Pauta Certamen 2, Programación II

Prof. Rodrigo Olivares
Ayud. Juan Carlos Tapia
Mayo 26, 2016

Instrucciones:

- El puntaje máximo del certamen es 100 %, siendo el 60 % el mínimo requerido para aprobar.
- Responda cada pregunta en la hoja indicada, agregando su nombre. Si no responde alguna pregunta, debe entregar la hoja con su nombre e indicar que **no responde**.
- El certamen es **individual**. Cualquier intento de copia, será sancionado con nota **1,0**.

1. 30pts. De las siguentes afirmaciones, encierre en un círculo la o las alternativas correctas.

i. **Un constructor**:
(a) Siempre existe.
(b) Puede no ser incluirlo por el desarrollador.
(c) Al ser private sólo instancia objetos dentro de la clase.
(b) Al ser protected, no puede ser sobrecargado.
(e) Debe siempre tener el mismo nombre de la clase.

ii. **La clase Vector**:
(a) Es un List.
(b) Es un ArrayList.
(c) Es synchronized.
(d) Es no synchronized.
(e) Es un nodo.

iii. **Paso de parámetros**:
(a) Puede ser por valor.
(b) Puede ser por omisión.
(c) Puede ser por referencia.
(b) Puede ser por convensión.
(c) Puede ser por default.

iv. **La instancia this**:
(a) Invoca al constructor de una clase padre.
(b) Invoca al garbage collection.
(c) Referencia al construtor de la clase.
(d) Referencia a los atributos de la clase.
(e) Referencia a los método de la clase.

v. **Una clase abstracta**:
(a) Poseé sólo métodos abstractos.
(b) Alguno de sus métodos son abstractos.
(c) Instancia objetos abstractos.
(d) Permite extender una interfaz.
(e) Permite implementar una clase padre.

vi. **Respecto a las interfaces**:
(a) Su constructor es creado en compilación.
(b) Sus métodos definidos deben ser public.
(c) Sus métodos son abstract.
(d) Se implementa.
(e) Se extiende.

vii. **Un TDA Bean**.
(a) Sólo tiene atributos.
(b) Sólo tiene métodos.
(c) No debe incluir lógica.
(d) El constructor debe ser siempre incluido.
(e) Es posible agregar métodos como equals() y toString().

viii. **Sobre la herencia**:
(a) En java se realiza con la palabra reservada extends.
(b) En java se realiza con la palabra reservada impelements.
(c) Todas las clases heredan de la clase Object.
(d) La clase padre hereda el comportamiento de la clase hija.
(e) La clase hija hereda el comportamiento de la clase padre.

ix. **En relación a la manipulación de archivos**.
(a) Se lee un archivo con la instancia FileWriter.
(b) Se lee un archivo con la instancia FileReader.
(c) Se leen sólo archivos con delimitar y de largo fijo.
(d) StringTokenizer se usa para archivos con delimitador.
(e) Se requiere de la clase Scanner para la lectura.

x. **En relación a la manipulación de archivos**.
(a) Se escribe en un archivo con la instancia FileWriter.
(b) Se escribe en un archivo con la instancia FileReader.
(c) No es factible agregar contenido a un archivo existente.
(d) Es necesario utilizar la clase InputStreamReader.
(e) FileWriter("f.txt", false) agrega el contenido al final.

2. 70pts. Como Ingeniero Civil en Informática, la escuela le ha pedido que desarrolle un sistema que permita:

   a) Cargar las notas de los alumnos de dos asignaturas. Para ello, debe leer 3 dataset:
      - alumnos.csv: Identificar, Apellido Paterno, Apellido Materno, Nombre(s).
      - asignatura1.csv: Identificar del alumno, Nota Quiz 1, Nota Quiz 2, Nota Quiz 3, Nota Tarea 1, Nota Tarea 2, Nota Certamen 1, Nota Certamen 2 y Nota Certamen 3
      - asignatura2.csv: Identificar del alumno, Nota Quiz 1, Nota Quiz 2, Nota Quiz 3, Nota Tarea 1, Nota Tarea 2, Nota Certamen 1, Nota Certamen 2 y Nota Certamen 3

   b) Calcular el promedio de cada alumno, por asignatura, de la siguiente forma:

$$PQ = \frac{\sum_{i=1}^{3} NQ_i}{3} \qquad PT = \frac{\sum_{i=1}^{2} NT_i}{2} \qquad PC = \frac{\sum_{i=1}^{3} NC_i}{3}$$

