		this.saldo = amm;
	}

	/* aggiunge al conto corrente il valore specificato
	(che potrebbe anche essere negativo)
	*/
	public void aggiungi(double valore){
		this.saldo += valore;
	}

	/* restituisce true se l'ammontare corrente risulta inferiore
	allo scoperto massimo consentito, e restituisce false altrimenti
	*/
	public boolean isScoperto(){
		return (ContoCorrente.massimoScoperto + this.saldo < 0);
	}

	/* restituisce una descrizione completa del conto corrente
	sotto forma di stringa */
	public String toString(){
		String s = "";
		s += "Intestatario: " + this.nome + " " + this.cognome + "\n";
		s += "Numero conto: " + this.numConto + "\n";
		s += "Saldo: " + this.saldo + "\n";
		s += "Scoperto: " + this.isScoperto();
		return s;
	}

	/* reimposta il valore del massimo scoperto
	con il nuovo valore specificato */
	public static void setMassimoScoperto (double valore){
		ContoCorrente.massimoScoperto = valore;
	}
}



import fond.io.*;

public class TestContoCorrente{
	public static void main(String[] args){
		ContoCorrente.setMassimoScoperto(2000);

		// test creazione di un conto con saldo iniziale pari a 0
		InputWindow in = new InputWindow();
		String nome = in.readString("Nome intestatario?");
		String cognome = in.readString("Cognome intestatario?");
```

```
            int numero = in.readInt("Numero di conto corrente?");
            ContoCorrente c = new ContoCorrente(nome,cognome,numero);
            OutputWindow out = new OutputWindow();
            out.writeln("Informazioni sul conto creato:");
            out.writeln(c.toString());

            // test di modifica del saldo
            double valore = in.readDouble("Aggiungi cifra (mettere il segno -
                                                                per sottrarre");
            c.aggiungi(valore);
            out.writeln("Informazioni sul conto modificato:");
            out.writeln(c.toString());
        }
}
```