Programació I - **Examen** - 9 de gener del 2015

Nom i Cognoms: .....
Dni: .....

ENUNCIAT: Feu un programa que permeti fer la gestió d'una galeria d'art.

A l'etapa de disseny s'ha decidit crear tres classes: **`Cuadro`** (per a representar la informació d'un quadre), **`Galeria`** (per a guardar el conjunt de tots els quadres de la galeria, com màxim es tindran 100 quadres, valor definit a la constant MAX_CUADROS) i **`Gestion`** (per a implementar el control principal de l'aplicació). Al final d'aquest document s'adjunta un llistat per a veure part de la implementació de la classe Gestion.

Per a cada quadre de la galería es vol guardar l'autor, el títol, l'amplada, l'alçada, l'estil (SIN_CLASIFICAR, IMPRESIONISTA, CLASICO, REALISMO) i el preu. Per defecte, un quadre té l'estil SIN_CLASIFICAR i l'amplada i l'alçada estandard són 100cm i 80cm respectivament. La galeria conté un conjunt de quadres i, com que es poden vendre quadres es vol guardar informació dels diners obtinguts en aquestes vendes. Inicialment, el saldo de la galeria és 0 euros.

El menú de l'aplicació ofereix les següents opcions:

1. Afegir quadre a la galeria.
2. Mostrar els quadres que actualment estan en la galeria.
3. Vendre quadre de la galeria.
4. Seleccionar quadres per a una exposició itinerant.
5. Sortir

PROBLEMES A RESOLDRE I ENTREGAR: cal entregar CADA PROBLEMA EN UN FULL PER SEPARAT i posar el Nom, Cognoms i DNI en tots els fulls.

NOTA IMPORTANT: És obligatori caracteritzar totes les seqüencies que utilitzis als següents exercicis.

1. (*2 punts*)

   (a) Defineix la classe **Cuadro**, justificant els atributs que caracteritzen aquesta classe i els seus tipus.
   SOLUC:

```
public class Cuadro {
private String autor;
private String titulo;
private float [] medida; //También pueden ser 2 atributos float o double
private int estilo;
private double precio;
..
}
```

   (b) Implementa només el següents mètodes:

   - 2 constructors de la classe, un sense paràmetres i un altre amb paràmetres que permeti passar les dades necessàries per a inicialitzar els atributs.
     SOLUC

```
  public Cuadro(String autor, String titulo, float [] medida, int esti
this.autor = autor;
this.titulo = titulo;
this.medida = new float[2];
this.medida[0] = medida[0]; //ancho medida estandard
this.medida[1] = medida[1]; //alto
this.estilo = estilo;
```

```
    this.precio = precio;
  }

  public Cuadro() {
    this.medida = new float[2];
    this.medida[0] = 100.0f; //ancho y ancho medida estandard
    this.medida[1] = 80.0f;
    this.estilo = Gestion.SIN_CLASIFICAR;
  }
```

- Un mètode que permeti realitzar crides d'aquest tipus:

```
Cuadro cuadro;
...
System.out.println(cuadro);
...
```

SOLUC

```
public String toString() {
String s = "";
s = s + "\n␣Autor:␣" + autor + "\n␣Titulo:␣" + titulo
+ "␣\n␣Ancho:"+ medida[0]+"\n␣Alto:␣" + medida[1] + "\n␣Estil:␣" + estilo
"\n␣Precio:␣" + precio + "\n";
return s;
}
```

- Un mètode `set` i un altre `get` de la classe Cuadro.
  SOLUC:

```
 public void setAutor(String autor){
    this.autor = autor;
  }
 public String getAutor(){
    return autor;
  }
```

NOTA: No és necessari implementar TOTS els mètodes `set` i `get` de la classe **Cuadro**, es suposen implementats, però al llarg de tot el codi pots utilitzar-los com si estiguessin implementats.