   Para calcular el promedio final:

Pseudo-código 1 Promedio_Asignatura

si $PC \geqslant 4{,}0$ **entonces**
  retornar $PC * 0{,}7 + NQ * 0{,}15 + NT * 0{,}15$
si no, **si** $PC \geqslant 3{,}5$ **entonces**
  retornar $PC * 0{,}8 + NQ * 0{,}10 + NT * 0{,}10$
si no, **si** $PC \geqslant 3{,}0$ **entonces**
  retornar $PC * 0{,}9 + NQ * 0{,}05 + NT * 0{,}05$
si no
  retornar PC
fin si

   c) Por último, almacene en un cuarto archivo, denominado consolidado.csv -delimitado por caracter- la siguiente información: Id del alumno, Nota Final Asignatura 1, Nota Final Asignatura 2.

¿Cómo seré evaluado en la pregunta 2?			
Tópico	**Logrado**	**Medianamente logrado**	**No logrado**
Manipulación de archivo.	25pts Lee correctamente los archivos, los mapea a entidad y escribe en el archivo.	13pts Realiza dos de las tres acciones del punto anterior.	0pts No realiza la acciones del punto anterior.
TDA Lista.	15pts Crea la clase TDA Lista e implementa todos los métodos.	7pts Crea la clase TDA Lista e implementa algunos métodos.	0pts No crea la clase TDA Lista.
TDA Bean / Entidad	10pts Crea la clase entidad para alumno y asignatura.	5pts Crea la clase entidad para alumno o la asignatura (no ambas).	0pts No crea las clases entidad.
Clase principal y método main.	5pts Crea la clase principal en un archivo independiente con el método main.	3pts Crea el método main en la misma clase.	0pts No crea el método main.
Paradigma Orientación a Objetos	15pts Resuelve el problema utilizando el POO.	7pts Utiliza parte del POO para resolver el problema.	0pts No utiliza el POO para dar solución al problema.
Total máximo puntaje pregunta 2	70pts	35pts	0pts

```
public class Principal {

    public static void main(String[] args) {
        ListaAlumnoNotaImp lan = new ListaAlumnoNotaImp();
        lan.calcularPromedios();
    }
}

public class Alumno {

    private int id;
    private String apellidoPaterno;
    private String apellidoMaterno;
    private String nombres;

    public int getId() {
        return id;
    }

    public void setId(int id) {
        this.id = id;
    }

    public String getApellidoPaterno() {
        return apellidoPaterno;
    }

    public void setApellidoPaterno(String apellidoPaterno) {
        this.apellidoPaterno = apellidoPaterno;
    }

    public String getApellidoMaterno() {
        return apellidoMaterno;
    }

    public void setApellidoMaterno(String apellidoMaterno) {
        this.apellidoMaterno = apellidoMaterno;
    }

    public String getNombres() {
        return nombres;
    }

    public void setNombres(String nombres) {
        this.nombres = nombres;
    }
}

public class NotaAlumno {

    private int idAlumno;
    private float notaQuiz1;
    private float notaQuiz2;
    private float notaQuiz3;
```

```
private float notaTarea1;
private float notaTarea2;
private float notaCertamen1;
private float notaCertamen2;
private float notaCertamen3;

public int getIdAlumno() {
    return idAlumno;
}

public void setIdAlumno(int idAlumno) {
    this.idAlumno = idAlumno;
}

public float getNotaQuiz1() {
    return notaQuiz1;
}

public void setNotaQuiz1(float notaQuiz1) {
    this.notaQuiz1 = notaQuiz1;
}

public float getNotaQuiz2() {
    return notaQuiz2;
}

public void setNotaQuiz2(float notaQuiz2) {
    this.notaQuiz2 = notaQuiz2;
}

public float getNotaQuiz3() {
    return notaQuiz3;
}

public void setNotaQuiz3(float notaQuiz3) {
    this.notaQuiz3 = notaQuiz3;
}

public float getNotaTarea1() {
    return notaTarea1;
}

public void setNotaTarea1(float notaTarea1) {
