

PARTE TEÓRICA - TEST [2,5 PUNTOS]:

Sólo una de las respuestas es válida. Las respuestas correctas se puntuarán con +1.0, mientras que las respondidas de manera incorrecta se puntuarán con -0.25. Las no contestadas no tendrán influencia ni positiva ni negativa en la nota.

Pregunta 1: Sobre los objetos, se puede decir (índica la respuesta **falsa**)

- a. Los objetos son especificados por las clases.
- b. Los objetos representan instancias de las clases.
- c. Los objetos se comunican con las clases con métodos.
- d. Una clase puede manejar objetos.

Pregunta 2: Dado este fragmento de código, ¿cuál sería el resultado de compilar/ejecutar el código?

Número de Línea Código

```
4     public static void main(String[] args) {  
5         int valor = 5;  
6         cambiarValor(valor);  
7         System.out.println(valor);  
8     }  
9     public static void cambiarValor(int valor) {  
10        private int valor = valor * 2;  
11    }
```

- a. 5
- b. 10
- c. Error en la línea 10
- d. Error en la línea 6

Pregunta 3: Dado el siguiente código, ¿cuál será su salida?

Número de Línea Código

```
4     class Cantante { public static String cantar() { return "la"; } }  
5     public class Tenor extends Cantante {  
6         public static String cantar() { return "fa"; }  
7         public static void main(String[] args) {  
8             Tenor t = new Tenor();  
9             Cantante s = new Tenor();  
10            System.out.println(t.cantar() + " " + s.cantar());  
11        }  
12    }
```

- a. fa fa
- b. fa la
- c. la la
- d. la fa

Pregunta 4: ¿Cuál de los siguientes condicionales compilaría sin errores?

```
int[] array = new int[15];  
// EL CÓDIGO IRÍA AQUÍ  
array[j] = j;
```

- a. for (int j=0; j<array.length; j++)
- b. for (int j=0; j<array.length(); j++)
- c. for (int j=0; j<array.size(); j++)
- d. for (int j=0; j<array.size(); j++)

Pregunta 5: Tienes que hacer una clase que almacena objetos únicos. No es necesario que estén ordenados. ¿Qué interfaz sería la más apropiada implementar en esta clase?

- a. Set
- b. List
- c. Map
- d. Vector

Pregunta 6: Cuando varios componentes de un software colaboran para completar una misma tarea se dice que entre ellos hay ...

- a. una clase clara y bien definida.
- b. una instancia clara y bien definida.
- c. una interfaz clara y bien definida.
- d. un proceso claro y bien definido.

Pregunta 7: Dado el siguiente código, ¿cuál será su salida?

Número de Línea Código

```
4      class Vehiculo {  
5          public void imprimirSonido() {  
6              System.out.print("Vehiculo");  
7          }  
8      }  
9      class Coche extends Vehiculo {  
10         public void imprimirSonido() {  
11             System.out.print("Coche");  
12         }  
13     }  
14     class Bicicleta extends Vehiculo {  
15         public void imprimirSonido() {  
16             System.out.print("Bicicleta");  
17         }  
18     }  
19     public class Test {  
20         public static void main(String[] args) {  
21             Vehiculo v = new Coche();  
22             Bicicleta b = (Bicicleta) v;  
23             v.imprimirSonido();  
24             b.imprimirSonido();  
25         }  
26     }
```

- a. Fallo de compilación.
- b. Lanza una excepción en tiempo de ejecución.
- c. Imprime "VehiculoCoche".
- d. Imprime "BicicletaBicicleta".

Pregunta 8: ¿Qué pasará si se compila / ejecuta este código?

Número de Línea Código

```
4     class Padre {}  
5     class Hijo extends Padre {}  
6     class Hijo2 extends Padre {}  
7     public class CEx{  
8         public static void main(String[] args){  
9             Padre p=new Padre();  
10            Hijo h=(Hijo) p;  
11        }  
12    }
```

- a. El código compilará y se ejecutará sin errores.
- b. El código daría un error a compilar.
- c. El código daría un error a ejecutar.
- d. El código no daría ningún error; sin embargo, h no tendría el tipo deseado.

Pregunta 9: ¿Cuál sería el resultado de ejecutar el método goo () ?

Número de Línea Código

```
4     public void goo() {  
5         foo f = new foo();  
6         System.out.println(f);  
7     }  
8     public class foo {  
9         String f = "22";  
10        public String toString(){  
11            return("44");  
12        }  
13        public foo () {}  
14    }
```

- a. null
- b. 22
- c. 44
- d. Un error de ejecución

Pregunta 10: ¿Cuál de las siguientes definiciones es correcta para una clase abstracta?

- a. abstract Animal {abstract void ladrar();}
- b. class abstract Animal {abstract void ladrar();}
- c. abstract class Animal {abstract void ladrar();}
- d. abstract class Animal {abstract void ladrar() {System.out.println("RRRRRRR");}}

Pregunta 11: ¿Cómo podemos detectar que el usuario ha hecho click en un botón en una interfaz Swing?

- a. Implementando public void eventPerformed(ActionEvent e) de la interfaz ActionListener
- b. Implementando public void actionPerformed(ActionEvent e) de la interfaz ActionListener
- c. Implementando public boolean actionPerformed(ActionEvent e) de la interfaz ActionListener
- d. Implementando public void actionPerformed(Event e) de la interfaz ActionListener

Pregunta 12: Las sentencias de código que podrían lanzar una excepción se protegen de la siguiente manera:

```
try {  
    // aquí se protege una o más sentencias  
}  
————— // aquí se informa y se recupera de la excepción  
}
```

¿Qué habría que colocar en el hueco correspondiente?

- a. catch (exception Exception)
- b. catch (Event exception)
- c. catch (Exception exception)
- d. catch (event Exception)

Pregunta 13: Las clases de un sistema corresponden a las X y los métodos a las Y. Donde X y Y son ...

- a. X = verbos, Y = sustantivos
- b. X = sustantivos, Y = verbos
- c. X = sustantivos, Y = sustantivos
- d. X = verbos, Y = verbos

Pregunta 14: ¿Cómo se puede crear una nueva instancia de la clase Vector y añadir un elemento? (índica la respuesta falsa)

- a. Vector<Integer> v = new Vector<Integer>(); v.add(99);
- b. Vector<Integer> v = new Vector<Integer>(99); v.add(99);
- c. Vector<Integer> v = new Vector<Integer>(99, 99); v.add(99);
- d. Vector<Integer> v = new Vector<Integer>(99); v.add(99, 99);

Pregunta 15: Sobre una variable local que se declara dentro del bloque "try", se puede decir que ...

- a. es visible dentro de los bloques "catch" y "finally".
- b. es visible dentro del bloque "catch" pero no del bloque "finally".
- c. es visible dentro del bloque "finally" pero no del bloque "catch".
- d. no es visible dentro de los bloques "catch" y "finally".

PARTE PRÁCTICA [6,5 PUNTOS]:

Una empresa de alquiler de automóviles tiene a su disposición un conjunto de vehículos indicados en la siguiente tabla. Se quiere diseñar e implementar un programa que almacene y gestione la información relacionada con estos vehículos.

Tipo de vehículo	Características
Motos	marca, matrícula, número de identificación, número de kilómetros, estado actual de depósito de gasolina.
Coches (turismos)	marca, matrícula, número de identificación, tipo (normal / familiar), número de puertas, número de kilómetros, tipo de motor (gasolina / gasoil), estado actual del depósito.
Coches (deportivos)	marca, matrícula, número de identificación, capacidad de motor, número de kilómetros, turbo o no, número de puertas, número de asientos, estado actual del depósito de gasolina.
Coches (4x4)	marca, matrícula, número de identificación, número de kilómetros, tipo de motor (gasolina / gasoil), número de asientos, estado actual de depósito.
Monovolúmenes	marca, matrícula, número de identificación, número de kilómetros, número de puertas, puertas laterales, número de asientos, tipo de motor (gasolina / gasoil), capacidad de carga, estado actual del depósito.
Furgonetas	marca, matrícula, número de identificación, número de kilómetros, capacidad de carga, altura, estado actual del depósito de gasoil.

Se pide:

- 1) **[1,5 puntos]** Identificar la estructura y las relaciones de herencia y de uso de las clases necesarias para almacenar y gestionar esta información.
- 2) **[1,5 puntos]** Dibujar un esquema de la organización de estas clases en el diseño global.
- 3) **[2,0 puntos]** Implementar la especificación de las clases.
- 4) **[1,5 puntos]** Se quiere declarar un array de objetos para almacenar todos los vehículos, independientemente del tipo.
 - a) ¿Cómo declararías el array?
 - b) ¿Qué métodos se necesitan para acceder a un vehículo concreto?
 - c) ¿Cómo se almacenan las diferencias entre los distintos tipos de vehículos?

PARTE TEÓRICA - TEST [2,5 PUNTOS]:

Sólo una de las respuestas es válida. Las respuestas correctas se puntuarán con +1.0, mientras que las respondidas de manera incorrecta se puntuarán con -0.25. Las no contestadas no tendrán influencia ni positiva ni negativa en la nota.

Pregunta 1: Dada la declaración de las siguientes variables, indicar cuáles de ellas son correctas.

1. float foo = -1;
2. float fool = 1.0;
3. float foo2 = 42e1;
4. float foo3 = 2.02f;
5. float foo4 = 3.03d;
6. float foo5 = 0x0123;

- a. 1 y 2
- b. 1 y 3
- c. 4 y 6
- d. 3 y 4

Pregunta 2: Dado el siguiente fragmento de código, indica cuál de las siguientes afirmaciones es correcta en relación al valor de la variable foo.

Número de Línea Código

```
4     int index = 1;
5     boolean[] test = new boolean[3];
6     boolean foo = test [index];
```

- a. foo tiene el valor 0
- b. foo tiene el valor null
- c. foo tiene el valor false
- d. Se produce una excepción y foo no posee ningún valor

Pregunta 3: Dadas las siguientes expresiones, indica cuál de las opciones es la correcta.

1. (1 > 1) && (1 > 1) == (1 > 1) == false
2. (1 == 1) | (10 > 1) == true | true == true

- a. La expresión 1 es evaluada como falsa y la expresión 2 como falsa.
- b. La expresión 1 es evaluada como falsa y la expresión 2 como verdadera.
- c. La expresión 1 es evaluada como verdadera y la expresión 2 como falsa.
- d. La expresión 1 es evaluada como verdadera y la expresión 2 como verdadera.

Pregunta 4: Dado el siguiente código, ¿cuál es su resultado?

Número de Línea Código

```
4     class Top {
5         public Top(String s) { System.out.print("B"); }
6     }
7     public class Bottom2 extends Top {
8         public Bottom2(String s) { System.out.print("D"); }
9         public static void main(String [] args) {
10            Bottom2 obj=new Bottom2("C");
11            System.out.println(" ");
12        }
13    }
```

- a. BD
- b. DB
- c. BDC
- d. Error de compilación

Pregunta 5: Dado el siguiente código, ¿cuál de las afirmaciones es cierta?

Número de Línea Código

```
4     class Hotel {  
5         public int reservas;  
6         public void reservar() {  
7             reservas++;  
8         }  
9     }  
10    public class SuperHotel extends Hotel {  
11        public void reservar() {  
12            reservas--;  
13        }  
14        public void reservar(int size) {  
15            reservar();  
16            super.reservar();  
17            reservas += size;  
18        }  
19        public static void main(String[] args) {  
20            SuperHotel hotel = new SuperHotel();  
21            hotel.reservar(2);  
22            System.out.print(hotel.reservas);  
23        }  
24    }
```

- a. Error de compilación.
- b. Lanza una excepción en tiempo de ejecución.
- c. 0.
- d. 2.

Pregunta 6: Según el texto de la bibliografía básica de la asignatura, indique cuál de las siguientes afirmaciones es correcta:

- a. La depuración es la actividad cuyo objetivo es determinar si una pieza de código produce el comportamiento pretendido.
- b. La prueba viene a continuación de la depuración.
- c. La depuración es una actividad dedicada a determinar si un segmento de código contiene errores.
- d. La depuración es el intento de apuntar con precisión y corregir un error en el código.

Pregunta 7: Según el texto de la bibliografía básica de la asignatura, indique cuál de las siguientes afirmaciones es correcta:

- a. Un encapsulamiento apropiado en las clases reduce el acoplamiento.
- b. El término acoplamiento describe cuánto se ajusta una unidad de código a una tarea lógica o a una entidad.
- c. El acoplamiento describe la conectividad de los propios objetos de una clase.
- d. Un sistema débilmente acoplado se caracteriza por la imposibilidad de modificar una de sus clases sin tener que realizar cambios en ninguna otra.

Pregunta 8: Según el texto de la bibliografía básica de la asignatura, indique cuál de las siguientes afirmaciones es FALSA en relación a los métodos polimórficos:

- a. Una variable polimórfica es aquella que puede almacenar objetos de diversos tipos.
- b. Las llamadas a métodos en Java no son polimórficas.
- c. El mismo método puede invocar en diferentes momentos diferentes métodos dependiendo del tipo dinámico de la variable usada para hacer la invocación.
- d. Cada objeto en Java tiene un método `toString()` que puede usarse para devolver un `String` de su representación.

Pregunta 9: Según el texto de la bibliografía básica de la asignatura, indique cuál de las siguientes opciones declarará un método en una clase que fuerza a una subclase a implementarlo:

- a. `static void methoda (double d1) {}`
- b. `public native double methoda();`
- c. `abstract public void methoda();`
- d. `protected void methoda (double d1) {}`

Pregunta 10: Dado el siguiente fragmento de código que pretende mostrar un ejemplo de sobrescritura, indique cuál de las siguientes opciones completaría el código para dar lugar a un ejemplo correcto de sobrescritura:

Número de Línea **Código**

```
4     class BaseClass {  
5         private float x = 1.0f ;  
6         protected float getVar () {return x;}  
7     }  
8     class Subclass extends BaseClass {  
9         private float x = 2.0f;  
10        //Insertar código aquí  
11    }
```

a. float getVar () { return x; }
b. public float getVar () { return x; }
c. float double getVar () { return x; }
d. public float getVar (float f) { return f; }

Pregunta 11: Según el texto de la bibliografía básica de la asignatura, indique cuál de las siguientes afirmaciones es correcta en relación a la programación por parejas:

- a. Consiste en programar una clase por duplicado con el objetivo de depurar los errores más fácilmente.
- b. Es una manera de producir código, opuesta a la programación extrema en la que un solo programador desarrolla las clases asignadas.
- c. Era una técnica de programación tradicional que las empresas eliminaron para reducir costes.
- d. Es uno de los elementos de una técnica que se conoce como programación extrema.

Pregunta 12: La ejecución del siguiente fragmento de código ...

Número de Línea **Código**

```
4     import javax.swing.*;  
5     class PrimerFrame extends JFrame  
6     {  
7         public PrimerFrame()  
8         {  
9             setTitle("Mi primer programa gráfico");  
10            setSize(400,100);  
11        }  
12    }  
13    public class FrameTest  
14    {  
15        public static void main (String[] args)  
16        {  
17            JFrame frame = new PrimerFrame();  
18            frame.setVisible (true);  
19        }  
20    }
```

Da lugar al siguiente programa:



Pero este último programa tiene el problema de que cuando se cierra la ventana, a pesar de que dejamos de verla, el programa no finaliza su ejecución. De esta forma, para que el programa funcione correctamente, hemos de interceptar el evento que se produce cuando cerramos la ventana y hacer que el programa termine su ejecución en ese momento. Indique qué clase hemos de definir en este caso y asociárselo al `JFrame` del ejemplo:

- a. ActionListener
- b. ComponentListener
- c. WindowListener
- d. ItemListener

Pregunta 13: En el siguiente fragmento de código hemos definido la ejecución de cinco bloques. Estos bloques se ejecutarán dependiendo de las excepciones que se produzcan en cada caso. Indique cuál de las siguientes afirmaciones es correcta:

Número de Línea **Código**

```
4      // Bloque1
5      try{
6          // Bloque2
7      }catch (ArithmetricException e) {
8          // Bloque3
9      }finally{
10         // Bloque4
11     }
12     // Bloque5
```

- a. El Bloque4 no se ejecutará si se produce una excepción de tipo aritmético en el Bloque2
- b. El Bloque4 no se ejecutará si se produce un acceso a un objeto nulo (null) en el Bloque2
- c. El Bloque4 se ejecutará antes que el Bloque3 si se produce una excepción de tipo aritmético en el Bloque2
- d. El Bloque4 se ejecutará antes de que la excepción producida por un acceso a un objeto nulo (null) en el Bloque2 se propague hacia arriba

Pregunta 14: Indique el resultado de ejecutar el siguiente código que se muestra a continuación:

Número de Línea **Código**

```
4      public class test {
5          public static void add3 (Integer i) {
6              int val = i.intValue();
7              val += 3;
8              i = new Integer (val);
9          }
10         public static void main (String args[]) {
11             Integer i = new Integer (0);
12             add3 (i);
13             System.out.println (i.intValue ( ) );
14         }
15     }
```

- a. El programa indicará un fallo en tiempo de compilación.
- b. El programa imprime por pantalla el valor “0”.
- c. El programa imprime por pantalla el valor “3”.
- d. El programa lanzará una excepción en la línea 6 (int val = i.intValue());.

Pregunta 15: Dado el siguiente código ...

Número de Línea **Código**

```
4      public class testJunio {
5          public void setVar (int a, int b, float c) {
6          }
7          // INSERTAR CÓDIGO AQUÍ
8      }
```

Y los siguientes métodos:

```
1      private void setVar (int a, float c, int b) { }
2      protected void setVar (int a, int b, float c) { }
3      public int setVar (float a, int b, int c) {return b;}
4      public int setVar (int a, int b, float c) {return a;}
5      protected float setVar (int a, int b, float c) {return c;}
```

Indique qué métodos permiten una sobrecarga del método setVar de manera correcta:

- a. 1 y 2
- b. 1 y 3
- c. 3 y 5
- d. 3 y 4

PARTE PRÁCTICA [6,5 PUNTOS]:

El juego del Cinquillo Solitario es una variedad del popular Cinquillo en el cual un jugador puede jugar de manera online contra el ordenador. El juego se inicia con el reparto de todas las cartas de una baraja española que consta de 48 naipes o cartas, clasificados en cuatro palos (oros, bastos, copas y espadas) y numerados del 1 al 12. El objetivo del juego consiste en descartarse (quedarse sin cartas) antes que el oponente.

El jugador que posee el cinco de oros lo coloca boca arriba encima de la mesa y de esta forma empieza el turno de descartes. En turnos alternativos, cada jugador puede descartarse de máximo un naipe. Solo se pueden colocar cinco o todas aquellas cartas que siguen en progresión ascendente o descendente a las que hay en la mesa y sean del mismo palo. Es decir, si por ejemplo solamente está colocado el cinco de oros en la mesa, los jugadores solo podrán colocar el seis o el cuatro de oros o un cinco de otro palo.

Si un jugador no puede colocar ninguna carta pasa, y le toca el turno al siguiente jugador. Nunca se puede pasar si se puede colocar alguna carta. El primer jugador que consigue colocar todas sus cartas sobre la mesa es el ganador.

En cuanto a la dinámica del juego, uno de los contrincantes será un jugador humano (introducimos sus datos y sus preferencias por el teclado) y el otro contrincante será el propio ordenador.

