Examen Parcial

8 d'Abril de 2015

Nom i Cognoms: ..... NIUB: .....

Contesta els següents exercicis implementant en Java el que es demana dins d'un projecte de Netbeans. Per evitar introduir errors a mesura que fas els canvis demanats als diferents exercicis, anomena al primer projecte Parcial1Exercici1 i fes una copia del projecte per cada exercici creant els projectes: Parcial1Exercici2, Parcial1Exercici3. Quan acabis, fes el lliurament a la tasca oberta del Campus Virtual amb un sol fitxer comprimit anomenat: Parcial1_Cognom1Cognom2Nom.

Enunciat:

A un vídeo club es desitja desenvolupar un programa que permeti gestionar el negoci, que consisteix en el préstec de pel·lícules i jocs de consola. Hi ha dos tipus de préstecs: els préstecs "llargs", que permeten prestar els productes diversos dies, i els préstecs "exprés", que permeten prestar els productes per hores. Per tots els productes s'ha de poder guardar el tipus de préstec que s'ha realitzat "llarg" o "exprés" i si el producte està disponible o no, ja que pot haver diversos motius per voler no prestar el producte. A més, el preu del préstec variarà segons el tipus de producte. En la següent taula teniu un resum dels preus per cada producte i cada tipus de préstec:

	Preu per dia (Préstec Llarg)	**Preu per hora (Préstec Exprés)**
Pel·lícula	5 €	0.50 €
Joc	7 €	0.75 €

Cada producte tindrà un identificador de producte i a més, guardarà la informació necessària per tal de poder mostrar per pantalla les seves característiques:

- En el cas de les pel·lícules, guardarem el títol de la pel·lícula, l'any en que es va rodar, el director i els actors.
- En el cas dels jocs, guardarem el nom del joc, l'any en que es va editar i l'empresa propietària.

Seguint aquest enunciat, implementa en llenguatge Java el que es demana en els següents exercicis:

Exercici 1.A (4 punts)
Implementa una interfície **Prestable**, una classe abstracta **ProductePrestable**, una classe **Pelicula** i una classe **Joc**.

A la interfície **Prestable** es defineixen els següents mètodes:

- *iniciarPrestec*: Iniciarà el préstec del producte. Només requereix la informació del tipus de préstec.
- *finalitzarPrestec*: Finalitzarà el préstec d'un producte i retornarà el cost del préstec.
- *getEstatDisponible*: Consultarà l'estat del producte, si està disponible per ser prestat o no.
- *setEstatDisponible*: Permetrà canviar l'estat per que estigui disponible o no per prestar.
- *getPreuPrestecLlarg:* Retornarà el preu per dia per aquest producte.
- *getPreuPrestecExpress:* Retornarà el preu per hora per aquest producte.

La classe abstracta **ProductePrestable** implementarà la interfície **Prestable.** A més, en aquesta classe has de:

- Guardar una variable per indicar si el producte està disponible per préstec o no i el tipus de préstec actual ("llarg" o "exprés").
- Guardar la data en que s'ha llogat (null en cas que no estigui llogat).
- Implementar un sol constructor on s'inicialitzen tots els atributs de la classe amb els valors passats per paràmetre, tenint en compte que els atributs genèrics dels productes han d'estar en aquesta classe.
- Implementar els següents mètodes:
  - *iniciarPrestec*: consistirà en activar la variable que indica que el producte no està disponible i guardar la data actual, per indicar l'inici del préstec.
  - *finalitzarPrestec*: calcularà el cost del préstec com el producte del número de dies pel preu per dies o el número d'hores pel preu per hores, segons si el tipus de préstec és llarg o exprés, i a partir de la data inicial i final. A més, posarà la data de lloguer a null i desactivarà la variable que indica que el producte no està en préstec.
  - *getEstatDisponible:* tornarà l'estat del producte, si està disponible per prestar o no.
  - *setEstatDisponible:* permetrà canviar l'estat per que estigui disponible per prestar o no.

- Mètodes consultors i modificadors necessaris pels atributs de la classe.
- Crear un mètode *mostrarDisponibilitat*, per mostrar si el producte està disponible o no. En concret, mostrarà per pantalla la frase "Producte disponible" o "Producte no disponible" depenent de si el producte està o no disponible.
- Crear un mètode abstracte *mostrarInfoProducte*, per mostrar la informació del producte.