    this.notaTarea1 = notaTarea1;
}

public float getNotaTarea2() {
    return notaTarea2;
}

public void setNotaTarea2(float notaTarea2) {
    this.notaTarea2 = notaTarea2;
}
```

```
    public float getNotaCertamen1() {
        return notaCertamen1;
    }

    public void setNotaCertamen1(float notaCertamen1) {
        this.notaCertamen1 = notaCertamen1;
    }

    public float getNotaCertamen2() {
        return notaCertamen2;
    }

    public void setNotaCertamen2(float notaCertamen2) {
        this.notaCertamen2 = notaCertamen2;
    }

    public float getNotaCertamen3() {
        return notaCertamen3;
    }

    public void setNotaCertamen3(float notaCertamen3) {
        this.notaCertamen3 = notaCertamen3;
    }

    public float getPromedioQuiz() {
        return (notaQuiz1 + notaQuiz2 + notaQuiz3) / 3;
    }

    public float getPromediTarea() {
        return (notaTarea1 + notaTarea2) / 2;
    }

    public float getPromedioCertamen() {
        return (notaCertamen1 + notaCertamen2 + notaCertamen3) / 3;
    }

    public float getPromedioFinal() {
        if (getPromedioCertamen() >= 4) {
            return (getPromedioCertamen() * 0.7f +
                   getPromedioQuiz() * 0.15f +
                   getPromediTarea() * 0.15f);
        } else if (getPromedioCertamen() >= 3.5) {
            return (getPromedioCertamen() * 0.8f +
                   getPromedioQuiz() * 0.1f +
                   getPromediTarea() * 0.1f);
        } else if (getPromedioCertamen() >= 3) {
            return (getPromedioCertamen() * 0.9f +
                   getPromedioQuiz() * 0.05f +
                   getPromediTarea() * 0.05f);
        } else {
            return getPromedioCertamen();
        }
    }
}
```

```
import java.util.ArrayList;
import java.util.List;
import java.util.StringTokenizer;

public class ListaAlumnoNotaImp {

    private List<NotaAlumno> notasAlumnosAsignatura1 = null;
    private List<NotaAlumno> notasAlumnosAsignatura2 = null;
    private List<Alumno> alumnos = null;
    private FuenteDeDatos fd = null;
    private List<String> lineas = null;
    private StringTokenizer stringTokenizer = null;
    private NotaAlumno notaAlumno = null;
    private Alumno alumno = null;

    public ListaAlumnoNotaImp() {
        notasAlumnosAsignatura1 = new ArrayList<NotaAlumno>();
        notasAlumnosAsignatura2 = new ArrayList<NotaAlumno>();
        alumnos = new ArrayList<Alumno>();
        fd = new FuenteDeDatos();
        cargarAlumnos();
        cargarNotasAlumnos("asignatura1.csv", notasAlumnosAsignatura1);
        cargarNotasAlumnos("asignatura2.csv", notasAlumnosAsignatura2);
    }

    private void cargarAlumnos() {
        lineas = fd.leerArchivo("alumnos.csv");
        for (String linea : lineas) {
            stringTokenizer = new StringTokenizer(linea, ";");
            if (stringTokenizer.hasMoreElements()) {
                alumno = new Alumno();
                alumno.setId(Integer.parseInt(stringTokenizer.nextToken()));
                alumno.setApellidoPaterno(stringTokenizer.nextToken());
                alumno.setApellidoMaterno(stringTokenizer.nextToken());
                alumno.setNombres(stringTokenizer.nextToken());
                alumnos.add(alumno);
            }
        }
    }

    private void cargarNotasAlumnos(String nombreArchivo, List<NotaAlumno> lista) {
        lineas = fd.leerArchivo(nombreArchivo);
        for (String linea : lineas) {
            stringTokenizer = new StringTokenizer(linea, ";");
            if (stringTokenizer.hasMoreElements()) {