- a) **[1,5 puntos]** Diseñe las clases necesarias que permita desarrollar el juego del Cinquillo Online utilizando un paradigma orientado a objetos. Debe hacerse uso de los mecanismos de la programación orientada a objetos siempre que sea posible y un diseño que permita la reutilización del código y facilite su mantenimiento.
- b) **[1,5 puntos]** Implemente un método que defina el funcionamiento del ordenador, teniendo en cuenta que todos sus procesos tienen que hacerse automáticamente sin la intervención del usuario.
- c) **[1,5 puntos]** Proporcione un método que muestre la lógica del juego, definiendo la información necesaria para establecer el uso de clases, interacciones entre elementos, declaración y uso de variables y métodos necesarios, etc.
- d) **[2,0 puntos]** Indique qué modificaciones son necesarias introducir en la aplicación para permitir la participación de varios jugadores humanos (hasta 4). Para ello el juego en lugar de constar de partidas individuales en las cuales gana el jugador que antes se descarta, para a ser una partida formada por un conjunto de rondas. El ordenador deberá llevar un registro de los puntos que cada jugador ha conseguido en cada ronda. El jugador que consigue descartarse primero logrará 3 puntos, el jugador o jugadores que se quede con un mayor número de cartas al finalizar la ronda obtendrá 0 puntos. El resto obtendrá 1 punto. La partida finaliza cuando un jugador consiga llegar al menos a los 10 puntos, ganando el que más puntos tenga en caso de superar esta puntuación varios jugadores. En caso de empate se jugará una ronda extra para decidir el ganador.

PARTE TEÓRICA - TEST [2,5 PUNTOS]:

Sólo una de las respuestas es válida. Las respuestas correctas se puntuarán con +1.0, mientras que las respondidas de manera incorrecta se puntuarán con -0.25. Las no contestadas no tendrán influencia ni positiva ni negativa en la nota.

Pregunta 1 ¿Cuál de las siguientes inicializaciones para un array es la correcta?

- a) Array nombreArray3 = new int[20];
- b) int [] nombreArray4 = new int[20];
- c) int [] nombreArray1 = new Array(20);
- d) Array nombreArray2 = new Array(20);

Pregunta 2 ¿Qué pasaría a continuación si se crea un objeto sin parámetros con la sentencia A objA = new A(); siendo la clase A la siguiente?

```
public class A
{
    int nA;
    public A(int nA) {this.nA=nA;}
}
```

- a) Se crearía un objeto objA ya que se llamaría al constructor por defecto de la clase A
- b) Se crearía un objeto objA, aunque al no tener parámetros, se pasaría un valor nulo al constructor publicA(int nA){this.nA = nA;}
- c) Daría un error de compilación debido a que si se declara un constructor en una clase el constructor por defecto deja de ser accesible
- d) Daría un error de compilación por poner la palabra reservada new delante de la expresión A()

Pregunta 3 ¿Qué es necesario si queremos utilizar el método de una clase predefinida como por ejemplo Integer.parseInt("String");?

- a) Solamente se crea una instancia de la clase predefinida.
En este caso: Integer integ = new Integer(); integ.parseInt("String");
- b) No es necesario importar la clase en cuestión ni crear una instancia de la clase predefinida
- c) Importar la clase en cuestión. En este caso: import java.lang.Integer; y crear una instancia de la clase predefinida. En este caso: Integer integ = new Integer(); integ.parseInt("String");
- d) Solamente se importa la clase en cuestión. En este caso: import java.lang.Integer;

Pregunta 4 Un método de clase static, ¿se podría llamar sin instanciarse un objeto de la clase?

- a) No, siempre hay que declarar el objeto y luego llamar al método
- b) No, siempre hay que declarar el objeto, crearlo y a continuación llamar al método
- c) Sí, se podría llamar al método desde la misma clase
- d) Sí, se podría llamar al método pero sólo si la clase es abstracta

Pregunta 5 Suponiendo que, en una estructura de herencia, cada subclase de la clase abstracta "figura" tiene su propio método "calcula_area", y dado un objeto "f" declarado como "figura", ¿cómo tendrían que estar declarados los métodos para poder ejecutar "f.calcula_area()"?

- a) "calcula_area" sólo en las subclases
- b) "calcula_area" en "figura" y en sus subclases
- c) "calcula_area" en figura y no en sus subclases
- d) No puede ser ejecutado de ninguna manera

Pregunta 6 Sea “c1” una clase con 6 atributos, y sea “c2” otra clase con 3 atributos, 3 de los atributos de “c1” son los de la clase “c2”. Tienen los mismos métodos aunque algunos se comportan de manera diferente. ¿Qué relación de herencia es la correcta y por qué?

- a) No se puede establecer herencia dado que los mismos métodos se comportan de manera diferente
- b) No se puede establecer herencia dado que los atributos no son exactamente los mismos
- c) “c2” es hija de “c1” dado que todos sus atributos están incluidos en “c2”
- d) “c1” es hija de “c2” dado que incluye sus atributos y añade nuevos

Pregunta 7 ¿Cuál de las siguientes declaraciones nunca podría generar dos objetos de la misma clase con los mismo valores?

- a) Triangulo objCirl=new Triangulo(5,8);
Triangulo objCir2=new Triangulo(objCirl);
- b) Triangulo objCirl=new Triangulo(5,8);
Triangulo objCir2=objCirl;
- c) Triangulo objCirl=new Triangulo(5,8);
Triangulo objCir2=new Triangulo(5,8);
- d) Triangulo objCirl=new Triangulo(5);
Triangulo objCir2=new Triangulo(5);

Pregunta 8 Una diferencia entre HashSet y TreeSetes ...

- a) HashSet implementa la interfaz Set y TreeSet no
- b) En TreeSet se permiten elementos repetidos y en HashSet no
- c) TreeSet mantiene todos los elementos ordenados en su orden natural o de acuerdo a como indique el Comparator que se indica en el constructor, y HashSet no hace esto
- d) HashSet se puede recorrer mediante un Iterator, mientras que TreeSet no

Pregunta 9 ¿Qué tipo de acceso se puede emplear para sobrecargar los métodos que implementan una interfaz?

- a) Pueden ser métodos con acceso public y private, pero no protected
- b) Sólo pueden ser con acceso public
- c) Sólo pueden ser con acceso private
- d) Pueden emplear cualquier tipo de acceso de Java

Pregunta 10 ¿Qué se entiende por cohesión en lo que a una unidad de código se refiere?

- a) Que la unidad de código es responsable de una y sólo una tarea
- b) Que la unidad de código no se descompone en otras unidades de programación (métodos)
- c) Que todos los métodos que forman parte de la unidad de código se encuentran en la misma clase
- d) Todas las anteriores son correctas

Pregunta 11 Respecto a las variables polimórficas en Java...

- a) Es aquella que puede almacenar varios objetos de diferentes tipos
- b) Ese concepto, al igual que la herencia múltiple, no se implementa en Java
- c) Cada variable objeto en Java es potencialmente polimórfica
- d) Se declaran siempre como public o private, pero nunca protected

Pregunta 12 Respecto a la declaración de las clases internas y su relación con su clase envolvente...

- a) La clase interna puede acceder tanto a los métodos privados como a los públicos y protegidos de la clase envolvente
- b) La clase interna puede acceder tanto a los métodos públicos como a los métodos protegidos de la clase envolvente, pero no a los métodos privados de la misma
- c) La clase interna puede acceder a los métodos públicos de la clase envolvente, pero no a los métodos privados ni a los protegidos de la misma
- d) La clase interna no puede acceder ni a los métodos públicos ni a los privados ni a los protegidos de la clase envolvente

Pregunta 13 En lo que se refiere a las clases internas anónimas, se puede afirmar...

- a) Suelen emplearse en los lugares en los que se requiere la implementación de una sola instancia
- b) Siempre se hará referencia la instancia mediante su supertipo
- c) Permiten definir una clase y crear una instancia de ella, todo en un solo paso
- d) Todas las anteriores son correctas

Pregunta 14 ¿Cuál de las siguientes afirmaciones es cierta para los interfaces en Java?

- a) Todos los métodos de la interfaz son abstractos, pero se permiten métodos con cuerpos. No es necesaria la palabra clave *abstract*
- b) Todos los métodos de la interfaz son abstractos, luego no se permiten métodos con cuerpos. Es necesaria la palabra clave *abstract*
- c) Todos los métodos de la interfaz son abstractos, luego no se permiten métodos con cuerpos. No es necesaria la palabra clave *abstract*
- d) Ninguna de las afirmaciones anteriores es cierta

Pregunta 15 ¿Qué instrucción permite cargar en la variable "a" el tamaño del array?

- a) int a; int [] b = new int[10]; a=b.size;
- b) int a; int [] b = new int[10]; a=b.size();
- c) int a; int [] b = new int[10]; a=b.length;
- d) int a; int [] b = new int[10]; a=b.length();

PARTE PRÁCTICA [6,5 PUNTOS]:

Un banco desea enviar a sus clientes una carta, mensaje de correo electrónico o mensaje al móvil (según los datos y preferencias de cada cliente) de agradecimiento por cada uno de los productos financieros que ha contratado el cliente. Cada cliente puede haber contratado varios productos (tarjetas de crédito, débito, plan de pensiones, seguro, fondos de inversión, etc.) y cada tipo de producto requiere una carta/mensaje diferente. Se quiere usar una jerarquía de clases para representar los diferentes tipos de producto y otra para los diferentes tipos de mensajes. Para el programa hay que usar un ArrayList del tipo más adecuado más un iterador para gestionar el envío de mensajes conjuntamente.

- a) **[2 puntos]** Identificar la estructura y las relaciones de herencia y de uso de las clases necesarias para la aplicación que realice el trabajo descrito.
- b) **[2 puntos]** Dibujar un esquema de la organización de estas clases en el diseño global.
- c) **[2 puntos]** Implementar el método “main” del programa mostrando cómo se gestiona el envío de los mensajes.
- d) **[0,5 puntos]** ¿Qué cambios serían necesarios en el diseño y programa para mandar otro tipo de mensaje, algo de publicidad sobre nuevos productos, conjuntamente con el mensaje de agradecimiento?

PARTE TEÓRICA - TEST [2,5 PUNTOS]:

Sólo una de las respuestas es válida. Las respuestas correctas se puntuarán con +1.0, mientras que las respondidas de manera incorrecta se puntuarán con -0.25. Las no contestadas no tendrán influencia ni positiva ni negativa en la nota.

Pregunta 1: Dada la siguiente clase TV:

```
1. public class TV {  
2.     private String marca;  
3.     private String modelo;  
4.  
5.     public TV(String marca, String modelo) {  
6.         this.marca = marca;  
7.         this.modelo = modelo;  
8.     }  
9.  
10.    public boolean equals(TV other) {  
11.        return marca.equals(other.marca) &&  
12.        modelo.equals(other.modelo);  
13.    }  
14.  
15. }
```

¿Cuál sería el resultado de ejecutar el siguiente código?

```
TV a = new TV("Philips", "42PFL5603D");  
TV b = new TV("Philips", "42PFL5603D");  
if(a.equals(b)) {  
    System.out.println("iguales");  
} else {  
    System.out.println("no son iguales");  
}
```

- a) iguales
- b) no son iguales
- c) Error de compilación en la línea 11
- d) Excepción en tiempo de ejecución en la línea 15

Pregunta 2: ¿Cuál es el resultado de la ejecución de las siguientes líneas de código?

```
28. Integer i = 5;  
29. switch(i) {  
30.     case 1: System.out.print(1); break;  
31.     case 3: System.out.print(3);  
32.     case 5: System.out.print(5);  
33.     case 7: System.out.print(7); break;  
34.     default: System.out.print("default");  
35. }
```

- a) 5
- b) 57
- c) 57default
- d) Error de compilación en la línea 29

Pregunta 3: Dado el siguiente código:

```
30. Set < Object >objetos = new HashSet< Object > ();  
31. String one = "hola";  
32. int two = 2;  
33. Boolean three = new Boolean(true);  
34. objetos.add(one);  
35. objetos.add(two);  
36. objetos.add(three);  
37. objetos.add(three);  
38. for(Object objeto : objetos) {  
39.     System.out.print(objeto); }
```

¿Cuál de las siguientes afirmaciones es cierta?

- a) La salida es hola, 2 y true en un orden no determinado.
- b) La salida es hola, 2, true y true en un orden no determinado.
- c) Error de compilación en la línea 35.
- d) Excepción en tiempo0 de ejecución en la línea 37.

Pregunta 4: Dadas las siguientes definiciones de clase y de interfaz:

```
1. //Legible.java  
2. public interface Legible {  
3.     public void leer();  
4.     public int MAX_LENGTH = 10;  
5. }  
  
1. //MiLector.java  
2. public class MiLector implements Legible {  
3.     public void leer() {  
4.         Legible.MAX_LENGTH = 25;  
5.         System.out.println(Legible.MAX_LENGTH);  
6.     }  
7. }
```

¿Cuál sería el resultado de ejecutar la siguiente línea de código? newMiLector().leer();

- a) 25
- b) 10
- c) Error de compilación en la línea 4 de Legible.java
- d) Error de compilación en la línea 4 de MiLector.java

Pregunta 5: ¿Cuál es la salida del siguiente código?

```
5. int x = 5 * 4 % 3;  
6. System.out.println(x);
```

- a) Error de compilación en la línea 5.
- b) 2
- c) 3
- d) 6

Pregunta 6: ¿Cuál sería la salida del siguiente código?

```
3. int x = 0;
4. String s = null;
5. if(x == s) {
6.     System.out.println("Exito");
7. } else {
8.     System.out.println("Fracaso");
9. }
```

- a) Éxito
- b) Fracaso
- c) Error de compilación en la línea 4.
- d) Error de compilación en la línea 5.

Pregunta 7: ¿Cuál sería la salida del siguiente código?

```
1. public class Forma {
2.     private String color;
3.
4.     public Forma(String color) {
5.         System.out.print("Forma");
6.         this.color = color;
7.     }
8.
9.     public static void main(String [] args) {
10.        new Rectangulo();
11.    }
12.}
13.
14. class Rectangulo extends Forma {
15.     public Rectangulo() {
16.         System.out.print("Rectangulo");
17.     }
18. }
```

- a) FormaRectangulo
- b) RectanguloForma
- c) Rectangulo
- d) Error de compilación en la línea 15

Pregunta 8: Dada la siguiente definición de clase:

```
1. import java.awt.*;
2. import java.awt.event.*;
3.
4. public class MyWindow {
5.     private Frame frame = new Frame();
6.
7.     public void registerEvents() {
8.         WindowAdapter wa = new WindowAdapter() {
9.             public void windowClosing(WindowEvent e) {
10.                 frame.setVisible(false);
11.                 frame.dispose();
12.             }
13.         };
14.         frame.addWindowListener(wa);
15.     }
16. }
```

¿Cuál de las siguientes afirmaciones es cierta?

- a) Hay un error de compilación en las líneas 10 y 11.
- b) El objeto que se instancia en la línea 8 no tiene acceso al campo frame de la línea 5 porque este es privado.
- c) El método de la línea 9 no se ejecuta nunca ya que deja de ser accesible a partir de la línea 15.
- d) La clase anónima anidada de la línea 8 extiende la clase WindowAdapter.

Pregunta 9: ¿Cuál es la salida del siguiente código?

```
3. int x = 10, y = 3;
4. if(x % y == 2)
5. System.out.print("dos");
6. System.out.print(x%y);
7. if(x%y == 1)
8. System.out.print("uno");
```

- a) dos1
- b) dos2
- c) uno
- d) 1uno

Pregunta 10: ¿Cuál es el resultado del siguiente código?

```
4. final char a = 'A', d = 'D';
5. char nota = 'B';
6. switch(nota) {
7.     case a :
8.     case 'B' :
9.         System.out.print(" enhorabuena ");
10.    case 'C' :
11.        System.out.print(" aprobado ");
12.        break;
13.    case d :
14.    case 'F' :
15.        System.out.print(" notgood ");
16. }
```

- a) enhorabuena
- b) enhorabuenaaprobado
- c) Error de compilación en la línea 4
- d) Error de compilación en la línea 7

Pregunta 11: ¿Cuál es la salida del siguiente código?

```
1. public class Incognita {  
2.     public static int metodoIncognita(String input) {  
3.         int count = 0;  
4.         int length = input.length();  
5.         int i = 0;  
6.  
7.         String lowercase = input.toLowerCase();  
8.         while(i < length) {  
9.             switch(lowercase.charAt(i)) {  
10.                 case 'a':  
11.                 case 'e':  
12.                 case 'i':  
13.                 case 'o':  
14.                 case 'u':  
15.                     count++;  
16.             }  
17.             i++;  
18.         }  
19.         return count;  
20.     }  
21.  
22.     public static void main(String [] args) {  
23.         int x = metodoIncognita("Otorrinolaringólogo");  
24.         System.out.print(x);  
25.     }  
26. }
```

- a) 0
- b) 9
- c) 19
- d) 20

Pregunta 12: ¿Cuál es el resultado del siguiente programa?

```
1. public class ComparadorRaro {  
2.     private Integer x;  
3.  
4.     public boolean compare(int y) {  
5.         return x == y;  
6.     }  
7.  
8.     public static void main(String [] args) {  
9.         ComparadorRaro u = new ComparadorRaro();  
10.        if(u.compare(21)) {  
11.            System.out.println("true");  
12.        } else {  
13.            System.out.println("false");  
14.        }  
15.    }  
16. }
```

- a) true
- b) false
- c) Error de compilación en la línea 5.
- d) La línea 5 lanza una excepción NullPointerException

Pregunta 13: Termina la frase. Si todos los campos no finales de una clase se declaran como privados y, además, la clase contiene métodos públicos para modificar o consultar dichos campos, esto es un ejemplo de:

- a) Encapsulamiento alto
- b) Acoplamiento bajo
- c) Cohesión alta
- d) Una relación “es un”

Pregunta 14: Dada la siguiente clase Television:

```
public class Television {  
    public int canal;  
    private boolean estaEncendida;  
    private int volumen;  
    public void cambiarCanal(int nuevoCanal) {  
        canal = nuevoCanal;  
    }  
    public int consultarCanal() {  
        return canal;  
    }  
    public void encender() {  
        estaEncendida = true;  
    }  
    public void apagar() {  
        estaEncendida = false;  
    }  
    public void subirVolumen() {  
        volumen += 1;  
    }  
    public void bajarVolumen() {  
        volumen -= 1;  
    }  
}
```

¿Qué podemos afirmar?

- a) La clase está altamente encapsulada.
- b) La clase está altamente acoplada
- c) La clase tiene un grado de cohesión alto
- d) La clase tiene un grado de cohesión bajo

Pregunta 15: Dada la siguiente declaración:

```
Map < String, Double > map = new HashMap< String, Double > ();
```

¿Cuál de las siguientes opciones es correcta?

- a) map.add(“pi”, 3.14159);
- b) map.add(“e”, 2.71828D);
- c) map.add(“log(1)”, new Double(0.0));
- d) Ninguna de las anteriores.

PARTE PRÁCTICA [6,5 PUNTOS]:

La Universidad Sin Distancias (USD) quiere diseñar un sistema de gestión de matrículas de alumnos, del que también forman parte los profesores. Las universidades a nivel nacional se identifican por el nombre, la dirección y el teléfono de información. Desde el punto de vista de la matrícula, la USD se considera como un conjunto de estudiantes, a cada uno de los cuales se le debe asignar un número de identificación personal. Los estudiantes asisten a cierto número de cursos, cada uno de los cuales es impartido por un profesor. En cuanto a los profesores, la USD se divide en departamentos, cada uno de los cuales está formado por varios profesores, uno de los cuales actúa como director del departamento. Cada profesor sólo puede pertenecer a un departamento. A su vez, los departamentos tienen la responsabilidad de impartir uno o más cursos. El sistema debe permitir añadir o borrar estudiantes, departamentos y profesores, así como realizar consultas sobre cada uno de estos estamentos.