A les classes **Pelicula** i **Joc** has d'incloure els atributs (que s'han de deduir de l'enunciat) i implementar els següents mètodes:

- Un sol constructor on s'inicialitzen tots els atributs de la classe
- El mètode *mostrarInfoProducte*:
  1. A la classe **Pelicula**, aquest mètode mostrarà per pantalla la frase "Aquesta pel·lícula és" seguida pel títol de la pel·lícula, l'any en que es va rodar, el director i els actors. A més, ha de mostrar el missatge "Producte disponible" o "Producte no disponible" depenent de si el producte està o no disponible.
  2. A la classe **Joc**, aquest mètode mostrarà per pantalla la frase "Aquest joc és" seguida del nom del joc, l'any en que es va editar i l'empresa propietària. A més, ha de mostrar el missatge "Producte disponible" o "Producte no disponible" depenent de si el producte està o no disponible.

- Mètodes consultors i modificadors necessaris pels atributs de les classes.

Nota1: Utilitza la classe **Data** implementada en Data.java i disponible al Campus Virtual.
Nota2: Per les pel·lícules, els actors es guarden en un text, separats per una coma.

Solució:

Prestable.java

```
package examenparcial1;

public interface Prestable {
   public void iniciarPrestec(boolean prestecLlarg);// Iniciarà el préstec del producte. Només requereix la informació del tipus de préstec.
   public double finalitzarPrestec(); //: Finalitza el préstec d'un producte i retorna el cost del préstec.
   public boolean getEstatDisponible();//Consultarà l'estat del producte, si està disponible per ser prestat o no.
   public void setEstatDisponible(boolean b) // Permetrà canviar l'estat per que estigui disponible o no per prestar.
   public double getPreuPrestecLlarg(); // Retornarà el preu per dia per aquest producte.
```

```
    public double getPreuPrestecExpress(); // Retornarà el preu per hora per aquest producte.
}
```

ProductePrestable.java

```
package examenparcial1;

public abstract class ProductePrestable implements Prestable {
    protected boolean disponible;
    protected boolean prestecLlarg;
    protected int idProducte;
    private Data dataIniciPrestec;

    public ProductePrestable(int idProducte){
        this.disponible = true;
        this.prestecLlarg = true;
        this.dataIniciPrestec = null;
        this.idProducte = idProducte;
    }

    @Override
    public void iniciarPrestec(boolean prestecLlarg){
        // Iniciarà el préstec del producte. Només requereix la informació del tipus de préstec.
        dataIniciPrestec = Data.getData();
        this.disponible=false;
        this.prestecLlarg = prestecLlarg;
    }

    @Override
    public double finalitzarPrestec(){
        //: Finalitza el préstec d'un producte i retorna el cost del préstec.
        double preu = 0;



        if(prestecLlarg){
            long dies=Data.getData().getDiffDays(dataIniciPrestec);
            preu = dies * getPreuPrestecLlarg();
        }else{
            long hores=Data.getData().getDiffHours(dataIniciPrestec);;
            preu =hores*getPreuPrestecExpress();
        }

        disponible=true;
        dataIniciPrestec = null;
        return preu;
    }

    @Override
    public boolean getEstatDisponible(){//Consultarà l'estat del producte, si està disponible per
ser prestat o no.
        return disponible;
    }

    @Override
    public void setEstatDisponible(boolean disponible){
        // Permetrà canviar l'estat per que estigui disponible o no per prestar.
        this.disponible = disponible;
    }
```

```
    // Getters i setters....

    public int getIdProducte(){
        return this.idProducte;
    }
    public void setIdProducte(int idProducte){
        this.idProducte = idProducte;
    }


    public abstract void mostrarInfoProducte();

    public void mostrarDisponibilitat(){
        if(disponible){
            System.out.println("Producte Disponible");

        }else{
             System.out.println("Producte No Disponible");
        }
    }

}
```

Pelicula.java

```
package examenparcial1;

public class Pelicula  extends ProductePrestable {
    private static final double preuPerDia = 5;
    private static final double preuPerHora = 0.5;
    private String titol;
    private int any;
    private String director;
    private String actors;

    public Pelicula(int idProducte, String titol, int any, String director, String actors) {
        super(idProducte);
        this.titol = titol;
        this.any = any;
        this.director = director;
        this.actors = actors;
    }

    @Override
   public void mostrarInfoProducte() {
       System.out.println("Aquesta pel·lícula és: " + titol + " " + Integer.toString(any) + " " + director
+ actors);
       mostrarDisponibilitat();
    }

    @Override
    public double getPreuPrestecLlarg() {
        return preuPerDia;
    }

    @Override
    public double getPreuPrestecExpress() {
        return preuPerHora;
```

```
    }

    // getters i setters...

}
```