(c) Explica breument en què consisteix el principi d'encapsulació. Has aplicat aquest principi en la definició de la classe Cuadro? Raona la teva resposta.

SOLUC: Cuando aplicamos este principio facilitamos la independencia de la implementación por parte del usuario, ocultamos (usando private) al usuario de la clase detalles internos y le proporcionamos una interfície pública (métodos public). La encapsulación proporciona también robustez al código. Si lo he aplicado, tanto en la definición private de los atributos como al definir funciones publicas que facilitan el acceso a los mismos (get set).

2. (*1,5 punts*)

(a) Defineix la classe **Galeria**, justificant els atributs que caracteritzen aquesta classe i els seus tipus.
SOLUC:

```
 public class Galeria{
private Cuadro [] cuadros;
private int numCuadros;
private double saldo;
}
```

(b) Completa el mètode `opcionAnyadir(...)` amb les declaracions i sentències que calen pel bon funcionament del programa:

```
static void opcionAnyadir(Scanner sc, Galeria galeria) {
    .....
    galeria.anyadirCuadro(cuadro);
    ....
}
```

SOLUC:

```
 static void opcionAnyadir(Scanner sc, Galeria galeria) {
Cuadro cuadro;
String autor, titulo;
float [] medida = new float[2];
int estilo; double precio;

System.out.println("Introduce␣los␣datos␣del␣cuadro");
System.out.println("Autor:");
autor = sc.next();
System.out.println("Título:");
titulo = sc.next();
System.out.println("Ancho:");
medida[0] = sc.nextFloat();
System.out.println("Alto:");
medida[1] = sc.nextFloat();
System.out.println("Estilo:␣(0\-SIN\_CLASIFICAR,␣1\-IMPRESIONISTA,␣2\-CLA
estilo = sc.nextInt();
System.out.println("Precio:");
precio = sc.nextFloat();
cuadro = new Cuadro(autor, titulo, medida, estilo, precio);
galeria.anyadirCuadro(cuadro);

}
```

A quina/es classe/s pertanyen els mètodes opcionAnyadir() i anyadirCuadro(). Són mètodes de classe o d'objecte? Per què?

SOLUC: A Gestion y Galeria respectivamente. opcionAnyadir() es un método de clase porque es el mismo para toda una clase de objetos y no depende de los datos concretos de ningún objeto, no hace falta un objeto para llamar al método. anyadirCuadro() es un método de objeto porque depende de los datos concretos de una instancia (objeto), necesitamos que exista un objeto para llamar al método.

(c) Implementa el mètode **anyadirCuadro**. Aquest mètode afegeix un nou quadre a la galeria d'art.

SOLUC:

```
public void anyadirCuadro(Cuadro p) {

if (numCuadros < cuadros.length){
  cuadros[numCuadros] = p;
  numCuadros++;
}
else{
  System.out.println("Has␣llegado␣al␣límite␣de␣cuadros");
}

}
```

NOTA: No és necessari implementar els mètodes `set` i `get` de la classe **Galeria**, es suposen implementats, però al llarg de tot el codi pots utilitzar-los com si estiguessin implementats.

3. (*3 punts*)

(a) Defineix el mètode **venderCuadro** de classe **Galeria**, que elimina un quadre de la galeria (el conjunt de quadres ha de quedar sense espais buits entre ells), actualitzant convenientment els atributs afectats per aquesta eliminació. Aquest mètode té com paràmetre el títol del quadre a vendre i retorna l'enter 1 indicant si s'ha fet correctament l'operació, en cas contrari retorna -1.

```
public int venderCuadro(String titulo) {
.....
}
```