- a. **[2 puntos]** Identifique las clases necesarias para resolver el problema. Indique, para cada una de ellas, sus miembros de clase y su ámbito, así como las relaciones existentes entre estas. Ilustre esto último con un diagrama de clases.
- b. **[1,5 puntos]** Suponiendo que existen y están definidos los métodos "get" y "set" para aquellos miembros de clase que haya definido como privados, escriba los métodos: "nuevaMatrícula" que permita realizar la gestión de una nueva matrícula tal y como se ha descrito en la especificación y el método "nuevoProfesor" que permita añadir un nuevo profesor en el organigrama de la Universidad. Indique además en qué clase/es incluiría estos métodos.
- a) **[1,5 puntos]** Suponiendo que existen y están definidos los métodos "get" y "set" para aquellos miembros de clase que haya definido como privados, escriba un método "nuevoDepartamento" que permita añadir un nuevo departamento con su correspondiente nuevo conjunto de cursos y profesores. Indique además en qué clase o clases se deberían incluir estos métodos.
- c. **[1,5 puntos]** En el caso de que la Universidad decidiera realizar una división interna en la que los departamentos pertenecieran a una determinada escuela o facultad, ¿qué modificaciones se tendrían que realizar en el diseño de las clases y en el método "nuevoProfesor"?

UNIVERSIDAD NACIONAL DE EDUCACIÓN A DISTANCIA – ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA
INFORMÁTICA

71901072 – PROGRAMACIÓN ORIENTADA A OBJETOS (GRADO EN INGENIERÍA INFORMÁTICA / TECNOLOGÍAS DE LA INFORMACIÓN)

JUNIO 2012 – MODELO A – **NO ESTÁ PERMITIDO EL USO DE MATERIAL ADICIONAL**

PARTE TEÓRICA - TEST [2,5 PUNTOS]:

Sólo una de las respuestas es válida. Las respuestas correctas se puntuarán con +1.0, mientras que las respondidas de manera incorrecta se puntuarán con -0.25. Las no contestadas no tendrán influencia ni positiva ni negativa en la nota.

Pregunta 1: Según el texto de la bibliografía básica de la asignatura, indique cuál de las siguientes afirmaciones es correcta:

- a. Los métodos pueden devolver información de algún objeto mediante un valor de retorno.
- b. Los métodos siempre tienen parámetros con los que obtener la información necesaria.
- c. A partir de una clase tan solo se puede crear un solo objeto.
- d. El estado de los objetos se representa mediante los parámetros de su constructor.

Pregunta 2: Dado el siguiente fragmento de código, indique cuál de las siguientes afirmaciones es el resultado de su ejecución:

```
if(" String ".trim() == "String")
    System.out.println("Igual");
else
    System.out.println("No Igual");
```

- a. El código compilará e imprimirá “Igual”.
- b. El código compilará e imprimirá “No Igual”.
- c. El código provocará un error de compilación.
- d. El código provocará un error en tiempo de ejecución.

Pregunta 3: Según el texto de la bibliografía básica de la asignatura, indique cuál de las siguientes afirmaciones es correcta:

- a. Los campos se conocen como variables de objeto.
- b. El alcance de una variable define la sección de código desde donde la variable puede ser declarada.
- c. Los constructores permiten que cada objeto sea preparado adecuadamente cuando es creado.
- d. El tiempo de vida de una variable describe el número de veces que es utilizada en un método.

Pregunta 4: Según el texto de la bibliografía básica de la asignatura, indique cuales de las siguientes expresiones resultan verdaderas:

1. ! (4 < 5)
2. (2 > 2) || ((4 == 4) && (1 < 0))
3. (2 > 2) || (4 == 4) && (1 < 0)
4. (2 > 2) || !((4 == 4) && (1 < 0))
5. (34 != 33) && ! false

- a. Las expresiones 3 y 4.
- b. Las expresiones 2 y 4.
- c. Las expresiones 3 y 5.
- d. Las expresiones 4 y 5.

Pregunta 5: Según el texto de la bibliografía básica de la asignatura, indique cuál de las siguientes afirmaciones es correcta:

- a. El lenguaje Java tiene tres variantes del ciclo for : for-each, for y for-do.
- b. Un ciclo while es similar en su estructura y propósito que el ciclo for-each.
- c. El tipo de la variable de ciclo no tiene porqué ser el mismo que el tipo del elemento declarado para la colección que estamos recorriendo con un ciclo.
- d. Un índice es un objeto que proporciona funcionalidad para recorrer todos los elementos de una colección.

Pregunta 6: La siguiente figura muestra una captura de pantalla del editor BlueJ con una línea de código recuadrada. Indica cual de las siguientes afirmaciones es correcta en relación a la línea recuadrada:

- a. Muestra un error en tiempo de ejecución.
- b. Muestra un error de compilación.
- c. Muestra un punto de interrupción.
- d. Muestra una el lanzamiento de una excepción.

```
ClienteDeCorreo
Class Edit Tools Options
Compile Undo Cut Copy Paste Find... Find Next Close

/*
 * Imprime el siguiente mensaje (si es que hay alguno) para este
 * usuario en la terminal de texto.
 */
public void imprimirMensajeSiguiiente()

    Mensaje unMensaje = servidor.getMensajeSiguiiente(usuario);
    if(unMensaje == null) {
        System.out.println("No hay ningún mensaje nuevo.");
    }
    else {
        unMensaje.imprimir();
    }
}
```

Pregunta 7: Según el texto de la bibliografía básica de la asignatura, indique cuál de las siguientes afirmaciones es correcta:

- a. Un objeto es inmutable si su contenido o su estado no puede ser cambiado una vez que se ha creado.
- b. Un objeto de tipo String puede ser modificado una vez que está creado, por tanto no es un ejemplo de objeto inmutable.
- c. La clase String tiene un método de nombre trim que permite modificar caracteres en cualquier posición de una cadena.
- d. Como regla general, las cadenas de texto de tipo String se suelen comparar mediante el operador "==".

Pregunta 8: Dado el siguiente fragmento de código, indique cuál de las siguientes afirmaciones es el resultado de su ejecución:

```
class Test
{
    public static void main (String args [])
    {
        int n, c = 1, serie = 5;
        System.out.print ("Cantidad de terminos: ");
        n = 7;
        while (c <= n)
        {
            System.out.print (", " + serie);
            serie += 5;
            c++;
        }
    }
}
```

- a. Cantidad de terminos: 5,10,15,20,25,30,
- b. Cantidad de terminos: ,5,10,15,20,25,30
- c. Cantidad de terminos: ,5,10,15,20,25,30,35
- d. Cantidad de terminos: ,5,10,15,20,25,30,35,40

Pregunta 9: Segundo el texto de la bibliografía básica de la asignatura, indique cuál de las siguientes afirmaciones es correcta:

- a. Las colecciones de objetos son objetos que pueden almacenar un número predeterminado e invariable de otros objetos.
- b. Un iterador es un objeto que proporciona funcionalidad para recorrer todos los elementos de una colección.
- c. Un ciclo consiste en la escritura repetida de un bloque de sentencias.
- d. Un arreglo (array) es un tipo especial de colección que puede almacenar un número variable de elementos.

Pregunta 10: Segundo el texto de la bibliografía básica de la asignatura, indique cuál de las siguientes afirmaciones es correcta en relación a la clase Vector de Java:

- a. Es Final
- b. Implementa java.util.List
- c. Es serializable
- d. Dispone de un solo constructor

Pregunta 11: Dado el siguiente fragmento de código, indique cuál de las siguientes afirmaciones es el resultado de su ejecución:

```
public class Test
{
    private int i = getJ();
    private int j = 10;

    private int getJ()
    {
        return j;
    }

    public static void main(String args[])
    {
        System.out.println((new Test()).i);
    }
}
```

- a. Error de compilación en relación a la restricción de acceso de variables privadas en la clase Test.
- b. Error de compilación en relación a las referencias realizadas.
- c. Sin errores de compilación, su salida es 0.
- d. Sin errores de compilación, su salida es 10.

Pregunta 12: Según el texto de la bibliografía básica de la asignatura, indique cuál de las siguientes afirmaciones es correcta:

- La interfaz de una clase describe lo que hace la clase y cómo puede usarse pudiendo mostrar parte de su implementación.
- Un mapa es una colección que almacena entradas de ternas de valores llave/valor/posición.
- La documentación de una clase debe ser suficientemente detallada como para que otros programadores puedan usar la clase sin necesidad de leer su implementación.
- Los modificadores de acceso definen las restricciones de uso de un objeto para determinados métodos, constructores o campos.

Pregunta 13: Dado el siguiente fragmento de código, indique cuál de las siguientes afirmaciones es el resultado de su ejecución:

```
public class Test
{
    public static void main(String args[])
    {
        char c = -1;
        System.out.println(c);
    }
}
```

- La expresión "char c = -1;" provocará un error de compilación debido a que el rango de la clase "char" es 0-2¹⁶⁻¹.
- No habrá error de compilación, la salida será -1.
- No habrá error de compilación, la salida no será ningún carácter ascii.
- No habrá error de compilación, la salida será un carácter Unicode.

Pregunta 14: Según el texto de la bibliografía básica de la asignatura, indique cuál de las siguientes afirmaciones es correcta:

- Una superclase es una clase que es implementada por otra.
- Una subclase es una clase que implementa a otra clase.
- Las clases que están vinculadas mediante una relación de herencia forman una jerarquía de herencia.
- La herencia nos permite heredar pero no reutilizar en un nuevo contexto clases que fueron escritas previamente.

Pregunta 15: Dado el siguiente fragmento de código, indique cuál de las siguientes afirmaciones es el resultado de su ejecución:

```
import java.awt.*;

public class TestFrame extends Frame
{
    public TestFrame()
    {
        setLayout(new GridLayout());
        for(int i = 1 ; i <= 4 ;++i)
        {
            add(new Button(Integer.toString(i)));
        }

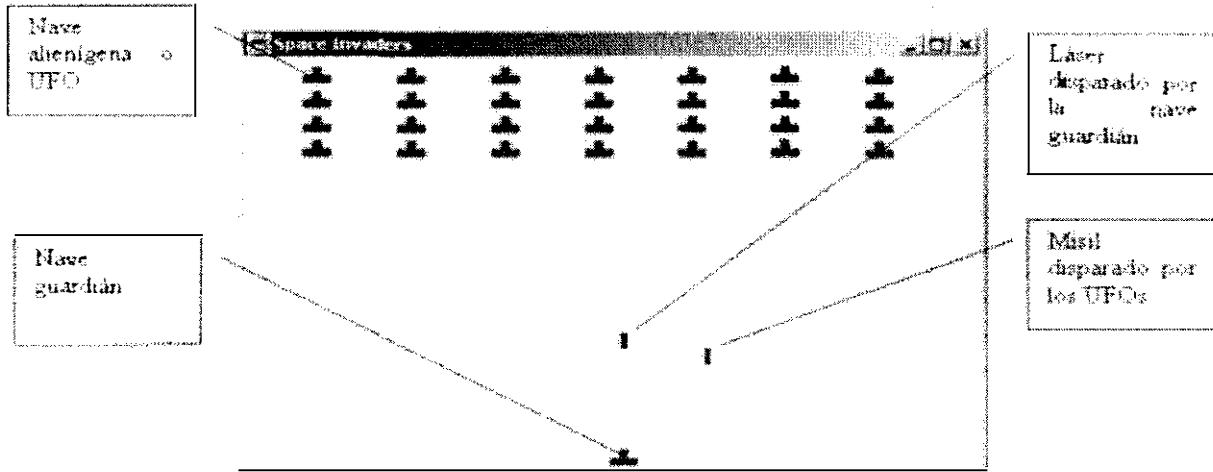
        pack();
        setVisible(true);
    }

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

- a. El código compila, su ejecución provoca que todos los botones aparezcan en una sola columna.
- b. El código compila, su ejecución provoca que todos los botones aparezcan en una sola fila.
- c. El código compila, su ejecución provoca que todos los botones se monten uno encima del otro y tan solo sea visible el último.
- d. El código compila, pero se produce un error en tiempo de ejecución cuando se añaden los componentes.

PARTE PRÁCTICA [6,5 PUNTOS]:

La práctica del presente curso ha sido una versión del legendario arcade “Space Invaders”. A continuación se muestra la propuesta del juego tal y como se solicitaba para la práctica del curso.



En el juego aparecen cuatro clases de elementos (Ver Figura):

1. Naves alienígenas o UFOs, que se mueven de izda. a dcha. y van bajando hacia abajo poco a poco. Esporádicamente lanzan misiles.
 2. La nave guardián es controlada por el jugador.
 3. El láser disparado por la nave guardián (trayectoria ascendente). Cuando el láser de la nave alcanza una nave enemiga, ésta desaparece del juego.
 4. Los misiles disparados por los UFOs (trayectoria descendente). Cuando un misil alcanza a la nave, finaliza el juego.
- a) **[2 puntos]** Diseñar utilizando un paradigma orientado a objetos, los elementos necesarios para la aplicación explicada de la práctica durante el curso. Es necesario identificar la estructura y las relaciones de herencia y de uso de las clases necesarias para almacenar y gestionar esta información. Debe hacerse uso de los mecanismos de herencia siempre que sea posible. Se valorará un buen diseño que favorezca la reutilización de código y facilite su mantenimiento.
 - b) **[1,5 puntos]** Implementa la clase `NaveGuardian`. Especifica sus atributos y métodos y justifica las decisiones de implementación que creas importantes.
 - c) **[1,5 puntos]** Implementa la siguiente regla del juego: “Varias filas de naves alienígenas o UFOs avanzan hacia la base defensora, con movimientos oscilatorios de izquierda a derecha, bajando poco a poco”. Especifica sus atributos y métodos y justifica las decisiones de implementación que creas importantes.
 - d) **[1,5 puntos]** Indique los cambios que serían necesarios en el diseño y programa para permitir que cada nave alienígena se moviera según una trayectoria independiente del resto de naves alienígenas pudiendo moverse libremente en las cuatro coordenadas (arriba, abajo, izquierda y derecha). Las naves alienígenas no podrían ocupar el mismo espacio de coordenadas, por tanto deberían chocar y cambiar su dirección.

**UNIVERSIDAD NACIONAL DE EDUCACIÓN A DISTANCIA – ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA
INFORMÁTICA**

71901072 – PROGRAMACIÓN ORIENTADA A OBJETOS (GRADO EN INGENIERÍA INFORMÁTICA / TECNOLOGÍAS DE LA INFORMACIÓN)

JUNIO 2012 – MODELO B – NO ESTÁ PERMITIDO EL USO DE MATERIAL ADICIONAL

PARTE TEÓRICA - TEST [2,5 PUNTOS]:

Sólo una de las respuestas es válida. Las respuestas correctas se puntuarán con +1.0, mientras que las respondidas de manera incorrecta se puntuarán con -0.25. Las no contestadas no tendrán influencia ni positiva ni negativa en la nota.

Pregunta 1: Según el texto de la bibliografía básica de la asignatura, indique cual de las siguientes afirmaciones es falsa:

- a. Únicamente las clases que implementan la interfaz List permiten el uso de iteradores.
- b. Un iterador es un objeto que proporciona funcionalidad para recorrer todos los elementos de una colección.
- c. Un iterador permite recorrer cualquier tipo de colección hacia adelante utilizando el método next() combinado con el método hasNext() para comprobar si se ha alcanzado el final de la colección.
- d. Una colección puede recorrerse tanto con un iterador como con un ciclo for-each. Ambas formas son equivalentes.

Pregunta 2: Respecto a los bucles, indique cual de las siguientes afirmaciones es falsa:

- a. El cuerpo de un bucle for-each puede repetirse 0 o más veces.
- b. Un bucle for-each puede aplicarse sobre cualquier clase que implemente la interfaz Iterable.
- c. El cuerpo de un bucle while siempre se ejecuta, como mínimo, una vez.
- d. Un bucle for-each puede aplicarse sobre arreglos (arrays).

Pregunta 3: Dado el siguiente código

```
String c1=new String("Hola");
String c2=new String("Mundo");
if (....)
    System.out.println("Ambas cadenas son iguales");
else
    System.out.println("Ambas cadenas no son iguales");
```

¿Cuál de las siguientes opciones debería ponerse en la línea de puntos para llevar a cabo la comparación de las cadenas c1 y c2 en función de la salida proporcionada por el programa?

- a. c1==c2
- b. c1.equals(c2)
- c. c1.compareTo(c2)>=0
- d. c1=c2

Pregunta 4: Indique cual de las siguientes afirmaciones es verdadera:

- a. Para definir una variable de instancia es necesario utilizar la palabra reservada **static**.
- b. Un método estático puede acceder a cualquier componente (método o variable) no estático de su clase.
- c. Los métodos estáticos pueden ser sobreescritos.
- d. Una variable de clase puede ser modificada sin necesidad de haber instanciado objeto alguno de dicha clase.

Pregunta 5: Indique cual de las siguientes afirmaciones es falsa:

- a. El objetivo de la sobrecarga de métodos es facilitar la invocación de un mismo método pasándole un conjunto de parámetros de entrada diferentes.
- b. Se puede sobrecargar un método variando el tipo de retorno de éste sin variar los parámetros de entrada.
- c. Un método puede ser sobrecargado en la misma clase o en una subclase.
- d. Los métodos sobrecargados pueden cambiar el modificador de acceso del método original.

Pregunta 6: Dada la siguiente definición de clase

```
public class TV {  
    private String marca;  
    private String modelo;  
    public TV(String marca, String modelo) {  
        this.marca = marca;  
        this.modelo = modelo;  
    }  
    public boolean equals(Object t) {  
        TV television=(TV)t;  
        return marca.equals(television.marca) &&  
modelo.equals(television.modelo);  
    }  
    public int hashCode() {  
        return marca.length() * 10 + modelo.length();  
    }  
}
```

¿Cuál sería el resultado visualizado al ejecutar el siguiente código en un método main?

```
TV tv1 = new TV("Sony", "Bravia");  
TV tv2 = new TV("Sony", "aivarB");  
  
if(tv1.equals(tv2)) {  
    System.out.println("los televisores son iguales");  
} else {  
    System.out.println("los televisores no son iguales");  
}
```

- a. Los televisores son iguales.
- b. Los televisores no son iguales.
- c. Error de compilación
- d. Error en tiempo de ejecución

Pregunta 7: Dado el siguiente código, indique cual de las siguientes afirmaciones es cierta:

```
public class MiClase {  
    int x = 2;  
    float y = 4.3f;  
    public static void main (String [] args) {  
        for (int z = 1; z < x; z++ )  
            System.out.println("Valor de y="+y);  
    }  
}
```

- a. Se produce un error en tiempo de ejecución.
- b. El código no compila.
- c. Se imprime en pantalla “Valor de y=4.3”
- d. Se imprime en pantalla “Valor de y=4.3000”

Pregunta 8: Sea la siguiente definición de clase:

```
public class ClaseA {  
    public ClaseA(String s) { System.out.print("Construyendo Clase A."); }  
}
```

Y la siguiente definición de una subclase:

```
public class ClaseB extends ClaseA {
```

```

public ClaseB(String s) { System.out.print("Construyendo Clase
B.");super(s); }

public static void main(String [] args) {
    new ClaseB("Objeto Clase B");
    System.out.println(" ");
}
}

```

¿Cuál de las siguientes afirmaciones es cierta al ejecutar el código?

- Se produce un error en tiempo de ejecución.
- Se muestra el mensaje “Construyendo Clase B. Construyendo Clase A.”.
- Se muestra el mensaje “Construyendo Clase A. Construyendo Clase B”.
- Error de compilación.

Pregunta 9: Dado el siguiente código, ¿Cuál de las siguientes afirmaciones es correcta?