Joc.java

```
package examenparcial1;

public class Joc  extends ProductePrestable {
    private static final double preuPerDia = 7;
    private static final double preuPerHora = 0.75;
    private String nom;
    private int any;
    private String empresa;

    public Joc(int idProducte , String nom, int any, String empresa) {
        super(idProducte);
        this.nom = nom;
        this.any = any;
        this. empresa = empresa;
    }

    @Override
    public void mostrarInfoProducte() {
        System.out.println("Aquesta joc és: " + nom + " " + Integer.toString(any) + " " + empresa);
        mostrarDisponibilitat();
    }

     @Override
    public double getPreuPrestecLlarg() {
        return preuPerDia;
    }

    @Override
    public double getPreuPrestecExpress() {
        return preuPerHora;
    }
    // getters i setters...
}
```

Exercici 1.B (1 punt)

Implementa una altra classe **GestorPrestecs1** amb un mètode main on es creï una llista de 3 productes (fent servir la classe ArrayList de Java) i s'ompli amb les següents instàncies:

1. La pel·lícula, amb identificador 1, titulada "Shrek" del 2001 dirigida per Andrew Adamson i sense actors
2. La pel·lícula, amb identificador 2, titulada "Blade Runner" del 1982 dirigida per Ridley Scott i protagonitzada per l'actor Harrison Ford.
3. El joc, "Mario Bros", amb identificador 3, del 1983 i propietat de Nintendo.

Un cop introduïdes les instàncies, afegeix al mètode main les següents accions:

- El préstec de la pel·lícula Blade Runner i el préstec del joc Mario Bros. Els dos préstecs són de tipus Express.
- Recorre aquesta llista fent servir un iterador i invoca el mètode mostrarInfoProducte per a cada element.

Solució:
Ha de mostrar per pantalla:

```
Aquesta pel·lícula és: Shrek 2001 Andrew Adamson
Producte Disponible
Aquesta pel·lícula és: Blade Runner 1982 Ridley ScottHarrison Ford
Producte No Disponible
Aquesta joc és: Mario Bros 1983 Nintendo
Producte No Disponible
```

GestorPrestecs1.java

```
package examenparcial1;

import java.util.ArrayList;
import java.util.Iterator;

public class GestorPrestecs1 {
     /**
     * @param args the command line arguments
     */
    public static void main(String[] args) {

        ArrayList< ProductePrestable > llistaProds = new ArrayList<ProductePrestable>();
        Pelicula peli1 = new Pelicula(1, "Shrek", 2001 , "Andrew Adamson", "");
        llistaProds.add(peli1);
        Pelicula peli2 = new Pelicula (2, "Blade Runner", 1982, "Ridley Scott", "Harrison Ford");
        llistaProds.add(peli2);
        Joc joc1 = new Joc(3, "Mario Bros", 1983, "Nintendo");
        llistaProds.add(joc1);

        boolean prestecLlarg = false;
        peli2.iniciarPrestec(prestecLlarg);
        joc1.iniciarPrestec(prestecLlarg);

        for(Iterator<ProductePrestable> i = llistaProds.iterator();i.hasNext();){
            i.next().mostrarInfoProducte();
        }

    } // fin de main()
}
```

Exercici 1.C (1 punt)

En un full a part, contesta a les següents preguntes sobre el codi implementat:

a) Es fa servir el polimorfisme en el mètode main? En cas afirmatiu, explica on exactament i com s'observa.
b) Explica com has implementat *getPreuPrestecExpress* i per què.

Solució:

a) Sí, es fa servir el polimorfisme en el mètode main quan es crida al mètode mostrarInfoProducte. S'observa al cridar al mètode mostrarInfoProducte, el qual pren una forma diferent segons si l'objecte que està contingut a la posició corresponent de la llista és una Pelicula o un Joc.

b) Es fa servir una constant pròpia per cada classe Pelicula o Joc i es retorna la constant corresponent en cada mètode.
private static final double preuPerDia = 5;

Exercici 2 (1.5 punts)
Implementa una classe **GestorPrestecs2** amb un mètode main on:

- Has de crear una llista de 5 productes qualsevols. Demana a l'usuari el tipus de productes que vol introduir a la llista: si introdueix 0, afegeixes una pel·lícula amb un String buit en cada camp, si introdueix un 1, afegeixes un joc amb un String buit en cada camp.
- Fes un recorregut per modificar el títol de totes les pel·lícules que hi ha en la llista (només les pel·lícules, no els jocs). El títol es canviarà per aquest altre títol: "Títol Desconegut".
- Imprimeix per pantalla el títol de les pel·lícules que es troben a la llista abans i després del canvi.

Solució:
Han d'utilitzar un instanceof.