SOLUC:

```
public int venderCuadro(String titulo) {
int index, i;

  index = buscarCuadro(titulo);
  if (index !=-1){
      saldo += cuadros[index].getPrecio();
      /* Identificacion de la secuencia: posiciones en el array a partir
       *
       * Primer: i=index;
       * Seguent(i): i++;
       * FinSequ(i): i >= numCuadros -1
       * Esquema a aplicar: recorregut
       */
      i = index;
      while (i < numCuadros-1){
          cuadros[i]=cuadros[i+1];
          i++;
      }
      numCuadros--;
      return 1;
  }
  else{
      return -1;
  }

}

 public int buscarCuadro(String titulo) {
   int i = 0;
   boolean trobat = false;
     /* Identificacion de la secuencia: posiciones en el array a partir de
      *
      * Primer: i=0;
      * Seguent(i): i++;
      * FinSequ(i): i >= numCuadros
      * Esquema a aplicar: cerca Condición: cuadros[i].getTitulo().equal
      */

    while (i < numCuadros && ! trobat){
      if (cuadros[i].getTitulo().equals(titulo)){
        trobat = true;
      }
      else{
        i++;
      }
    }
    if (trobat)
```

```
        return i;
      else
        return -1;
  }
```

(b) Implementa el mètode **opcionVenderCuadro** de la classe **Gestio**. Aquest mètode, donat el títol del quadre (introduït per teclat pel usuari), realitza la venda del quadre indicat.

```
public void opcionVenderCuadro(Scanner sc, Galeria galeria) {
.....
}
```

SOLUC:

```
            static void opcionVenderCuadro(Scanner sc, Galeria galeria) {
  String titulo;

  System.out.println("Introduce␣el␣titulo␣del␣cuadro:");
  titulo = sc.next();

  if (galeria.venderCuadro(titulo) == 1){
    System.out.println("Cuadro␣" + titulo +  "␣vendido");
    System.out.println("Saldo␣actual␣de␣la␣galeria:␣" + galeria.getSaldo())
  }
  else{
    System.out.println("No␣se␣ha␣podido␣vender␣porque␣no␣existe␣el␣cuadro␣d
  }
}
```

4. (*3,5 punts*)

(a) A vegades es fan exposicions fora de la galeria. Implementa el mètode **CuadrosExposicion** de la classe Galeria per a què retorni els quadres d'estil IMPRESIONISTA que es poden enviar a l'exposició, tenint en compte que es vol ocupar només el 50% de l'area de la paret d'exposició. El mètode cuadrosExposicion rep el paràmetre areaExpo (l'area de la paret on es van a exposar els quadres) i retorna els quadres a exposar i un enter que indica el nombre de quadres que s'enviaran a l'exposició. El nombre màxim de quadres (MAX_EXPO) que es poden enviar a l'exposició és 10, aquest valor està definit com una constant a la classe Gestió. Recorda que l'àrea d'un rectangle es calcula com el producte de la amplada per la alçada.

```
public int cuadrosExposicion(float areaExpo, Cuadro[ ] cuadros) {
.....
}
```

SOLUC:

```
 int cuadrosExposicion(float areaExpo, Cuadro[] cuadros){
int i=0; int nExp=0; double area;
double sumArea = 0.0; boolean trobat;

        /* Identificacion de la secuencia: indices del array de cuadros
         * que hay a partir del vendido
         * Primer: i=0 nExp=0;
         * Seguent(i): i++ nExp++
         * FinSequ(i): i >= numCuadros
         * Esquema a aplicar: Cerca
         * Condición cerca: SumArea > areaExpo/2 || nExp >= Gestion.MAX_EXP
         */
  trobat = false;
```

```
    while (i < numCuadros && !trobat){

        if (this.cuadros[i].getEstilo() == Gestion.IMPRESIONISTA){
            sumArea += this.cuadros[i].getArea();
            if (sumArea > areaExpo/2 || nExp >= Gestion.MAX_EXPO ){
              trobat = true;
            }
            else{
             cuadros[nExp]=this.cuadros[i];
             nExp++;
             i++;
            }
        }
        else{
         i++;
        }

        }
    return nExp;
}
```

(b) Quin tipus de paràmetre són `areaExpo` i `cuadros`: d'entrada, de sortida o d'entrada i sortida? Enraona la teva resposta.