```

Set < Object > objetos = new HashSet<Object>();
String obj1 = "JAVA";
int obj2 = 5;
Boolean obj3 = new Boolean(true);
objetos.add(obj3);
objetos.add(obj1);
objetos.add(obj2);
objetos.add(obj3);
for(Object object : objetos) {
System.out.print(object);
}

```

- Error en tiempo de ejecución.
- Se muestran por pantalla JAVA 5 y true en un orden no determinado.
- Se muestran por pantalla JAVA 5 y true en el orden exacto en el que fueron insertadas en la colección.
- Se muestran por pantalla JAVA 5 y true en un orden no determinado y, además, “true” se muestra dos veces.

Pregunta 10: Dados las siguientes definiciones de clases:

```

public abstract class Disparo {
    protected int velocidad=10;
    abstract public void disparar();
}

public class DisparoUFO extends Disparo {
    public void disparar() {
        this.velocidad=20;
        System.out.println("Dispara la nave");
    }
}

public class DisparoNave extends Disparo{
    public void disparar() {
        this.velocidad=10;
        System.out.println("Dispara la nave");
    }
}

public class TestUFO {
    public static void main(String[] args) {
        Disparo dn=new DisparoNave();
        new TestUFO().inicio(dn);
    }
    public void inicio(Disparo d)
    {
        d.disparar();
    }
}

```

```
    }  
}
```

Podemos afirmar:

- a. El método disparar está sobrecargado.
- b. Muestra por pantalla el mensaje “Dispara la nave”.
- c. No se muestra por pantalla ningún mensaje.
- d. Obtenemos un error en tiempo de ejecución.

Pregunta 11: Según el texto de la bibliografía básica de la asignatura, cuando un objeto permite realizar un conjunto de tareas muy relacionadas entre sí, podemos afirmar que:

- a. El objeto presenta una alta cohesión.
- b. El objeto está muy acoplado.
- c. El objeto está poco encapsulado.
- d. El objeto presenta una baja cohesión.

Pregunta 12: Dado el siguiente código:

```
public class ClaseUno {  
    ClaseUno obj;  
    ClaseUno() {}  
    ClaseUno(ClaseUno m) { obj = m; }  
  
    void inicializar() { System.out.print("Inicializando. "); }  
}  
  
public class Test {  
    public static void main(String[] args) {  
        ClaseUno obj1 = new ClaseUno();  
        ClaseUno obj2 = new ClaseUno(obj1);  
        obj2.inicializar();  
        ClaseUno obj3 = obj2.obj;  
        obj3.inicializar();  
        ClaseUno obj4 = obj1.obj;  
        obj4.inicializar();  
    }  
}
```

Podemos afirmar que:

- a. Se mostrará el mensaje “Inicializando. Inicializando. Inicializando.”
- b. Se mostrará el mensaje “Inicializando. Inicializando”.
- c. Se mostrará el mensaje “Inicializando. Inicializando” seguido de una excepción.
- d. Se mostrará el mensaje “Inicializando. Inicializando. Inicializando.” seguido de una excepción

Pregunta 13: Indique cual de las siguientes afirmaciones es cierta:

- a. Una interfaz puede implementar alguno de los métodos que declara.
- b. Una interfaz puede declarar variables de instancia o de clase.
- c. Cuando una clase implementa una interfaz específica no hace falta que implemente todos los métodos que ésta declara.
- d. Una clase puede implementar más de una interfaz al mismo tiempo.

Pregunta 14: Dada la siguiente clase Prueba:

```
public class Prueba {  
    public static void main(String[] args) {  
        ArrayList < Integer > valores = new ArrayList < Integer > ();  
        valores.add(4);  
        valores.add(5);
```

```

        valores.set(1, 6);
        valores.remove(0);
        for(Integer v : valores) {
            System.out.print(v);
        }
    }

}

```

Al ejecutar el código obtendremos:

- a. Un error en tiempo de ejecución
- b. Se mostrará 4
- c. Se mostrará 5
- d. Se mostrará 6

Pregunta 15: Si quisiera organizar los componentes de una interfaz gráfica de acuerdo a una tabla utilizaría como gestor de contenido:

- a. GridLayout
- b. BoxLayout
- c. FlowLayout
- d. BorderLayout

PARTE PRÁCTICA [6,5 PUNTOS]:

La práctica del presente curso ha sido una versión del legendario arcade “Space Invaders”. A continuación se muestra la propuesta del juego tal y como se solicitaba para la práctica del curso. En el juego aparecen cuatro clases de elementos (Ver Figura):

- 1) Naves alienígenas o UFOs, que se mueven de izda. a dcha. y van bajando hacia abajo poco a poco. Esporádicamente lanzan misiles.
- 2) La nave guardián es controlada por el jugador.
- 3) El láser disparado por la nave guardián (trayectoria ascendente). Cuando el láser de la nave alcanza una nave enemiga, ésta desaparece del juego.
- 4) Los misiles disparados por los UFOs (trayectoria descendente). Cuando un misil alcanza a la nave, finaliza el juego.

Se pide diseñar utilizando una aproximación orientada a objetos una ampliación a la práctica realizada a lo largo del curso que permita la existencia de un nuevo tipo de nave alienígena (NaveDeReconocimiento) que se desplace horizontalmente por la parte superior de la pantalla (es decir, por encima del conjunto de UFOs que se van desplazando hacia la nave guardián) y que al llegar a uno de los extremos de la pantalla desaparezca por este y vuelva a aparecer por el extremo opuesto. Para ello el alumno deberá responder de manera razonada a los siguientes apartados:

- a) **[X puntos]** Proponga, utilizando diagramas de clase, y explique como modelaría este nuevo tipo nave alienígena aprovechando el diseño que ha realizado en la práctica. Debe hacerse uso de herencia siempre que sea posible. Se valorará un buen diseño que favorezca la reutilización de código y facilite su mantenimiento.
- b) **[X puntos]** Implemente la nueva clase NaveDeReconocimiento especificando sus atributos y métodos, justificando las decisiones de implementación que considere relevantes.
- c) **[X puntos]** Implemente la regla de juego: “La nave de reconocimiento aparecerá en uno de los extremos de la pantalla y se desplazará hasta el extremo contrario de manera horizontal. Una vez alcanzado el extremo opuesto desaparecerá por éste y volverá a aparecer en el extremo inicial de manera iterativa hasta que sea destruida por un disparo”. En caso de que lo considere necesario puede apoyarse en aquellas clases que ha utilizado en su práctica explicando claramente sus funcionalidades.
- d) **[X puntos]** Explique razonadamente qué cambios serían necesarios en el diseño que ha realizado en los apartados anteriores para que la NaveDeReconocimiento también pudiera realizar disparos.

PARTE TEÓRICA - TEST [2,5 PUNTOS]:

Sólo una de las respuestas es válida. Las respuestas correctas se puntuarán con +1.0, mientras que las respondidas de manera incorrecta se puntuarán con -0.25. Las no contestadas no tendrán influencia ni positiva ni negativa en la nota.

Pregunta 1: ¿Cuál es el resultado de ejecutar el siguiente fragmento de código?

```
if ("String".toString() == "String")
    System.out.println("Igual");
else
    System.out.println("No Igual");
```

- a. El código compilará e imprime “Igual”.
- b. El código compilará e imprime “No Igual”.
- c. El código compilará pero producirá un error de ejecución.
- d. El código no compilará.

Pregunta 2: ¿Cuál es el resultado de ejecutar el siguiente código?

```
public class Ejemplo {
    private int i=j;
    private int j=10;
    public static void main (String []args) {
        System.out.println ((new Ejemplo()).i);
    }
}
```

- a. Da un error de compilación debido a las restricciones de acceso a las variables privadas de Ejemplo.
- b. Da un error de compilación debido a la referencia que se hace a variables declaradas con posterioridad.
- c. No da ningún error de compilación y produce como salida el valor 0.
- d. No da ningún error de compilación y produce como salida el valor 10.

Pregunta 3: ¿Cuál de las siguientes es una característica de la clase java.lang.Exception?

- a. private.
- b. extends Throwable.
- c. implements Throwable.
- d. final.

Pregunta 4: Sean “Mamífero” y “Gato” dos clases que mantienen una relación de herencia padre-hijo. ¿Qué habría que modificar en el siguiente código para que sea correcto y por qué?

```
Animal a;          /* Linea 1 */
Gato b;           /* Linea 2 */
a = new Animal (); /* Linea 3 */
b = a;            /* Linea 4 */
```

- a. Nada. Es correcto.
- b. No se puede asignar un objeto a otro de otra clase, luego cambiamos la línea 2: Animal b; .
- c. Es necesario explicitar el tipo cuando asignamos un objeto a otro objeto perteneciente a una clase hija, luego cambiamos la línea 4: b = (Gato) a; .
- d. Es necesario explicitar el tipo y crear una nueva instancia cuando asignamos un objeto a otro objeto perteneciente a una clase hija, luego cambiamos la línea 4: b = new (Gato) a; .

Pregunta 5: Una variable de clase, definida como static ...

- a. No puede ser accedida desde otra clase.
- b. Si se modifica, lo hace para todas las instancias de la clase.
- c. Es de valor constante.
- d. Solo puede ser accedida desde clases del mismo paquete.

Pregunta 6: En una estructura switch, ¿en qué lugar tiene que ser colocado el bloque de sentencias “default”?

- a. Antes de las diferentes sentencias case.
- b. Despues de todas las sentencias case.
- c. Despues de las sentencias case pero antes de la sentencia finally.
- d. Puede colocarse en el lugar que se quiera.

Pregunta 7: Dada la siguiente instrucción:

x = y--;

¿Cuál de las siguientes afirmaciones es verdadera DESPUÉS de ejecutarse la instrucción?

- a. La instrucción da un error de compilación.
- b. x > y.
- c. x == y.
- d. x < y.

Pregunta 8: ¿Qué ocurre si se compila y ejecuta el siguiente código?

Número de Código

Línea

```
4     public class Clase {  
5         public static void main (String []arguments) {  
6             met (arguments); }  
7         public void met (String []arguments) {  
8             System.out.println(arguments);  
9             System.out.println(arguments[1]); }
```

- a. Da un error de compilación porque no se puede hacer referencia al método no-estático met.
- b. Da un error de compilación porque el método main no puede ser estático.
- c. Da un error de compilación porque el array arguments no puede pasarse como parámetro al método met.
- d. Da un error de ejecución porque en el acceso al array arguments nos salimos del rango de dicho array.

Pregunta 9: ¿Cuál de los siguientes no es un identificador válido en Java?

- a. #variable.
- b. \$variable.
- c. _variable.
- d. vari_able.

Pregunta 10: En la definición de una interface en Java :

- a. Es necesaria emplear la palabra clave abstract.
- b. La signatura de los métodos de una interfaz tienen visibilidad public o private, pero no protected.
- c. No se permiten campos constantes.
- d. Aunque no se indique usando la palabra clave final, todos los campos son tratados como si así fuesen.

Pregunta 11: Una clase interna:

- a. Puede acceder a los campos y métodos públicos y protegidos de la clase envolvente, pero no privados.
- b. Puede acceder a los campos y métodos públicos de la clase envolvente, pero no a privados ni a protegidos.
- c. Puede acceder a los campos y métodos públicos y privados de la clase envolvente.
- d. No puede acceder a los campos y métodos privados de la clase envolvente.

Pregunta 12: Segundo el texto de la bibliografía básica de la asignatura, indique cuál de las siguientes afirmaciones es correcta:

- a. La depuración es la actividad cuyo objetivo es determinar si una pieza de código produce el comportamiento pretendido.
- b. La prueba viene a continuación de la depuración.
- c. La depuración es una actividad dedicada a determinar si un segmento de código contiene errores.
- d. La depuración es el intento de apuntar con precisión y corregir un error en el código.

Pregunta 13: ¿De qué clase deriva la clase ArrayList?

- a. AbstractList.
- b. AbstractCollection.
- c. ArrayCollection.
- d. ListCollection.

Pregunta 14: Cuando queremos que un objeto oiga eventos de acción disparados por el usuario, el objeto tiene que implementar la interfaz ...

- a. ActionEvent.
- b. ActionListener.
- c. ListenerAction.
- d. ListenerEvent.

Pregunta 15: Un conjunto es una:

- a. Que almacena cada elemento individual una sola vez como mínimo. No mantiene un orden específico.
- b. Que almacena cada elemento individual una sola vez como mínimo. Mantiene un orden específico.
- c. Que almacena cada elemento individual una sola vez como máximo. No mantiene un orden específico.
- d. Que almacena cada elemento individual una sola vez como máximo. Mantiene un orden específico.

PARTE PRÁCTICA [6,5 PUNTOS]:

Consideré para su estudio una versión del conocido juego Space Invaders. Este juego consiste en que varias filas de naves alienígenas o UFOs avanzan hacia la base defensora, con movimientos oscilatorios de izquierda a derecha, bajando poco a poco. Así, una nave guardián defiende la base y trata de evitar los misiles lanzados esporádicamente por las naves invasoras. La nave guardián lanza disparos de uno en uno. El juego finaliza cuando todos los invasores han sido alcanzados o cuando los invasores llegan a la base.

- a) **[2,5 puntos]** Suponga que la implementación del juego se hace en base a la existencia de una clase Nave, de tipo abstracto, que sirve de clase de referencia para otras posibles subclases (tanto las naves alienígenas como la nave guardiana). A partir de ésta, se genera una nueva clase denominada NaveUFO que sirve para modelar UNA nave UFO. El juego dispondrá de un total cuatro filas de siete naves UFO cada una de ellas, que se encuentran en la parte superior de la pantalla del juego. Proporcione la estructura de ambas clases (y del bloque completo de naves UFO), así como los diferentes atributos que considere imprescindibles para cada una de ellas y los principales métodos accesores y modificadores. Si se necesita del uso de alguna otra clase auxiliar, debe definirse también en este apartado.
- b) **[2,5 puntos]** Proporcione el método desplazarNavesUFO que simula el movimiento de las naves UFO a derecha e izquierda. El movimiento se realizará cuando el salta un determinado timer (que no hay que implementar, sólo el método que se llama cuando este timer salta). Las naves parten de la zona superior izquierda y se desplazan hasta la parte derecha de la pantalla. Cuando llegan al final, bajan todas las naves una posición y comienzan a desplazarse ahora hacia la izquierda. Se deja a su elección el prototipo que tienen que tener estos métodos, pero han de ser coherentes con lo expuesto en el apartado anterior. Si se necesita del uso de alguna otra clase auxiliar, debe definirse también en este apartado.
- c) **[1,5 puntos]** Proporcione el método disparaMisil, que simula el disparo de un proyectil ascendente por parte de una nave UFO. Se recuerda la restricción de que en un momento determinado sólo puede haber activo un único misil. Se deja a su elección el prototipo que tienen que tener estos métodos, pero han de ser coherentes con lo expuesto en el primer apartado. Si se necesita del uso de alguna otra clase auxiliar, debe definirse también en este apartado.

PARTE TEÓRICA - TEST [2,5 PUNTOS]:

Sólo una de las respuestas es válida. Las respuestas correctas se puntuarán con +1.0, mientras que las respondidas de manera incorrecta se puntuarán con -0.25. Las no contestadas no tendrán influencia ni positiva ni negativa en la nota.

Pregunta 1: ¿Qué incluye como mínimo la descripción de un patrón?

- a. Un nombre, una descripción del problema, una descripción de la solución y las consecuencias del uso del patrón.
- b. Un nombre, una clase, una descripción del problema y las consecuencias del uso del patrón.
- c. Una clase, una descripción del problema, una descripción de la solución y las consecuencias del uso del patrón.
- d. Un nombre, una clase, una descripción de la solución y las consecuencias del uso del patrón.

Pregunta 2: ¿Cuál es el resultado de ejecutar el siguiente código?

```
public class Ejemplo
{
    private int i= dameJ();
    private int j=10;

    private int dameJ ()
    {
        return j;
    }

    public static void main (String []args)
    {
        System.out.println ((new Ejemplo()).i);
    }
}
```

- a. Da un error de compilación debido a las restricciones de acceso a las variables privadas de Ejemplo.
- b. Da un error de compilación debido a la referencia que se hace a métodos declarados con posterioridad.
- c. No da ningún error de compilación y produce como salida el valor 0.
- d. No da ningún error de compilación y produce como salida el valor 10.

Pregunta 3: ¿Cuál de las siguientes instrucciones compila sin provocar un warning o un error?

- a. char c="a";
- b. byte b=257;
- c. boolean b=null;
- d. int i=10;

Pregunta 4: ¿Java permite herencia múltiple?

- a. Sí, es una característica del lenguaje.
- b. No, por definición no admite herencia múltiple y no puede implementarse de modo alguno.
- c. No, pero puede implementarse mediante combinación de “extends” y de “implements interface”.
- d. No, pero puede implementarse mediante combinación de “implements” y de “extends interface”.

Pregunta 5: ¿Qué ocurre si se compila y ejecuta el siguiente código?

```
public class Q {
    public static void main(String argv[])
    {
        int anar[] = new int[5];
        System.out.println(anar[0]);
    }
}
```

- a. Error: anar is referenced before it is initialized
- b. null
- c. 0
- d. 5

Pregunta 6: ¿Cuál de las siguientes afirmaciones es falsa?

- a. System.out.println(-1 >> 2); producirá un resultado mayor que 10.
- b. System.out.println(-1 >> 2); producirá un resultado mayor que cero (positivo).
- c. System.out.println(2 >> 2); producirá el resultado 0.
- d. System.out.println(1 << 2); producirá un resultado 4.

Pregunta 7: ¿Qué se mostrará por pantalla cuando se intenta compilar y ejecutar el siguiente código?