                notaAlumno = new NotaAlumno();
                notaAlumno.setIdAlumno(Integer.parseInt(stringTokenizer.nextToken()));
                notaAlumno.setNotaQuiz1(Float.parseFloat(stringTokenizer.nextToken().replace(",", ".")))
                notaAlumno.setNotaQuiz2(Float.parseFloat(stringTokenizer.nextToken().replace(",", ".")))
                notaAlumno.setNotaQuiz3(Float.parseFloat(stringTokenizer.nextToken().replace(",", ".")))
                notaAlumno.setNotaTarea1(Float.parseFloat(stringTokenizer.nextToken().replace(",", ".")
                notaAlumno.setNotaTarea2(Float.parseFloat(stringTokenizer.nextToken().replace(",", ".")
                notaAlumno.setNotaCertamen1(Float.parseFloat(stringTokenizer.nextToken().replace(",", "
```

```
                notaAlumno.setNotaCertamen2(Float.parseFloat(stringTokenizer.nextToken().replace(",", "
                notaAlumno.setNotaCertamen3(Float.parseFloat(stringTokenizer.nextToken().replace(",", "
                lista.add(notaAlumno);
            }
        }
    }

    public void calcularPromedios() {
        lineas = new ArrayList<String>();
        String linea;
        for (Alumno alumnoAux : alumnos) {
            linea = alumnoAux.getId() + ";";
            for (NotaAlumno notaAlumnoAux : notasAlumnosAsignatura1) {
                if (notaAlumnoAux.getIdAlumno() == alumnoAux.getId()) {
                    linea += String.format("%f", notaAlumnoAux.getPromedioFinal()) + ";";
                    break;
                }
            }
            for (NotaAlumno notaAlumnoAux : notasAlumnosAsignatura2) {
                if (notaAlumnoAux.getIdAlumno() == alumnoAux.getId()) {
                    linea += String.format("%f", notaAlumnoAux.getPromedioFinal());
                    break;
                }
            }
            lineas.add(linea);
        }
        fd.escribirArchivo("consolidado.csv", lineas);
    }
}

import java.io.BufferedReader;
import java.io.File;
import java.io.FileReader;
import java.io.FileWriter;
import java.io.IOException;
import java.io.PrintWriter;
import java.util.ArrayList;
import java.util.List;

public class FuenteDeDatos {

    public List<String> leerArchivo(String nombreArchivo) {
        File archivo;
        FileReader fileReader = null;
        List<String> lineas = null;
        BufferedReader bufferedReader;

        try {
            archivo = new File(nombreArchivo);
            lineas = new ArrayList<String>();
            String linea;
            fileReader = new FileReader(archivo);
            bufferedReader = new BufferedReader(fileReader);
            while ((linea = bufferedReader.readLine()) != null) {
```

```
                lineas.add(linea);
            }
        } catch (IOException e) {
            System.out.println(e);
        } finally {
            try {
                if (fileReader != null) {
                    fileReader.close();
                }
            } catch (IOException e) {
                System.out.println(e);
            }
        }
        return lineas;
    }

    public void escribirArchivo(String nombreArchivo, List<String> lineas) {
        FileWriter archivo;
        PrintWriter printWriter = null;
        try {
            archivo = new FileWriter(nombreArchivo);
            printWriter = new PrintWriter(archivo);
            for (String linea : lineas) {
                printWriter.println(linea);
            }
        } catch (IOException e) {
            System.out.println(e);
        } finally {
            printWriter.close();
        }
    }
}
```