GestorPrestecs2.java

```
package examenparcial1;

import java.util.ArrayList;
import java.util.Iterator;
import java.util.Scanner;

public class GestorPrestecs2 {

   public static void main(String[] args) {
      Scanner sc = new Scanner(System.in);
      ArrayList< ProductePrestable > llistaProds = new ArrayList<ProductePrestable>();



      boolean prestecLlarg = false;
      for(int i=1;i<=5;i++){
         System.out.println("Introdueix un 0 per Pel·lícula i 1 per Joc");
         int op = sc.nextInt();
         if(op==0){
            prod = new Pelicula(i, "", 2000 , "", "");
         }else{
            prod = new Joc(i, "", 2000, "");
         }
         prod.iniciarPrestec(prestecLlarg);
         llistaProds.add(prod);
      }

      ProductePrestable prod;
      for(Iterator<ProductePrestable> i = llistaProds.iterator();i.hasNext();){
        prod = i.next();
        if( prod instanceof Pelicula){
           Pelicula peli = (Pelicula) prod;
           System.out.println(peli.getTitol());
           peli.setTitol("Títol Desconegut");
           System.out.println(peli.getTitol());
        }
      }
   } // fin de main()
}
```

Exercici 3 (1.5 punts)
Amb el mètode *setEstatDisponible* es pot modificar l'estat del producte entre disponible o no disponible per bloquejar un cert producte. Abans de fer el canvi a disponible s'hauria de fer una comprovació de si el producte està en préstec o no, ja que si és el cas no s'hauria de permetre activar l'etiqueta disponible i s'hauria d'avisar a l'usuari.
Per tal de fer aquest control:

- Crea una excepció pròpia **ExceptionProductePrestable** amb el següent missatge per defecte: "Exception del Producte Prestable".
- Implementa, en el mètode *setEstatDisponible*, la comprovació de si el producte està en préstec o no, de manera que si està en préstec no es podrà posar com a disponible i es llançarà una excepció per mostrar el següent missatge d'avís a l'usuari: "No pots assignar l'etiqueta disponible a un producte prestat".
- Per revisar que el funcionament d'aquest control d'errors és correcte, implementa una classe **GestorPrestecs3** amb un mètode main que contindrà la gestió de l'excepció i on has de fer el següent:
    - Crea una instància de **Pelicula**, amb identificador 1, titulada "Shrek" del 2001 dirigida per Andrew Adamson i sense actors
    - Inicia el préstec de la pel·lícula
    - Canvia el seu estat a disponible

Solució:

S'ha de crear la classe:
ExceptionProductePrestable.java

```
package examenparcial1;

public class ExceptionProductePrestable extends Exception{
   public static String MyMessage; // Constant

   public ExceptionProductePrestable(){
      super();
      MyMessage="Exception del Producte Prestable";
   }
   public ExceptionProductePrestable(String msg){
      super(msg);
   }
}
```

Ha de sortir per pantalla:
"No pots assignar disponible a un producte prestat"

Han de canviar la Interfície Prestable afegint:
Prestable.java

```
public void setEstatDisponible(boolean b) throws ExceptionProductePrestable;
```

Han de canviar el mètode setEstatDisponible de la classe ProductePrestable:
ProductePrestable.java

```
public void setEstatDisponible(boolean disponible) throws ExceptionProductePrestable{
      // Permetrà canviar l'estat per que estigui disponible o no per prestar.

      if(disponible == true && this.dataIniciPrestec==null){
         this.disponible = disponible;
      }else{
         throw new ExceptionProductePrestable("No pots assignar disponible a un producte
prestat");
      }
 }
```

GestorPrestecs3.java

```
package examenparcial1;

public class GestorPrestecs3 {

   public static void main(String[] args) {
      Pelicula prod = new Pelicula(2, "Shrek", 2001 , "Andrew Adamson", "");

      boolean prestecLlarg = false;
      prod.iniciarPrestec(prestecLlarg);

      try {
         prod.setEstatDisponible(true);
      } catch (ExceptionProductePrestable ex) {
         System.out.println(ex.getMessage());
      }
   } // fin de main()
}
```

Exercici 4 (1 punts)
Indiqueu si són certes o falses les següents afirmacions i justifiqueu les respostes als espais:

a) En la sobrecàrrega d'un mètode no podem canviar el tipus dels paràmetres.

Solució: Fals. Quan sobrecarreguem un mètode hem de canviar la llista de paràmetres: el número o el seu tipus.

b) Donada A una classe i B una subclasse de A. La següent assignació és una conversió de tipus implícita:
A a = new B();

Solució: Cert, una conversió implícita es pot fer així A a = new B().
Ja que el tipus de la variable (tipus estàtic) i tipus dinàmic és diferent.

c) No cal que existeixi cap objecte per a poder aplicar un mètode de classe.

Solució: Cert. Un mètode de classe és un mètode que és el mateix per a tota una classe d'objectes i no depèn de les dades concretes de cap instància. No cal que existeixi cap objecte per a poder-lo aplicar.

d) El tipus estàtic d'una variable es determina en temps d'execució.

Solució: Fals. El tipus estàtic d'una variable es determina en temps de compilació.