```
import java.awt.*;
public class Butt extends Frame{
    public static void main(String argv[]) {
        Butt MyBut=new Butt();
    }

    Butt() {
        Button HelloBut=new Button("Hola");
        Button ByeBut=new Button("Adios");
        add(HelloBut);
        add(ByeBut);
        setSize(300,300);
        setVisible(true);
    }
}
```

- a. Dos botones juntos ocupando todo el frame. Hola en la izquierda y Adios en la derecha.
- b. Un botón ocupando todo el frame diciendo Hola.
- c. Un botón ocupando todo el frame diciendo Adios.
- d. Dos botones en la parte superior del frame diciendo uno Hola y el otro Adios.

Pregunta 8: Dado el siguiente fragmento, ¿podemos instanciar un objeto de esta clase?

Número de Línea Código

```
4     public abstract class ClaseAbstracta {
5         abstract void MetodoAbstracto(int a);
6     }
```

- a. Sí, creando una nueva clase que extienda a ClaseAbstracta.
- b. Sí, creando una nueva clase que extienda a ClaseAbstracta e implemente el método MetodoAbstracto.
- c. Si, las clases abstractas se pueden instanciar como cualquier otra clase si necesidad de extenderlas si se redefina el método MetodoAbstracto.
- d. Ninguna respuesta anterior es correcta.

Pregunta 9: Según el texto de la bibliografía básica de la asignatura, indique cuál de las siguientes afirmaciones es correcta:

- a. Un encapsulamiento apropiado en las clases reduce el acoplamiento.
- b. El término acoplamiento describe cuánto se ajusta una unidad de código a una tarea lógica o a una entidad.
- c. El acoplamiento describe la conectividad de los propios objetos de una clase.
- d. Un sistema débilmente acoplado se caracteriza por la imposibilidad de modificar una de sus clases sin tener que realizar cambios en ninguna otra.

Pregunta 10: Si se ejecuta el siguiente código, ¿qué se imprime por pantalla?

Número de Línea **Código**

```
4     String s = new String ("Bicycle");
5     int iBegin = 1;
6     char iEnd = 3;
7     System.out.println(s.substring(iBegin,iEnd));
```

- a. Bic.
- b. ic.
- c. icy.
- d. error: no method matching substring(int, char).

Pregunta 11: ¿Cuál es el resultado de la siguiente operación: System.out.println (4 | 3); ?

- a. 6.
- b. 0.
- c. 1.
- d. 7.

Pregunta 12: Necesita crear una clase que almacene como elemento base de la misma objetos únicos. No se necesita que guarden orden alguno, pero sí que no se repitan. ¿Qué interfaz sería la más apropiada para este fin?

- a. Set.
- b. List.
- c. Map.
- d. Vector.

Pregunta 13: Dado el siguiente código, indique qué ocurriría al llamar al método ejemplo();

Número de Línea **Código**

```
4     class Examen {
5         private int i;
6         public void ejemplo () {
7             for (int i=0; i<5; i++)
8                 System.out.print (this.i++);
9         }
10    }
```

- a. Imprime 00000.
- b. Imprime 01234.
- c. Imprime infinitos ceros.
- d. Se producirá un error en tiempo de compilación por no estar inicializada la propiedad i.

Pregunta 14: Dado el siguiente fragmento de código:

Número de Línea **Código**

```
4     String cadena1 = "Halo";
5     String cadena2 = "HALO";
6     cadena1.toUpperCase();
```

¿Cuál sería el resultado de evaluar: if (cadena1.equals(cadena2)) ?

- a. true.
- b. false.
- c. Error en la expresión.
- d. Ninguna de las anteriores es correcta.

Pregunta 15: En lo que se refiere a las clases internas anónimas, se puede afirmar que:

- a. Suelen emplearse en los lugares en los que se requiere la implementación de una sola instancia.
- b. Siempre se hará referencia la instancia mediante su supertipo.
- c. Permiten definir una clase y crear una instancia de ella, todo en un solo paso.
- d. Todas las anteriores son correctas.

PARTE PRÁCTICA [6,5 PUNTOS]:

Considera para su estudio una versión del conocido juego Space Invaders. Este juego consiste en que varias filas de naves alienígenas o UFOs avanzan hacia la base defensora, con movimientos oscilatorios de izquierda a derecha, bajando poco a poco. Así, una nave guardián defiende la base y trata de evitar los misiles lanzados esporádicamente por las naves invasoras. La nave guardián lanza disparos de uno en uno. El juego finaliza cuando todos los invasores han sido alcanzados o cuando los invasores llegan a la base.

- a) **[2,5 puntos]** Suponga que la implementación del juego se hace en base a la existencia de una clase Nave, de tipo abstracto, que sirve de clase de referencia para otras posibles subclases (tanto las naves alienígenas como la nave guardiana). A partir de ésta, se genera una nueva clase denominada NaveGuardian que sirve para modelar la nave guardián que se encuentra en la parte inferior de la pantalla del juego. Proporcione la estructura de ambas clases, así como los diferentes atributos que considere imprescindibles para cada una de ellas y los principales métodos accesores y modificadores. Si se necesita del uso de alguna otra clase auxiliar, debe definirse también en este apartado.
- b) **[2,5 puntos]** Proporcione los métodos desplazarIzquierda y desplazarDerecha que simular el movimiento de la nave a derecha e izquierda. El movimiento se realizará cuando el usuario pulse las teclas 'P' en el caso de la derecha, u 'O' en el caso de la izquierda. Se deja a su elección el prototipo que tienen que tener estos métodos, pero han de ser coherentes con lo expuesto en el apartado anterior. Si se necesita del uso de alguna otra clase auxiliar, debe definirse también en este apartado.
- c) **[1,5 puntos]** Proporcione el método disparaMisil, que simula el disparo de un proyectil ascendente por parte de la nave guardiana al pulsar la tecla Barra Espaciadora. Se recuerda la restricción de que en un momento determinado sólo puede haber activo un único misil. Se deja a su elección el prototipo que tienen que tener estos métodos, pero han de ser coherentes con lo expuesto en el primer apartado. Si se necesita del uso de alguna otra clase auxiliar, debe definirse también en este apartado.

**UNIVERSIDAD NACIONAL DE EDUCACIÓN A DISTANCIA – ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA
INFORMÁTICA**

71901072 – PROGRAMACIÓN ORIENTADA A OBJETOS (GRADO EN INGENIERÍA INFORMÁTICA / TECNOLOGÍAS DE LA INFORMACIÓN)

JUNIO 2013 – MODELO B – NO ESTÁ PERMITIDO EL USO DE MATERIAL ADICIONAL

PARTE TEÓRICA - TEST [2,5 PUNTOS]:

Sólo una de las respuestas es válida. Las respuestas correctas se puntuarán con +1.0, mientras que las respondidas de manera incorrecta se puntuarán con -0.25. Las no contestadas no tendrán influencia ni positiva ni negativa en la nota.

Pregunta 1: Dado el siguiente fragmento de código, cuál es el resultado del comando **java test 2**:

```
public class test {  
    public static void main(String args[]) {  
        Integer intObj=Integer.valueOf(args[args.length-1]);  
        int i = intObj.intValue();  
  
        if(args.length > 1)  
            System.out.println(i);  
        if(args.length > 0)  
            System.out.println(i - 1);  
        else  
            System.out.println(i - 2);  
    }  
}  
  
a. test  
b. test-1  
c. 1  
d. 2
```

Pregunta 2: Dado el siguiente fragmento de código, indique cuál de los siguientes resultados es el resultado de su ejecución:

```
public class test {  
    public static void main(String args[]) {  
        int i, j=1;  
        i = (j>1)?2:1;  
        switch(i) {  
            case 0: System.out.print(0); break;  
            case 1: System.out.print(1);  
            case 2: System.out.print(2); break;  
            case 3: System.out.print(3); break;  
        }  
    }  
  
a. 01  
b. 12  
c. 13  
d. 23
```

Pregunta 3: Según el texto de la bibliografía básica de la asignatura, indique cuál de las siguientes afirmaciones es incorrecta sobre las bibliotecas para la construcción de interfaces gráficas de usuario en Java:

- a. AWT utiliza clases de Swing.
- b. Swing utiliza clases de AWT.
- c. Hay clases equivalentes en AWT y Swing.
- d. Se identifican las clases de Swing con la letra J como prefijo.

Pregunta 4: Según el texto de la bibliografía básica de la asignatura, indique cuáles de las siguientes expresiones resultan verdaderas:

1. $! (4 > 5)$
2. $(1 > 2) \text{ || } ((3 == 3) \text{ && } (2 < 1))$
3. $(3 > 3) \text{ || } (3 == 3) \text{ && } (1 < 2)$
4. $(3 > 3) \text{ || } !((3 == 3) \text{ && } (1 > 0))$
5. $(33 != 33) \text{ && } ! \text{ false}$

- a. Las expresiones 3 y 4.
- b. Las expresiones 2 y 4.
- c. Las expresiones 3 y 5.
- d. Las expresiones 1 y 3.

Pregunta 5: Según el texto de la bibliografía básica de la asignatura, indique cuál de las siguientes afirmaciones es incorrecta:

- a. Las clases definen tipos.
- b. Las diagramas de clases muestran las clases de una aplicación y la relación entre ellos.
- c. Las clases son objetos.
- d. Las clases definen métodos.

Pregunta 6: Dado el siguiente fragmento de código, indique cuál de las siguientes afirmaciones es el resultado de su ejecución:

```
1. class Uno{  
2.     protected Uno yoMismo(){ return this;}  
3. }  
4. class Dos extends Uno{  
5.     public Dos yoMismo(){  
6.         return super.yoMismo();  
7.     }  
8. }
```

- a. No hay errores en el código. El resultado sería una referencia a un objeto del tipo Uno
- b. No hay errores en el código. El resultado sería una referencia a un objeto del tipo Dos
- c. Incompatibilidad de tipos línea 6.
- d. El método yoMismo de la clase Uno no es visible en línea 6.

Pregunta 7: Según el texto de la bibliografía básica de la asignatura, indique cuál de las siguientes afirmaciones es correcta:

- a. El encapsulamiento apropiado en las clases reduce su acoplamiento.
- b. El acoplamiento describe el encapsulamiento de las clases.
- c. El encapsulamiento apropiado en las clases reduce su cohesión.
- d. La cohesión de una unidad de código refleja su acoplamiento.

Pregunta 8: Dado el siguiente fragmento de código, indique cuál de las siguientes afirmaciones es el resultado de su ejecución:

```
public class TestSet {  
    static void add(Set set) {  
        set.add("Hola");  
        set.add(1);  
        System.out.println(set.size());  
    }  
    public static void main(String[] args) {  
        Set<String> set = new HashSet<String>();  
        add(set);  
    }  
}
```

- a. 0
- b. 1
- c. 2
- d. NullPointerException

Pregunta 9: ¿Cuál de las siguientes es un **palabra reservada** en Java?:

- a. NULL
- b. new
- c. instanceof
- d. wend

Pregunta 10: ¿Cuál de las siguientes afirmaciones es correcta sobre la clase Math en Java?:

- a. public class MyCalc extends Math
- b. Math.max(10);
- c. Math.round(9.99,1);
- d. Ninguna de las anteriores.

Pregunta 11: Dado el siguiente fragmento de código, indique cuál de las siguientes afirmaciones es el resultado de su ejecución:

```
public class test {  
    public static void main(String args[]) {  
        int i=1, j=1;  
        try {  
            i++;  
            j--;  
            if(i == j)  
                i++;  
        }  
        catch(ArithmeticException e) {  
            System.out.print(0);  
        }  
        catch(ArrayIndexOutOfBoundsException e) {  
            System.out.print(1);  
        }  
        catch(Exception e) {  
            System.out.print(2);  
        }  
        finally {  
            System.out.print(3);  
        }  
        System.out.print(",4");  
    }  
}
```

- a. 0,4
- b. 1,4
- c. 2,4
- d. 3,4

Pregunta 12: Según el texto de la bibliografía básica de la asignatura, indique cuál de las siguientes definiciones de un método m, que lanza IOException, y que devuelve void, es correcta:

- a. void m() throws IOException {}
- b. void m() throw IOException {}
- c. void m(void) throws IOException {}
- d. void m() {} throws IOException

Pregunta 13: Dado el siguiente fragmento de código, indique cuál es el resultado de su compilación:

```
1. class Parent {  
2.     Double get() {  
3.         return 1.0;  
4.     }  
5. }  
6. class Child extends Parent {  
7.     Integer get() {  
8.         return 2;  
9.     }  
10. }
```

- a. Éxito.
- b. get() en Child no puede extender get() en Parent, tipos del retorno son incompatibles.
- c. get() en Child no puede extender get() en Parent, no son clases públicas.
- d. get() en Child ya definido en Parent.

Pregunta 14: Según el texto de la bibliografía básica de la asignatura, indique cuál de las siguientes afirmaciones es correcta:

- a. Si una clase tiene algún método abstracto hay que declararla como abstracta.
- b. Todos los métodos en una clase abstracta tienen que ser declarados como abstractos.
- c. Una clase que hereda de una clase abstracta no tiene que implementar todos los métodos abstractos para no ser abstracta.
- d. Una clase abstracta no puede implementar ninguna interface.

Pregunta 15: Dado el siguiente fragmento de código, indique cuál es la salida de su compilación/ejecución:

```
1. String nombre = null;  
2. File file = new File("/folder", nombre);  
3. System.out.print(file.exists());
```

- a. true
- b. false
- c. NullPointerException en línea 2.
- d. NullPointerException en línea 3.

PARTE PRÁCTICA [6,5 PUNTOS]:

La práctica del presente curso ha sido una versión del juego “R-Type”. A continuación se presentan las reglas del juego tal y como se solicitaba para la práctica del curso:

1. El juego comenzará con una pantalla de bienvenida a partir de la cual se podrá seleccionar el modo de juego (FACIL, NORMAL, COMPLICADO, IMPOSIBLE) y comenzar a jugar.
2. El juego constará de un único nivel donde el jugador deberá acabar con una horda de naves alienígenas. El número de alienígenas con los que acabar dependerá del modo de juego seleccionado. Fácil=10, Normal=15, Complicado=20, Imposible=30.
3. El jugador controlará la nave aliada y dispondrá de 1 sola vida.
4. Las naves alienígenas serán controladas por el ordenador.
5. Las naves alienígenas no disparan.
6. No hay que implementar relieve. Es decir, no hay que mostrar ningún tipo de suelo o techo como en el juego original.
7. La nave aliada podrá moverse arriba (Tecla Q), abajo (Tecla A), izquierda (Tecla O) y derecha (Tecla P). Así mismo podrá disparar su laser utilizando la tecla ESPACIO.
8. El área de movimiento permitido para la nave será toda la pantalla, aunque habrá que comprobar que la nave no salga de estos límites.
9. El disparo que realiza la nave aliada es continuo, es decir, no es necesario esperar a que el misil disparado abandone la pantalla para que la nave aliada pueda volver a disparar.
10. La nave aliada sólo puede realizar un tipo de disparo que se desplazará horizontalmente hacia la derecha de la pantalla, sin variar su trayectoria y a velocidad constante.
11. Las naves alienígenas se mueven a velocidad constante y podrán ser de dos tipos:

- a. **Nave Alienígena Tipo A.** Aparecen por la parte derecha de la pantalla y se mueven horizontalmente hacia la izquierda a velocidad constante sin variar su trayectoria, es decir, su coordenada "y" no varía en todo el desplazamiento.
 - b. **Nave alienígena Tipo B.** Aparecen por la parte derecha de la pantalla y se mueven horizontalmente hacia la izquierda a velocidad constante. La principal diferencia con las Naves de Tipo A es que éstas pueden variar su trayectoria, es decir, en su desplazamiento horizontal pueden variar su coordenada "y" de manera aleatoria.
12. La velocidad a la que se mueven las naves alienígenas dependerá del modo de juego seleccionado. Todas las naves se mueven a la misma velocidad.
13. Cuando las naves alienígenas alcancen la parte izquierda de la pantalla volverán a aparecer por la parte derecha de ésta.
14. Se deberán de detectar dos tipos de colisiones.
- a. Las colisiones entre la nave aliada y las naves alienígenas, lo que supondrá el final del juego.
 - b. Las colisiones entre los misiles disparados por la nave aliada y las naves alienígenas, lo que supondrá la destrucción de la nave alienígena contra la que ha chocado el misil.
15. Si el jugador finaliza el nivel del juego deberá aparecer un mensaje de felicitación y se volvería a mostrar el menú inicial.
- a) **[1,5 puntos]** Diseñar utilizando un paradigma orientado a objetos, los elementos necesarios para la aplicación explicada de la práctica durante el curso. Es necesario identificar la estructura y las relaciones de herencia y de uso de las clases necesarias para almacenar y gestionar esta información. Debe hacerse uso de los mecanismos de herencia siempre que sea posible. Se valorará un buen diseño que favorezca la reutilización de código y facilite su mantenimiento.
- b) **[1,5 puntos]** Implementa la clase NaveAliada. Especifica sus atributos y métodos y justifica las decisiones de implementación que creas importantes.
- c) **[1,5 puntos]** Implementa la siguiente regla del juego: "Las naves alienígenas Tipo A aparecen por la parte derecha de la pantalla y se mueven horizontalmente hacia la izquierda a velocidad constante sin variar su trayectoria, es decir, su coordenada "y" no varía en todo el desplazamiento". Especifica sus atributos y métodos y justifica las decisiones de implementación que creas importantes.
- a) **[2 puntos]** Indique los cambios que serían necesarios en el diseño y programa para permitir que cada nave alienígena se moviera según una trayectoria independiente del resto de naves alienígenas pudiendo moverse libremente en las cuatro coordenadas (arriba, abajo, izquierda y derecha). Las naves alienígenas no podrían ocupar el mismo espacio de coordenadas, por tanto deberían chocar y cambiar su dirección.

PARTE TEÓRICA - TEST [2,5 PUNTOS]:

Sólo una de las respuestas es válida. Las respuestas correctas se puntuarán con +1.0, mientras que las respondidas de manera incorrecta se puntuarán con -0.25. Las no contestadas no tendrán influencia ni positiva ni negativa en la nota.

Pregunta 1: Segundo el texto de la bibliografía básica de la asignatura, el alcance de una variable

- a. Define la forma en la que la variable puede ser accedida.
- b. Define el conjunto de métodos que puede acceder a la variable.
- c. Define la sección de código en la que la variable puede ser accedida.
- d. Ninguna de las anteriores.

Pregunta 2: Segundo el texto de la bibliografía básica de la asignatura, un prototipo es

- a. Una versión de la aplicación en la que se simula una parte de ella, en vías a experimentar con las restantes partes.
- b. Una versión de la aplicación en la que se simulan varias partes, en vías a experimentar con una de sus partes.
- c. Una versión de la aplicación en la que se simulan varias partes, en vías a experimentar con las restantes partes.
- d. Ninguna de las anteriores

Pregunta 3: Respecto a las excepciones en Java, podemos afirmar

- a. Todas las subclases de la clase estándar de Java RunTimeException son excepciones comprobadas.
- b. Todas las subclases de la clase estándar de Java Exception son excepciones comprobadas.
- c. Error es una subclase directa de Throwable, mientras que Exception es una subclase directa de Error.
- d. Tanto Error como Exception son subclases directas de Throwable.

Pregunta 4: ¿Qué mecanismo usa Java para implementar herencia múltiple?

- a. En Java no se permite la herencia múltiple de clases, ni tampoco la implementación múltiple de interfaces.
- b. En Java no se permite la herencia múltiple de clases, pero sí la implementación múltiple de interfaces.
- c. En Java se permite la herencia múltiple de clases, pero no la implementación múltiple de interfaces.
- d. En Java se permite la herencia múltiple de clases, y también la implementación múltiple de interfaces.

Pregunta 5: ¿Cuál es el valor de la variable d después de ejecutar la siguiente línea de código?