SOLUC: areaExpo de entrada porque proporciona información de entrada al método, cuadros es de salida porque se devuelve información a través de ese parámetro.

(c) Implementa el mètode `opcionExposicion` de la classe **Gestio**. Aquest mètode donades les dimensions (amplada i alçada) de la paret on es farà l'exposició (introduïdes per teclat pel usuari) selecciona el conjunt de quadres de la Galeria que es poden enviar a la exposició i els mostra per pantalla.

```
        public void opcionExposicion(Scanner sc, Galeria galeria) {
        .....
        }
```

SOLUC:

```
                static void opcionExposicion(Scanner sc, Galeria galeria) {
     float areaExpo;
     Cuadro [] cuadros = new Cuadro[MAX_EXPO];
     int nCuadros;

     System.out.println("Introduzca␣el␣ancho␣y␣alto␣de␣la␣pared␣de␣exposición:
     areaExpo = sc.nextFloat() * sc.nextFloat();

     nCuadros = galeria.cuadrosExposicion(areaExpo, cuadros);

     System.out.println("Los␣cuadros␣impresionistas␣que␣se␣enviaran␣a␣la␣expos

     for (int i=0; i < nCuadros; i++){
       System.out.println(cuadros[i]);
     }
    }
```

Llistat de la classe principal de l'aplicació, amb els mètodes que ja teniu implementats: (podeu observar quins són els paràmetres d'entrada i els valors de sortida dels mètodes que heu d'implementar als problemes)

```
public class Gestion{
  public  static final int ANYADIR_CUADRO    = 1;
  public  static final int MOSTRAR_CUADROS   = 2;
  public  static final int VENDER_CUADRO     = 3;
  public  static final int EXPOSICION        = 4;
  public  static final int SALIR             = 5;

  public  static final int MAX_CUADROS      = 100;
  public  static final int MAX_EXPO         = 10;


  public  static final int SIN_CLASIFICAR = 0;
  public  static final int IMPRESIONISTA  = 1;
  public  static final int CLASICO        = 2;
  public  static final int REALISMO       = 3;


  public static void main (String [] args) {
    Scanner sc;
    Galeria galeria = new Galeria(MAX_CUADROS);
    int     op;

    sc = new Scanner(System.in);
    op = Menu(sc);
    while (op!= SALIR) {
      switch(op) {
        case ANYADIR_CUADRO:  opcionAnyadir(sc, galeria); break;
        case MOSTRAR_CUADROS: opcionMostrarCuadros(sc, galeria); break;
        case VENDER_CUADRO:   opcionVenderCuadro(sc, galeria); break;
        case EXPOSICION:      opcionExposicion(sc, galeria); break;
        case SALIR: break;
        default: break;
      }
      op = Menu(sc);
    }
    System.out.println("Fin␣del␣Programa.");
  }
  public static int Menu(Scanner sc) {
    System.out.println("\nQue␣quieres␣hacer?");
    System.out.println("1.␣Anyadir␣cuadro␣a␣la␣galeria");
    System.out.println("2.␣Mostrar␣cuadros␣de␣la␣galeria");
    System.out.println("3.␣Vender␣cuadro");
    System.out.println("4.␣Exposicion");
    System.out.println("5.␣Salir");
    return sc.nextInt();
  }

  static void opcionAnyadir(Scanner sc, Galeria galeria) {
    ...
    galeria.anyadirCuadro(cuadro);
   ...
  }
```

```
  static void opcionMostrarCuadros(Scanner sc, Galeria galeria) {

    System.out.println("La␣galeria␣tiene␣" + galeria.getNumCuadros() +"␣cuadros:");
    System.out.println(galeria);

  }

   static void opcionVenderCuadro(Scanner sc, Galeria galeria) {
   ...
   }

   static void opcionExposicion(Scanner sc, Galeria galeria) {
    ...

   }

}
}
```