```
double d = Math.round (2.5 + Math.random());
```

- a. 2
- b. 3
- c. 4
- d. 2.5

Pregunta 6: Sea el siguiente código:

```
switch (x) {  
    default:  
        System.out.println("Hola");  
}
```

¿Qué dos posibles tipos son aceptables para x?

1: byte 2: long 3: char 4: float 5: Short 6: Long

- a. 1 y 3.
- b. 2 y 4.
- c. 3 y 5.
- d. 4 y 6.

Pregunta 7: ¿Qué interfaz proporciona la capacidad de almacenar objetos usando un valor llave?

- a. Java.util.Map.
- b. Java.util.Set.
- c. Java.util.List.
- d. Java.util.Collection.

Pregunta 8: ¿Cuál de las siguientes sentencias son correctas?

(1) int w = (int)888.8;
(2) byte x = (byte)1000L;
(3) long y = (byte) 100;
(4) byte z = (byte) 100L;

- a. 1 y 2.
- b. 2 y 3.
- c. 3 y 4.
- d. Todas son correctas.

Pregunta 9: ¿Cuál de las siguientes sentencias declara legalmente, construye e inicializa un array?

- a. int [] miLista = {"1", "2", "3"};
- b. int [] miLista = (5, 8, 2);
- c. int miLista [] [] = {4,9,7,0};
- d. int miLista [] = {4, 3, 7};

Pregunta 10: ¿Cuál de las siguientes listas contiene sólo palabras clave de Java?

- a. class, if, void, long, Int, continue.
- b. goto, instanceof, native, finally, default, throws.
- c. try, virtual, throw, final, volatile, transient.
- d. byte, break, assert, switch, include.

Pregunta 11: ¿Cuál es la salida que produce el siguiente programa?

```
public class Test
{
    public static void leftshift(int i, int j)
    {
        i <= j;
    }
    public static void main(String args[])
    {
        int i = 4, j = 2;
        leftshift(i, j);
        System.out.println(i);
    }
}
```

- a. 2
- b. 4
- c. 8
- d. 16

Pregunta 12: ¿Cuál es la salida del siguiente programa?

```
class Test {
    public static void main(String [] args) {
        Test p = new Test();
        p.start();
    }
}
```

```

void start() {
    boolean b1 = false;
    boolean b2 = fix(b1);
    System.out.println(b1 + " " + b2);
}
boolean fix(boolean b1) {
    b1 = true;
    return b1;
}
}

```

- a. true true
- b. false true
- c. true false
- d. false false

Pregunta 13: ¿Cuál es la salida que se obtiene al ejecutar este programa?

```

1. public class Test {
2.     public int aMethod() {
3.         static int i = 0;
4.         i++;
5.         return i;
6.     }
7.     public static void main(String args[]) {
8.         Test test = new Test();
9.         int i = test.aMethod();
10.        int j = test.aMethod();
11.        System.out.println(j);
12.    }
13. }

```

- a. 0
- b. 1
- c. 2
- d. La compilación falla.

Pregunta 14: ¿Cuál es la salida del siguiente programa?

```

for (int i = 0; i <= 4; i += 2) {
    System.out.print(i + " ");
}
System.out.println(i);

```

- a. 0 2 4
- b. 0 2 4 5
- c. 0 1 2 3 4
- d. La compilación falla.

Pregunta 15: ¿Qué código hay que añadir en la posición indicada en el código para que compile?

```

public class ExceptionTest {
    class TestException extends Exception {}
    public void runTest() throws TestException {}
    public void test() /* Código a añadir */ {
        runTest();
    }
}

```

- a. No hay que añadir código alguno.
- b. throws Exception
- c. catch (Exception e)
- d. throws RuntimeException

PARTE PRÁCTICA [6,5 PUNTOS]:

La práctica del presente curso ha sido una versión del juego **R-Type**. A continuación se presentan las reglas del juego tal y como se solicitaba para la práctica del curso:

1. El juego comenzará con una pantalla de bienvenida a partir de la cual se podrá seleccionar el modo de juego (FACIL, NORMAL, COMPLICADO, IMPOSIBLE) y comenzar a jugar.
2. El juego constará de un único nivel donde el jugador deberá acabar con una horda de naves alienígenas. El número de alienígenas con los que acabar dependerá del modo de juego seleccionado. Fácil=10, Normal=15, Complicado=20, Imposible=30.
3. El jugador controlará la nave aliada y dispondrá de 1 sola vida.
4. Las naves alienígenas serán controladas por el ordenador.
5. Las naves alienígenas no disparan.
6. No hay que implementar relieve. Es decir, no hay que mostrar ningún tipo de suelo o techo como en el juego original.
7. La nave aliada podrá moverse arriba (Tecla Q), abajo (Tecla A), izquierda (Tecla O) y derecha (Tecla P). Así mismo podrá disparar su laser utilizando la tecla ESPACIO.
8. El área de movimiento permitido para la nave será toda la pantalla, aunque habrá que comprobar que la nave no salga de estos límites.
9. El disparo que realiza la nave aliada es continuo, es decir, no es necesario esperar a que el misil disparado abandone la pantalla para que la nave aliada pueda volver a disparar.
10. La nave aliada sólo puede realizar un tipo de disparo que se desplazará horizontalmente hacia la derecha de la pantalla, sin variar su trayectoria y a velocidad constante.
11. Las naves alienígenas se mueven a velocidad constante y podrán ser de dos tipos:
 - a. **Nave Alienígena Tipo A.** Aparecen por la parte derecha de la pantalla y se mueven horizontalmente hacia la izquierda a velocidad constante sin variar su trayectoria, es decir, su coordenada x no varía en todo el desplazamiento.
 - b. **Nave Alienígena Tipo B.** Aparecen por la parte derecha de la pantalla y se mueven horizontalmente hacia la izquierda a velocidad constante. La principal diferencia con las Naves de Tipo A es que éstas pueden variar su trayectoria, es decir, en su desplazamiento horizontal pueden variar su coordenada y de manera aleatoria.
12. La velocidad a la que se mueven las naves alienígenas dependerá del modo de juego seleccionado. Todas las naves se mueven a la misma velocidad.
13. Cuando las naves alienígenas alcancen la parte izquierda de la pantalla volverán a aparecer por la parte derecha de ésta.
14. Se deberán de detectar dos tipos de colisiones.
 - a. Las colisiones entre la nave aliada y las naves alienígenas, lo que supondrá el final del juego.
 - b. Las colisiones entre los misiles disparados por la nave aliada y las naves alienígenas, lo que supondrá la destrucción de la nave alienígena contra la que ha chocado el misil.
15. Si el jugador finaliza el nivel del juego deberá aparecer un mensaje de felicitación y se volvería a mostrar el menú inicial.

Se pide diseñar utilizando una aproximación orientada a objetos una ampliación a la práctica realizada a lo largo del curso que permite la existencia de un nuevo tipo de nave alienígena (Tipo C).

Este nuevo tipo de nave es una modificación de la nave Tipo A, con la particularidad de que, cuando alcanza la parte izquierda, en lugar de aparecer por la parte derecha, lo que hace es desplazarse de izquierda a derecha, hasta que alcanza nuevamente el límite derecho, volviendo entonces a desplazarse hacia la izquierda (y así sucesivamente hasta ser destruida).

En un momento determinado, sólo puede haber una nave de Tipo C en el juego, y no se puede crear otra mientras la anterior de ese tipo no haya sido destruida. Además, como máximo, se podrán generar 2 naves de Tipo C en el modo Fácil; 3 en el modo Normal; 4 en el modo Complicado y 5 en el modo Imposible.

- a) **[1,5 puntos]** Proponga y explique, utilizando diagramas de clase, cómo modelaría este nuevo tipo de nave alienígena aprovechando el diseño que ha realizado en la práctica. Debe hacerse uso de herencia siempre que sea posible. Se valorará un buen diseño que favorezca la reutilización de código y facilite su mantenimiento.
 - b) **[1,5 puntos]** Implemente la nueva clase especificando sus atributos y métodos, justificando las decisiones de implementación que considere relevantes.
 - c) **[2,0 puntos]** Implemente la regla del juego: **La nave alienígena Tipo C se comporta como una nave Tipo A, con la particularidad de que, cuando alcanza la parte izquierda, en lugar de aparecer por la parte derecha, lo que hace es desplazarse de izquierda a derecha, hasta que alcanza nuevamente el límite derecho, volviendo entonces a desplazarse hacia la izquierda (y así sucesivamente hasta ser destruida)**.
- a) **[1,5 puntos]** Explique razonadamente qué cambios serían necesarios en el diseño que ha realizado en los apartados anteriores para que la nave alienígena Tipo C pudiera realizar un número N de idas y venidas (de derecha a izquierda y viceversa) y que cuando pasaran ese número de veces sin ser destruida, entonces se autodestruyera.

PARTE TEÓRICA - TEST [2,5 PUNTOS]:

Sólo una de las respuestas es válida. Las respuestas correctas se puntuarán con +1.0, mientras que las respondidas de manera incorrecta se puntuarán con -0.25. Las no contestadas no tendrán influencia ni positiva ni negativa en la nota.

Pregunta 1: ¿Qué ocurrirá al compilar y ejecutar el siguiente código?

```
class Padre {}  
class ClaseHija extends Padre {}  
class ClaseHija2 extends Padre {}  
public class Test{  
    public static void main(String argv[]){  
        Padre b=new Padre ();  
        ClaseHija s=(ClaseHija) b;  
        System.out.print("Ejecutando Aplicación");  
    }  
}
```

- a. Compilará y se ejecutará sin problemas
- b. Error de Compilación
- c. Excepción en tiempo de ejecución
- d. Excepción en tiempo de ejecución y luego mostrará el mensaje “Ejecutando Aplicación”

Pregunta 2: Según la bibliografía básica, ¿Qué elementos cree que definen a un objeto?

- a. Sus cardinalidad y su tipo
- b. Sus atributos y sus métodos
- c. La forma en que establece comunicación e intercambia mensajes
- d. Su interfaz y los eventos asociados

Pregunta 3: Dada la siguiente definición de clase, ¿Cuál sería el contenido más coherente a implementar en el constructor?

```
class Test {  
    int var;  
    Test(int var) { CONTENIDO CONSTRUCTOR }  
}
```

- a. var=var;
- b. int var=var;
- c. this.var=var;
- d. No se puede llamar igual el parámetro del constructor que el atributo de la clase

Pregunta 4: De acuerdo a la bibliografía básica ¿Qué es el bytecode en Java?

- a. Un formato de intercambio de datos
- b. El formato que obtenemos tras compilar una clase .java
- c. Un tipo de variable
- d. Un depurador de código

Pregunta 5: Dado el siguiente fragmento de programa, indique que afirmación es cierta:

```
int cont;
for (cont=5 ; cont>0 ; cont--)
System.out.print(cont);
System.out.print(cont);
```

- a. Se imprime en pantalla 543210
- b. Se imprime en pantalla 5432100
- c. Se imprime en pantalla 554433221100
- d. Se imprime en pantalla 543210-1

Pregunta 6: Dados los siguientes fragmentos de código ¿Cuál de ellos asociaría a una Interfaz en Java?

- a. public class Componente interface Product
- b. Componente cp = new Componente (interfaz)
- c. public class Componente implements Printable
- d. Componente cp = new Componente.interfaz

Pregunta 7: De acuerdo a la bibliografía básica ¿Qué significa instanciar una clase?

- a. Duplicar una clase
- b. Eliminar una clase
- c. Crear un objeto a partir de la clase
- d. Conectar dos clases entre sí

Pregunta 8: Dado el siguiente código, el resultado será:

```
class MiClase {public int valor; }

class Test {

    public static void main(String[] args) {
        MiClase a1 = new MiClase ();
        MiClase a2 = new MiClase ();
        MiClase a3 = new MiClase ();
        a1.valor=150;
        a2.valor=150;
        a3 = a2;
        if (a1 == a2) { System.out.println(" UNO"); }
        if (a1 == a3) { System.out.println(" DOS"); }
        if (a2 == a3) { System.out.println(" TRES"); }

    }
}
```

- a. UNO
- b. UNO TRES
- c. UNO DOS TRES
- d. TRES

Pregunta 9: Dadas las siguientes definiciones de clases:

```
class ClasePadre{}
class ClaseHija1 extends ClasePadre {}
class ClaseHija2 extends ClasePadre {}
```

y las siguientes instanciaciones:

```
ClasePadre var0 = new ClasePadre();
ClaseHija1 var1 = new ClaseHija1();
ClaseHija2 var2 = new ClaseHija2();
ClasePadre var3 = new ClaseHija1();
ClasePadre var4 = new ClaseHija2();
```

¿Cuál de las asignaciones es válida?

- a. var0 = var1;
- b. var2 = (ClaseHija2)var1;
- c. var2 = var4;
- d. var1 = var2;

Pregunta 10: ¿Qué ocurrirá al compilar y ejecutar el siguiente código?

```
public class MiClase{
    static int variableEstatica;
    public static void main(String argv[]){
        System.out.println(variableEstatica);
    }
}
```

- a. Error en tiempo de ejecución. La variable “variableEstatica” no ha sido inicializada
- b. Se mostrará en pantalla null
- c. Se mostrará en pantalla 1
- d. Se mostrará en pantalla 0

Pregunta 11: De acuerdo a la bibliografía básica ¿Qué significa sobrecargar un método?

- a. Editarlo para modificar su comportamiento
- b. Cambiarle el nombre dejándolo con la misma funcionalidad
- c. Crear un método con el mismo nombre pero diferentes argumentos
- d. Añadirle funcionalidades a un método

Pregunta 12: ¿Qué se mostrará en pantalla al ejecutar el siguiente código?

```
import java.awt.*;
import javax.swing.JFrame;

public class AppBoton extends JFrame{

    public static void main(String argv[]){
        AppBoton MiAppBoton=new AppBoton ();
    }

    public AppBoton(){
        Button boton1=new Button("BOTON 1");
        Button boton2=new Button("BOTON 2");
        add(boton1);
        add(boton2);
        setSize(100,100);
        setVisible(true);
    }
}
```

- a. Dos botones, uno junto a otro ocupando todo el frame. En el botón de la izquierda aparecerá BOTON 1 y en el de la derecha aparecerá BOTON 2.
- b. Un botón ocupando todo el frame con la etiqueta BOTON 1.
- c. Un botón ocupando todo el frame con la etiqueta BOTON 2.
- d. Dos botones en la parte superior del frame, uno de ellos con la etiqueta BOTON 1 y otro de ellos con la etiqueta BOTON 2.

Pregunta 13: De acuerdo a la bibliografía básica, ¿Cuál es la descripción que crees que define mejor el concepto 'clase' en la programación orientada a objetos?

- a. Es un concepto similar al de 'array'
- b. Es un tipo particular de variable
- c. Es un modelo o plantilla a partir de la cual creamos objetos
- d. Es una categoría de datos ordenada secuencialmente

Pregunta 14: De acuerdo a la bibliografía básica, el que una variable en una clase sea estática implica

- a. Hace falta crear un objeto para usarla.
- b. Cualquier objeto de esa clase puede modificar su valor.
- c. Todos los objetos tienen una copia de la variable.
- d. Que es una variable global y se puede usar en cualquier parte de la aplicación.

Pregunta 15: Segundo el código siguiente ¿Qué se visualizará en pantalla?

```
class ClaseA{
    public ClaseA ( int x ){
        System.out.print("ClaseA-" + x);
    }
}

class ClaseB extends ClaseA{
    public ClaseB( ){
        super(6);
        System.out.print(" ClaseB-");
    }
}

public class ClasePrincipal{
    public static void main(String[] args) {
        ClaseB objB1=new ClaseB();
        ClaseB objB2;
        System.out.println(" FIN");
    }
}
```

- a. ClaseA-6 ClaseB- FIN
- b. ClaseB- ClaseA-6 FIN
- c. Hay un error en la clase B. La sentencia "super(6);;" no puede ser la primera en el constructor
- d. Hay un error en la clase ClasePrincipal. Falta el new en "ClaseB objB2;"

PARTE PRÁCTICA [6,5 PUNTOS]:

La práctica del presente curso ha sido una versión del juego "R-Type". A continuación se presentan las reglas del juego tal y como se solicitaba para la práctica del curso:

1. El juego comenzará con una pantalla de bienvenida a partir de la cual se podrá seleccionar el modo de juego (FACIL, NORMAL, COMPLICADO, IMPOSIBLE) y comenzar a jugar.
2. El juego constará de un único nivel donde el jugador deberá acabar con una horda de naves alienígenas. El número de alienígenas con los que acabar dependerá del modo de juego seleccionado. Fácil=10, Normal=15, Complicado=20, Imposible=30.

3. El jugador controlará la nave aliada y dispondrá de 1 sola vida.
4. Las naves alienígenas serán controladas por el ordenador.
5. Las naves alienígenas no disparan.
6. No hay que implementar relieve. Es decir, no hay que mostrar ningún tipo de suelo o techo como en el juego original.
7. La nave aliada podrá moverse arriba (Tecla Q), abajo (Tecla A), izquierda (Tecla O) y derecha (Tecla P). Así mismo podrá disparar su laser utilizando la tecla ESPACIO.
8. El área de movimiento permitido para la nave será toda la pantalla, aunque habrá que comprobar que la nave no salga de estos límites.
9. El disparo que realiza la nave aliada es continuo, es decir, no es necesario esperar a que el misil disparado abandone la pantalla para que la nave aliada pueda volver a disparar.
10. La nave aliada sólo puede realizar un tipo de disparo que se desplazará horizontalmente hacia la derecha de la pantalla, sin variar su trayectoria y a velocidad constante.
11. Las naves alienígenas se mueven a velocidad constante y podrán ser de dos tipos:
 - a. **Nave Alienígena Tipo A.** Aparecen por la parte derecha de la pantalla y se mueven horizontalmente hacia la izquierda a velocidad constante sin variar su trayectoria, es decir, su coordenada "y" no varía en todo el desplazamiento.
 - b. **Nave Alienígena Tipo B.** Aparecen por la parte derecha de la pantalla y se mueven horizontalmente hacia la izquierda a velocidad constante. La principal diferencia con las Naves de Tipo A es que éstas pueden variar su trayectoria, es decir, en su desplazamiento horizontal pueden variar su coordenada "y" de manera aleatoria.
12. La velocidad a la que se mueven las naves alienígenas dependerá del modo de juego seleccionado. Todas las naves se mueven a la misma velocidad.
13. Cuando las naves alienígenas alcancen la parte izquierda de la pantalla volverán a aparecer por la parte derecha de ésta.
14. Se deberán de detectar dos tipos de colisiones.
 - a. Las colisiones entre la nave aliada y las naves alienígenas, lo que supondrá el final del juego.
 - b. Las colisiones entre los misiles disparados por la nave aliada y las naves alienígenas, lo que supondrá la destrucción de la nave alienígena contra la que ha chocado el misil.
15. Si el jugador finaliza el nivel del juego deberá aparecer un mensaje de felicitación y se volvería a mostrar el menú inicial.

Tomando como punto de partida la práctica realizada, se pide implementar algunas mejoras para que la nave aliada disponga de un escudo protector. Este escudo protector podrá activarse pulsando la tecla "E" y, en ese momento, hará que la nave sea inmortal durante 5 segundos. El jugador dispondrá inicialmente de 3 escudos protectores que irán disminuyendo conforme éste los vaya activando.

- a) **[1,5 puntos]** Muestre el diagrama de clases general que ha diseñado en la práctica, explicando claramente la funcionalidad de cada una de las clases y sus métodos asociados.
- b) **[1,5 puntos]** Explicar qué clases de la práctica habría que modificar para materializar los cambios solicitados. Muestre claramente las variables de instancia nuevas que habría que definir y los métodos que habría que implementar, indicando en cada caso a qué clases afectarían.
- c) **[2 puntos]** Implemente las modificaciones propuestas.
- d) **[1,5 puntos]** Explique razonadamente qué cambios serían necesarios en el diseño que ha realizado en los apartados anteriores para que aparezcan aleatoriamente **Bonus de Escudos Protectores** que la nave aliada pueda recoger al pasar por encima de ellos. El efecto de recoger un bonus de escudos protectores será el aumento del número de escudos disponibles para su uso.

**UNIVERSIDAD NACIONAL DE EDUCACIÓN A DISTANCIA – ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA
INFORMÁTICA**

**71901072 – PROGRAMACIÓN ORIENTADA A OBJETOS (GRADO EN INGENIERÍA INFORMÁTICA /
TECNOLOGÍAS DE LA INFORMACIÓN)**

SEPTIEMBRE 2013 – MODELO D – NO ESTÁ PERMITIDO EL USO DE MATERIAL ADICIONAL

PARTE TEÓRICA - TEST [2,5 PUNTOS]:

Sólo una de las respuestas es válida. Las respuestas correctas se puntuarán con +1.0, mientras que las respondidas de manera incorrecta se puntuarán con -0.25. Las no contestadas no tendrán influencia ni positiva ni negativa en la nota.

Pregunta 1: Según el texto de la bibliografía básica de la asignatura, indique cuál de las siguientes afirmaciones es correcta:

- a. La signatura es el encabezado de un método y proporciona la información necesaria para invocarlo.
- b. La signatura está formada por los parámetros de un método y proporciona la información necesaria para invocarlo.
- c. La signatura es el nombre de un método y puede tener parámetros para proporcionar información adicional para realizar una tarea.
- d. La signatura es el encabezado de un método y puede tener parámetros para proporcionar información adicional para realizar una tarea.

Pregunta 2: Dado el siguiente fragmento de código:

```
int indice = 1;
boolean[] examen = new boolean[8];
boolean poo = examen [indice];
```

Indica cual de las siguientes afirmaciones es correcta en relación al valor de la variable poo.

- a. poo tiene el valor 0
- b. poo tiene el valor null
- c. poo tiene el valor false
- d. Se produce una excepción y poo no posee ningún valor

Pregunta 3: Según el texto de la bibliografía básica de la asignatura, indique cuál de las siguientes afirmaciones es correcta:

- a. Los campos se conocen como variables de objeto.
- b. El alcance de una variable define la sección de código desde donde la variable puede ser declarada.
- c. Los constructores permiten que cada objeto sea preparado adecuadamente cuando es creado.
- d. El tiempo de vida de una variable describe el número de veces que es utilizada en un método.

Pregunta 4: Dado el siguiente fragmento de código, indique cuál de las siguientes afirmaciones es el resultado de su ejecución:

```
if(" String ".trim() == "String")
    System.out.println("Igual");
else
    System.out.println("No Igual");
```

- a. El código compilará e imprimirá “Igual”.
- b. El código compilará e imprimirá “No Igual”.
- c. El código provocará un error de compilación.
- d. El código provocará un error en tiempo de ejecución.

Pregunta 5: Según el texto de la bibliografía básica de la asignatura, indique cuál de las siguientes afirmaciones es correcta:

- a. Los campos se definen dentro de los constructores y de los métodos.
- b. Los campos se usan para almacenar datos que nunca persisten durante la vida del objeto.
- c. Los campos tienen un tiempo de vida que perdura después de terminar el objeto.
- d. La accesibilidad de los campos se extiende a toda clase y por este motivo pueden usarse dentro de cualquier constructor o método de clase en la que estén definidos.

Pregunta 6: Según el texto de la bibliografía básica de la asignatura, indique cuales de las siguientes expresiones resultan verdaderas:

1. `! (4 < 5)`
2. `(2 > 2) || ((4 == 4) && (1 < 0))`
3. `(2 > 2) || (4 == 4) && (1 < 0)`
4. `(2 > 2) || !((4 == 4) && (1 < 0))`
5. `(34 != 33) && ! false`

- a. Las expresiones 4 y 5.
- b. Las expresiones 3 y 4.
- c. Las expresiones 2 y 4.
- d. Las expresiones 3 y 5.

Pregunta 7: Según el texto de la bibliografía básica de la asignatura, indique cual de las siguientes afirmaciones es correcta:

- a. Un objeto de tipo String puede ser modificado una vez que está creado, por tanto no es un ejemplo de objeto inmutable
- b. La clase String tiene un método de nombre trim que permite modificar caracteres en cualquier posición de una cadena
- c. Como regla general, las cadenas de texto de tipo String se suelen comparar mediante el operador `==`
- d. Un objeto es inmutable si su contenido o su estado no puede ser cambiado una vez que se ha creado

Pregunta 8: Según el texto de la bibliografía básica de la asignatura, indique cual de las siguientes afirmaciones es correcta:

- a. El término acoplamiento describe cuánto se ajusta una unidad de código a una tarea lógica o a una entidad
- b. El acoplamiento describe la conectividad de los propios objetos de una clase
- c. Un encapsulamiento apropiado en las clases reduce el acoplamiento
- d. Un sistema debilmente acoplado se caracteriza por la imposibilidad de modificar una de sus clases sin tener que realizar cambios en ninguna otra

Pregunta 9: Dado el siguiente fragmento de código que pretende mostrar un ejemplo de sobrescritura:

```
class Examen {  
    private float pregunta = 1.0f ;  
    protected float getNota () {return pregunta;}  
}  
  
class Test extends Examen {  
    private float nota = 2.0f;  
    //Insertar código aquí  
}
```

Indique cual de las siguientes opciones completaría el código anterior para dar lugar a un ejemplo correcto de sobrescritura:

- a. public float getNota (float valor) { return valor;}
- b. public float getNota () { return nota;}
- c. float getNota () { return nota;}
- d. float double getNota () { return nota;}

Pregunta 10: Según el texto de la bibliografía básica de la asignatura, indique cual de las siguientes afirmaciones es correcta:

- a. Un mapa es una colección que almacena pares llave/valor como entradas.
- b. Un mapa es una colección que almacena tríos llave/índice/valor como entradas.
- c. Un mapa es una colección que almacena pares índice/valor como entradas.
- d. Un mapa es una colección que almacena tríos índice/posición/valor como entradas.

Pregunta 11: Dado el siguiente fragmento de código, indique cuál de las siguientes afirmaciones es el resultado de su ejecución:

```
public class Examen {  
  
    public static void main(String args[])  
    {  
        char nota = -1;  
        System.out.println(nota);  
    }  
  
}
```

- a. No habrá error de compilación, la salida será -1.
- b. La expresión “char nota = -1;” provocará un error de compilación debido a que el rango de la clase “char” es $0-2^{(128-1)}$.
- c. La expresión “char nota = -1;” provocará un error de compilación debido a que el rango de la clase “char” es $0-2^{(16-1)}$.
- d. No habrá error de compilación, la salida será un carácter Unicode.

Pregunta 12: Según el texto de la bibliografía básica de la asignatura, indique cual de las siguientes opciones declarará un método en una clase que fuerza a una subclase a implementarlo:

- a. protected void metodoPI (double d1){}
- b. abstract public void metodoPI ();
- c. static void metodoPI (double d1) {}
- d. public native double metodoPI ();

Pregunta 13: Según el texto de la bibliografía básica de la asignatura, indique cuál de las siguientes afirmaciones es correcta:

- a. Una superclase es una clase que es implementada por otra.
- b. Una subclase es una clase que implementa a otra clase.
- c. La herencia nos permite heredar pero no reutilizar en un nuevo contexto clases que fueron escritas previamente.
- d. Las clases que están vinculadas mediante una relación de herencia forman una jerarquía de herencia.

Pregunta 14: En el siguiente fragmento de código hemos definido la ejecución de cinco bloques. Estos bloques se ejecutarán dependiendo de las excepciones que se produzcan en cada caso.

```
// Bloque1
try{
    // Bloque2
} catch (ArithmException e) {
    // Bloque3
} finally{
    // Bloque4
}
// Bloque5
```

Indique cual de las siguientes afirmaciones es correcta:

- a. El Bloque4 no se ejecutará si se produce una excepción de tipo aritmético en el Bloque2
- b. El Bloque4 se ejecutará antes de que la excepción producida por un acceso a un objeto *null* en el Bloque2 se propague hacia arriba
- c. El Bloque4 no se ejecutará si se produce un acceso a un objeto *null* en el Bloque2
- d. El Bloque4 se ejecutará antes que el Bloque3 si se produce una excepción de tipo aritmético en el Bloque2

Pregunta 15: Según el texto de la bibliografía básica de la asignatura, indique cuál de las siguientes afirmaciones es correcta:

- a. En Javadoc la etiqueta @param indica el número de parámetros del método
- b. En Javadoc la etiqueta @deprecated indica que el valor devuelto por el método puede contener errores
- c. En Javadoc la etiqueta @see indica una referencia cruzada
- d. En Javadoc la etiqueta @throws indica el modo en que debe ser lanzado un método

PARTE PRÁCTICA [6,5 PUNTOS]:

La práctica del presente curso ha sido una versión del juego “R-Type”. A continuación se presentan las reglas del juego tal y como se solicitaba para la práctica del curso:

1. El juego comenzará con una pantalla de bienvenida a partir de la cual se podrá seleccionar el modo de juego (FACIL, NORMAL, COMPLICADO, IMPOSIBLE) y comenzar a jugar.
2. El juego constará de un único nivel donde el jugador deberá acabar con una horda de naves alienígenas. El número de alienígenas con los que acabar dependerá del modo de juego seleccionado. Fácil=10, Normal=15, Complicado=20, Imposible=30.
3. El jugador controlará la nave aliada y dispondrá de 1 sola vida.
4. Las naves alienígenas serán controladas por el ordenador.
5. Las naves alienígenas no disparan.
6. No hay que implementar relieve. Es decir, no hay que mostrar ningún tipo de suelo o techo como en el juego original.
7. La nave aliada podrá moverse arriba (Tecla Q), abajo (Tecla A), izquierda (Tecla O) y derecha (Tecla P). Así mismo podrá disparar su laser utilizando la tecla ESPACIO.
8. El área de movimiento permitido para la nave será toda la pantalla, aunque habrá que comprobar que la nave no salga de estos límites.
9. El disparo que realiza la nave aliada es continuo, es decir, no es necesario esperar a que el misil disparado abandone la pantalla para que la nave aliada pueda volver a disparar.
10. La nave aliada sólo puede realizar un tipo de disparo que se desplazará horizontalmente hacia la derecha de la pantalla, sin variar su trayectoria y a velocidad constante.
11. Las naves alienígenas se mueven a velocidad constante y podrán ser de dos tipos:
 - a. **Nave Alienígena Tipo A.** Aparecen por la parte derecha de la pantalla y se mueven horizontalmente hacia la izquierda a velocidad constante sin variar su trayectoria, es decir, su coordenada “y” no varía en todo el desplazamiento.
 - b. **Nave Alienígena Tipo B.** Aparecen por la parte derecha de la pantalla y se mueven horizontalmente hacia la izquierda a velocidad constante. La principal diferencia con las Naves de Tipo A es que éstas pueden variar su trayectoria, es decir, en su desplazamiento horizontal pueden variar su coordenada “y” de manera aleatoria.
12. La velocidad a la que se mueven las naves alienígenas dependerá del modo de juego seleccionado. Todas las naves se mueven a la misma velocidad.
13. Cuando las naves alienígenas alcancen la parte izquierda de la pantalla volverán a aparecer por la parte derecha de ésta.
14. Se deberán de detectar dos tipos de colisiones.
 - a. Las colisiones entre la nave aliada y las naves alienígenas, lo que supondrá el final del juego.
 - b. Las colisiones entre los misiles disparados por la nave aliada y las naves alienígenas, lo que supondrá la destrucción de la nave alienígena contra la que ha chocado el misil.
15. Si el jugador finaliza el nivel del juego deberá aparecer un mensaje de felicitación y se volvería a mostrar el menú inicial.

Se pide diseñar utilizando una aproximación orientada a objetos una ampliación a la práctica realizada a lo largo del curso que permite la existencia de un nuevo tipo de nave alienígena (Tipo C).

Este nuevo tipo de nave es una modificación de la nave Tipo A, con la particularidad de que, cuando alcanza la parte izquierda, en lugar de aparecer por la parte derecha, lo que hace es desplazarse de izquierda a derecha, hasta que alcanza nuevamente el límite derecho, volviendo entonces a desplazarse hacia la izquierda (y así sucesivamente hasta ser destruida).

En un momento determinado, sólo puede haber una nave de Tipo C en el juego, y no se puede crear otra mientras la anterior de ese tipo no haya sido destruida. Además, como máximo, se podrán generar 2 naves de Tipo C en el modo Fácil; 3 en el modo Normal; 4 en el modo Complicado y 5 en el modo Imposible.

Se pide:

- a) **[1,5 puntos]** Diseñar utilizando un paradigma orientado a objetos, los elementos necesarios para la aplicación explicada de la práctica durante el curso. Es necesario identificar la estructura y las relaciones de herencia y de uso de las clases necesarias para almacenar y gestionar esta información. Debe hacerse uso de los mecanismos de herencia siempre que sea posible. Se valorará un buen diseño que favorezca la reutilización de código y facilite su mantenimiento.
- a) **[1,5 puntos]** Implementa la clase NaveAlienigenaTipoB. Especifica sus atributos y métodos y justifica las decisiones de implementación que creas importantes.
- b) **[1,5 puntos]** Implementa un método que gestione las colisiones entre los misiles disparados por la nave aliada y las naves alienígenas.
- c) **[2 puntos]** Indique los cambios que serían necesarios en el diseño y programa para permitir que las naves alienígenas de Tipo B pudieran variar su velocidad de desplazamiento a medida que se incrementa el número de naves alienígenas destruidas. La velocidad se determina en el momento de aparición de la nave en la pantalla y por tanto podría darse el caso de que en la pantalla co-existiesen naves alienígenas con velocidades diferentes.

UNIVERSIDAD NACIONAL DE EDUCACIÓN A DISTANCIA – ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA INFORMÁTICA
71901072 – PROGRAMACIÓN ORIENTADA A OBJETOS (GRADO EN INGENIERÍA INFORMÁTICA / TECNOLOGÍAS DE LA
INFORMACIÓN)

JUNIO 2014 – MODELO A – **NO ESTÁ PERMITIDO EL USO DE MATERIAL ADICIONAL**

PARTE TEÓRICA - TEST [2,5 PUNTOS]:

Solo una de las respuestas es válida. Las respuestas correctas se puntuarán con +1.0, mientras que las respondidas de manera incorrecta se puntuarán con -0.25. Las no contestadas no tendrán influencia ni positiva ni negativa en la nota.

Pregunta 1: Un patrón de diseño ...

- a. proporciona una descripción de un problema común sin dar ningún detalles de implementación.
- b. no se aplica en resolución de problemas de orientación a objetos.
- c. proporciona una descripción de un pequeño conjunto de clases que ayuda a resolver un problema.
- d. no puede implementarse en Java

Pregunta 2: Respecto a las clases internas ...

- a. Las instancias de la clase interna no están necesariamente asociadas a instancias de la clase circundante.
- b. No se consideran una parte de la clase circundante.
- c. No pueden acceder a los métodos privados de la clase circundante.
- d. Presentan un acoplamiento muy estrecho con la clase circundante.

Pregunta 3: Un método cohesionado ...

- a. Será responsable de al menos una tarea bien definida, pero puede serlo de más.
- b. Será responsable de una y sólo una tarea bien definida.
- c. Es aquel método abstracto que se ha instanciado en una clase determinada.
- d. Es aquel que se crea en una clase interna para ser invocado desde la clase circundante.

Pregunta 4: Supongamos que reescribimos el ejemplo BouncingBall del libro de la forma en que se muestra a continuación:

```
1  public class BouncingBall {  
2      int n;  
3      public static void main (String args [])  
4      {  
5          if (n!=0)  
6          {  
7              n = n + 1;  
8              System.out.println("El número es " + n);  
9          }  
10     }  
11 }
```

¿Cuál es la línea que provoca que el código produzca un error de compilación?

- a. Línea 5.
- b. Línea 3.
- c. Línea 2.
- d. Línea 1.

Pregunta 5: Sea el siguiente fragmento de código de código:

```
1  Random randomGenerator;  
2  randomGenerator = new Random();  
3  int index = randomGenerator.nextInt();  
4  System.out.println(index);
```

¿En qué línea del código anterior se produce un error de compilación?

- a. No se produce error de compilación
- b. En la línea 1
- c. En la línea 2
- d. En la línea 3

Pregunta 6: Sea el siguiente fragmento de código modificado de la clase `MailItem` mostrada en el libro de texto:

```
1  public class MailItem {  
2      static int num1 = 10;  
3      public static void main (String args []) {  
4          int num2 = 5;  
5          new MailItem ();  
6      }  
7      public MailItem () {  
8          int aux = this.num2;  
9          if (aux > 1) {  
10              System.out.println(aux);  
11          }  
12      }  
13  }
```

¿Cuál es el resultado que produce?

- a. Se produce un error de compilación.
- b. Se produce un error de ejecución.
- c. No produce ningún error pero no muestra nada por pantalla.
- d. No se produce ningún error y muestra por pantalla el valor 5.

Pregunta 7: ¿Cuál de las siguientes sentencias se ejecuta de manera correcta?

- a. String matriz [] = {"Coche", "Avión", "Tren"};
- b. String matriz = {"Coche", "Avión", "Tren"};
- c. String matriz [] = new String {"Coche" "Avión" "Tren"};
- d. String matriz [] = { "Coche" "Avión" "Tren"};

Pregunta 8: Dado que un elemento `Button` puede propiciar el lanzamiento de un `ActionEvent`, ¿qué tipo de listener habría que implementar en la clase que quiera gestionar este evento?

- a. WindowListener
- b. ActionListener
- c. ComponentListener
- d. PushListener

Pregunta 9: ¿En qué condiciones puede volverse a invocar un constructor de una clase para un objeto después de que ese objeto haya sido creado?

- a. Cuando queremos resetear todos los campos del objeto a sus valores iniciales.
- b. Cuando se ha creado un objeto abstracto y se le quiere dar valores iniciales a sus atributos.
- c. Cuando se implementa una interfaz para el objeto en cuestión.
- d. Nunca.

Pregunta 10: Sea el siguiente código modificado de la clase `MusicOrganizer` mostrada en el libro base:

```
1  import java.util.*;  
2  public class MusicOrganizer {
```

```

3     public static void main (String args []) {
4         ArrayList <String> a = new ArrayList (5);
5         for (int i=0; i<=5; i++)
6         {
7             a.add("Hola");
8         }
9         System.out.println("Funciona");
10    }
11 }
```

¿Cuál es el resultado de compilar y ejecutar este código?

- a. Se produce un error de ejecución al definir un ArrayList de 5 elementos y querer insertar 6 elementos.
- b. No se produce ningún tipo de error y proporciona el resultado por pantalla “Funciona”.
- c. La línea 4 provoca un warning pero se ejecuta sin problemas proporcionando el resultado por pantalla “Funciona”.
- d. La línea 7 provoca un warning pero se ejecuta sin problemas proporcionando el resultado por pantalla “Funciona”.

Pregunta 11: ¿Cómo se llama el entorno de pruebas que soporta la prueba estructurada de unidades y las pruebas de regresión en Java?

- a. JDK
- b. JBoss
- c. Javadoc
- d. JUnit

Pregunta 12: Respecto al constructor de la subclase ...

- a. Debe siempre invocar al constructor de su superclase como primera instrucción. Si no incluye esta llamada, Java intentará insertar una llamada automáticamente.
- b. No debe invocar nunca al constructor de su superclase como primera instrucción. Si la incluye esta llamada, Java ignorará esta llamada automáticamente.
- c. Debe siempre invocar al constructor de su superclase como última instrucción. Si no incluye esta llamada, Java intentará insertar una llamada automáticamente.
- d. Debe siempre invocar al constructor de su superclase como última instrucción. Si no incluye esta llamada, Java generará un error de compilación.

Pregunta 13: Respecto a las variables polimórficas ...

- a. Toda variable de objeto en Java es potencialmente polimórfica.
- b. Son aquellas que exclusivamente pertenecen a clases abstractas.
- c. Son la instancia de una clase abstracta, permitiendo sólo almacenar objetos de ese tipo.
- d. Son aquellas que implementan una interfaz y que provienen de una clase abstracta.

Pregunta 14: Si una clase B extiende una clase abstracta A que tiene un método abstracto met, ¿qué podemos afirmar?

- a. Que necesariamente B es abstracta.
- b. Que si B implementa el método met, entonces seguro que B no es abstracta.
- c. Que no puedo crear instancias de A.
- d. Que puedo crear instancias de A.

Pregunta 15: Se define como excepción comprobada ...

- a. Aquellas subclases de la clase estándar RunnertimeException

- b. Aquellas subclases de la clase estándar RunneableTimeException
- c. Aquellas subclases de la clase estándar RunningtimeException
- d. Aquellas subclases de la clase estándar RuntimeException

PARTE PRÁCTICA [6,5 PUNTOS]:

La práctica del presente curso ha sido una versión del legendario arcade "Pac-Man". A continuación se muestra la propuesta del juego tal y como se solicitaba para la práctica del curso.

1. El juego constará de un solo nivel donde el jugador deberá comer todos los puntos de la pantalla.
 2. El jugador controlará a Pac-Man y dispondrá de 1 vida.
 3. Los fantasmas serán controlados por el ordenador teniendo en cuenta el comportamiento diferente de cada uno.
 4. Pac-Man podrá moverse (Utilizando las flechas del teclado) arriba (Tecla Up), abajo (Tecla Down), izquierda (Tecla Left) y derecha (Tecla Right). Así mismo podrá pausar el juego pulsando la tecla "P".
 5. El área de movimiento permitido para Pac-Man y los fantasmas será el mapa del único nivel disponible.
 6. Será necesario comprobar que tanto Pac-Man como los fantasmas no superen los límites del mapa.
 7. Los caminos del mapa solo permiten el paso de un individuo al mismo tiempo, por tanto habrá que tener en cuenta las colisiones.
 8. Los fantasmas deben implementar comportamientos diferentes:
 - a. Blinky, el fantasma rojo, buscará colisionar con Pac-Man. Para acercarse a Pac-Man calculará la distancia (por ejemplo medido en filas y columnas) e intentará primero acercarse verticalmente y luego horizontalmente.
 - b. Pinky. Buscará colisionar con Pac-Man. Para acercarse a Pac-Man calculará la distancia (por ejemplo medido en filas y columnas) e intentará primero acercarse horizontalmente y luego verticalmente.
 - c. Clyde. Él no persigue a Pac-Man, si no que deambula sin una ruta específica.
 9. Se deberán de detectar dos tipos de colisiones.
 - a. Las colisiones entre Pac-Man y los fantasmas, lo que supondrá la pérdida de una vida o el final del juego en caso de no disponer de más vidas.
 - b. Las colisiones entre los fantasmas, que supondrá un cambio de dirección en los fantasmas involucrados.
 10. Habrá cuatro puntos más grandes de lo normal situados cerca de las esquinas del laberinto y proporcionarán a Pac-Man la habilidad temporal (5 segundos) de comerse a los fantasmas (todos ellos se vuelven azules mientras Pac-Man tiene esa habilidad). Después de haber sido tragados, los fantasmas se regeneran en "casa de fantasmas".
 11. Será necesario implementar un contador con los puntos obtenidos en cada momento, teniendo en cuenta los objetos comidos. Un punto pequeño supone 10 puntos. Comer un fantasma 100 puntos.
 12. Si el jugador finaliza el nivel del juego deberá aparecer un mensaje de felicitación y se volvería a mostrar la página inicial.
- a) **[2 puntos]** Diseñar utilizando un paradigma orientado a objetos, los elementos necesarios para la aplicación explicada de la práctica durante el curso. Es necesario identificar la estructura y las relaciones de herencia y de uso de las clases necesarias para almacenar y gestionar esta

información. Debe hacerse uso de los mecanismos de herencia siempre que sea posible. Se valorará un buen diseño que favorezca la reutilización de código y facilite su mantenimiento.

- b) **[1,5 puntos]** Implementa la clase FantasmaSusto. Este fantasma lo que hace es huir del Pac-Man, de tal manera que mide la distancia que hay entre Pac-Man y él, y se mueve arriba, izquierda, derecha o abajo en función de si la distancia que consigue con ese desplazamiento es el mayor alejamiento posible. Especifica sus atributos y métodos y justifica las decisiones de implementación que creas importantes. Como se ha dicho, para alejarse de Pac-Man calculará la distancia según el criterio que se considere oportuno (por ejemplo, medido en filas y columnas).
- c) **[1,5 puntos]** Implementa un método que gestione el efecto "terremoto". El terremoto consiste en que el sistema coloca automáticamente todos los fantasmas en posiciones aleatorias del tablero, y a Pac-Man lo coloca en el centro de todos los fantasmas. Se debe especificar cómo se calcula la posición donde se coloca Pac-Man (a definir por el alumno).
- d) **[1,5 puntos]** Indique los cambios que serían necesarios en el diseño y programa para permitir que conforme pasa el tiempo, los fantasmas vayan aumentando su velocidad de movimiento hasta que Pac-Man se come un punto grande, momento en el que todos los fantasmas vuelven a restaurarse a la velocidad de movimiento inicial.

PARTE TEÓRICA - TEST [2,5 PUNTOS]:

Solo una de las respuestas es válida. Las respuestas correctas se puntuarán con +1.0, mientras que las respondidas de manera incorrecta se puntuarán con -0.25. Las no contestadas no tendrán influencia ni positiva ni negativa en la nota.

Pregunta 1: Si la primera parte de una clase TicketMachine tuviera la siguiente estructura:

```
public class TicketMachine {  
    private String nombre = "ACME";  
    private String registro = "0000";  
  
    public TicketMachine(String registro) {  
        this.registro = registro;  
    }  
  
    public TicketMachine(String nombre) {  
        this.nombre = nombre;  
    }  
  
    public static void main (String [] args){  
        String nombre = "maquina1";  
        TicketMachine tm = new TicketMachine(nombre);  
    }  
}
```

¿Qué pasaría al ejecutar el método main?

- a. tm.nombre tendría el valor de "maquina1".
- b. tm.registro tendría el valor de "maquina1".
- c. El programa daría un error de ejecución.
- d. El programa daría un error de compilación.

Pregunta 2: Se dice que un objeto es inmutable si:

- a. Su contenido o estado cambia después de su creación.
- b. Su contenido o estado no puede cambiarse después de su creación.
- c. Existiría más que una copia de su contenido o estado después de su creación.
- d. Su contenido o estado es visible fuera de la clase en la que está definido.

Pregunta 3: Hay una clase MessagePost que hereda de otra Post. Si ambas clases tuvieran la siguiente estructura:

```
class Post {  
    public String mensaje = "En Post";  
  
    public void enviarMensaje(){  
        System.out.println(mensaje);  
    }  
}  
  
public class MessagePost extends Post {  
    public String mensaje = "En MessagePost";  
  
    public void enviarMensaje(){  
        System.out.println(mensaje);  
    }  
}
```

```

        public static void main(String args[ ]) {
            Post p = new MessagePost();

            System.out.print(p.mensaje + " ");
            p.enviarMensaje();
        }
    }
}

```

¿Cuál sería el resultado de ejecutar el método main?:

- a. En Post En Post
- b. En MessagePost En Post
- c. En Post En MessagePost
- d. En MessagePost En MessagePost

Pregunta 4: ¿En BlueJ, cómo se pueden ver los métodos que tiene una librería del sistema como java.lang.String?:

- a. En el menú 'Edit(Editar)' hay una entrada para manejar las clases en la librería.
- b. En el menú 'View(Vista)' hay una entrada para manejar las clases en la librería.
- c. En el menú 'Tools(Herramientas)' hay una entrada para manejar las clases en la librería.
- d. No se puede hacer en BlueJ.

Pregunta 5: En una simulación de los zorros y los conejos se puede definir una clase abstracta Animal. En una versión de la simulación, el código podría ser:

```

import java.util.List;
abstract class Animal {
    String nombre = "Animal";

    abstract public void act(List<Animal> newAnimals);
}

class Zorro extends Animal {
    String nombre = "Zorro";
    public String nombreAnimal(){
        return(nombre);
    }
}

public class ZorrosConejos {
    public static void main(String[] args) {
        Zorro z = new Zorro();
        System.out.println(z.nombreAnimal());
    }
}

```

¿Cuál sería el resultado de ejecutar el método main?:

- a. Animal
- b. Zorro
- c. Un error de compilación
- d. Un error de ejecución

Pregunta 6: ¿Cuál sería la signatura de un método público suma que tenga un parámetro que es un array de int y que devuelva un int?:

- a. public int suma(int numeros)
- b. public int suma(int[] números[])
- c. public int[] suma(int numeros)
- d. public int suma(int[] numeros)

Pregunta 7: Para anticipar las excepciones existe la instrucción try. Dado el siguiente código para extraer la extensión de un archivo:

```
public class Archivo {  
    public void tipoExtension(String nombre) {  
        try {  
            String ext = nombre.substring(nombre.indexOf('.'),  
                nombre.length());  
            System.out.println(ext);  
        } catch (StringIndexOutOfBoundsException ex) {  
            System.out.print("Archivo no tiene extensión");  
        } catch (ArithmException ex) {  
            System.out.print("Error aritmético");  
        } catch (NullPointerException ex) {  
            System.out.print("Error del puntero");  
        } finally {  
            System.out.print(". En clausula finally");  
        }  
        System.out.print(". Después del try");  
    }  
  
    public static void main(String[] args) {  
        Archivo ae = new Archivo();  
        ae.tipoExtension("foo");  
    }  
}
```

¿Cuál sería el resultado de ejecutar el método main?:

- a. Archivo no tiene extensión. En clausula finally. Despues del try
- b. Error aritmético. En clausula finally. Despues del try
- c. Error del puntero. En clausula finally. Despues del try
- d. En clausula finally. Despues del try

Pregunta 8: En un reloj digital la clase ClockDisplay gestiona las horas y los minutos. Se podría añadir un método alarma para comprobar para activar un despertador:

```
private final String alarma = "11:00";  
public void alarma(String hora){  
    System.out.println("Hora == Alarma is:" + hora == alarma);  
}  
  
public static void main(String[] args) {  
    Archivo ae = new Archivo();  
    ae.alarma("11:00");  
}
```

¿Cuál sería el resultado de ejecutar el método main()?:

- a. Hora == Alarma is:
- b. Hora == Alarma is: 11:00
- c. Hora == Alarma is: 11:00 false
- d. false

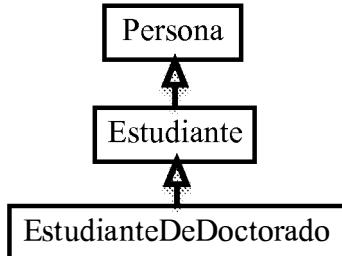
Pregunta 9: En la práctica de este año hay que crear un tablero para el juego Pacman. Para ello se puede dibujar una matriz de cuadros negros, dejando otros en blanco. ¿Cuál de los siguientes métodos nos permite dibujar un cuadrado negro entero en la pantalla?:

- a. fillRect(int x, int y, int anchura, int altura);
- b. fillRect(int anchura, int altura);
- c. fillCuadro(int x, int y, int anchura, int altura);
- d. fillCuadro(int anchura, int altura);

Pregunta 10: ¿Qué es un banco de pruebas?:

- a. Uno o más objetos que se emplean en más de una prueba.
- b. Uno o más objetos que se emplean en una sola prueba.
- c. Uno o más objetos que se emplean para encontrar errores sintácticos en el código.
- d. Ninguno de los anteriores.

Pregunta 11: Dada la siguiente jerarquía de clases:



¿Cuáles de las siguientes asignaciones serían legales?:

- 1. Persona p1 = new Estudiante();
 - 2. Persona p2 = new EstudianteDeDoctorado();
 - 3. EstudianteDeDoctorado edd = new Estudiante();
 - 4. Estudiante e1 = new EstudianteDeDoctorado();
-
- a. 1,2,3,4
 - b. 1,2,3
 - c. 2,3,4
 - d. 1,2,4

Pregunta 12: ¿Existen varios modelos para la construcción de software. ¿Cuáles son dos de los más conocidos?:

- a. Modelo en cascada y modelo de análisis
- b. Modelo en cascada y desarrollo iterativo
- c. Modelo en cascada y prueba incremental
- d. Desarrollo iterativo y prueba incremental

Pregunta 13: En el organizador de música podemos usar un ArrayList para guardar los nombres de las canciones:

```
import java.util.ArrayList;
public class OrganizadorMusica {

    ArrayList<String> canciones = new ArrayList<String>();
    public static void main(String[] args) {
        OrganizadorMusica mo = new OrganizadorMusica();
        mo.canciones.add("Al amanecer");
        System.out.println("¿Existe la canción?: " + mo.existe("Al amanecer"));
    }

    public boolean existe(String cancion) {
        XXX
        if(titulo.equals(cancion))
            return(true);
    }
    return(false);
}
```

}

¿Qué habrá que poner en vez de los XXX para que al ejecutar main el programa produzca el resultado “Existe la canción?: true”?

- a. while (String titulo: canciones) {
- b. for (String titulo in canciones) {
- c. for (String titulo: canciones) {
- d. for (int i=0; i < canciones.length; i++) { String titulo = canciones.get(i);

Pregunta 14: En las diferentes versiones de un proyecto del juego zuul se pueden plantear diferentes versiones del método getExitString; por ejemplo:

<pre>public String getExitString() { String returnString = "Exits:"; if (northExit != null) returnString += "north "; if (eastExit != null) returnString += "east "; if (westExit != null) returnString += "west "; if (southExit != null) returnString += "south "; return returnString; }</pre>	<pre>public String getExitString() { String returnString = "Exits:"; Set<String> keys = exits.keySet(); for(String exit : keys) { returnString += " " + exit; } return returnString; }</pre>
Versión A	Versión B

¿Cuál de las dos versiones muestra más acoplamiento?

- a. A
- b. B
- c. Son iguales
- d. No muestran acoplamiento ninguno

Pregunta 15: En un visor de imágenes se usa botones para que el usuario pueda cambiar el tamaño de la imagen; por ejemplo:

```
smallerButton = new JButton("Smaller");
smallerButton.addActionListener(XXX {
    public void actionPerformed(ActionEvent e) { makeSmaller(); }
});
toolbar.add(smallerButton);
```

¿Qué tipo de Listener habrá que usar (donde están los XXX en el código anterior) para detectar que el usuario ha hecho clic en el botón?

- a. new EventListener()
- b. new ButtonListener()
- c. new ActionListener()
- d. new ActionPerformedListener()

PARTE PRÁCTICA [6,5 PUNTOS]:

La práctica del presente curso ha sido una versión del legendario arcade “Pac-Man”. A continuación se muestra la propuesta del juego tal y como se solicitaba para la práctica del curso.

- 1- El juego constará de un solo nivel donde el jugador deberá comer todos los puntos de la

pantalla.

- 2- El jugador controlará a Pac-Man y dispondrá de 1 vida.
- 3- Los fantasmas serán controlados por el ordenador teniendo en cuenta el comportamiento diferente de cada uno.
- 4- Pac-Man podrá moverse (Utilizando las flechas del teclado) arriba (Tecla Up), abajo (Tecla Down), izquierda (Tecla Left) y derecha (Tecla Right). Así mismo podrá pausar el juego pulsando la tecla "P".
- 5- El área de movimiento permitido para Pac-Man y los fantasmas será el mapa del único nivel disponible.
- 6- Será necesario comprobar que tanto Pac-Man como los fantasmas no superen los límites del mapa.
- 7- Los caminos del mapa solo permiten el paso de un individuo al mismo tiempo, por tanto habrá que tener en cuenta las colisiones.
- 8- Los fantasmas deben implementar comportamientos diferentes:
 - a. Blinky, el fantasma rojo, buscará colisionar con Pac-Man. Para acercarse a Pac-Man calculará la distancia (por ejemplo medido en filas y columnas) e intentará primero acercarse verticalmente y luego horizontalmente.
 - b. Pinky. Buscará colisionar con Pac-Man. Para acercarse a Pac-Man calculará la distancia (por ejemplo medido en filas y columnas) e intentará primero acercarse horizontalmente y luego verticalmente.
 - c. Clyde. Él no persigue a Pac-Man, si no que deambula sin una ruta específica.
- 9- Se deberán de detectar dos tipos de colisiones.
 - a. Las colisiones entre Pac-Man y los fantasmas, lo que supondrá la pérdida de una vida o el final del juego en caso de no disponer de más vidas.
 - b. Las colisiones entre los fantasmas, que supondrá un cambio de dirección en los fantasmas involucrados.
- 10- Habrá cuatro puntos más grandes de lo normal situados cerca de las esquinas del laberinto y proporcionarán a Pac-Man la habilidad temporal (5 segundos) de comerse a los fantasmas (todos ellos se vuelven azules mientras Pac-Man tiene esa habilidad). Después de haber sido tragados, los fantasmas se regeneran cada uno en una esquina del laberinto.
- 11- Será necesario implementar un contador con los puntos obtenidos en cada momento, teniendo en cuenta los objetos comidos. Un punto pequeño supone 10 puntos. Comer un fantasma 100 puntos.
- 12- Si el jugador finaliza el nivel del juego deberá aparecer un mensaje de felicitación y se volvería a mostrar la página inicial.
 - a) **[2 puntos]** Diseñar utilizando un paradigma orientado a objetos, los elementos necesarios para la aplicación explicada de la práctica durante el curso. Es necesario identificar la estructura y las relaciones de herencia (mediante el uso de un diagrama de clases) y de uso de las clases necesarias para almacenar y gestionar esta información. Debe hacerse uso de los mecanismos de herencia siempre que sea posible. Se valorará un buen diseño que favorezca la reutilización de código y facilite su mantenimiento.
 - b) **[1,5 puntos]** Implementa la clase FantasmaPinky. Especifica sus atributos y métodos y justifica las decisiones de implementación que creas importantes. Recuerda que este

fantasma buscará colisionar con Pac-Man. Para acercarse a Pac-Man calculará la distancia (por ejemplo, medido en filas y columnas) e intentará primero acercarse horizontalmente y luego verticalmente.

- c) **[1,5 puntos]** Implementa un método que gestione las dos tipos de colisiones que puede haber entre Pac-Man y los fantasmas, lo que supondrá la pérdida de una vida o el final del juego en caso de no disponer de más vidas, y las colisiones entre los fantasmas, que supondrán un cambio de dirección en los fantasmas involucrados.
- d) **[1,5 puntos]** Indique los cambios que serían necesarios en el diseño y la implementación para permitir que haya diferentes niveles y que en cada uno de éstos se añada un nuevo tipo de fantasma (además de los que ya existen).

PARTE TEÓRICA - TEST [2,5 PUNTOS]:

Solo una de las respuestas es válida. Las respuestas correctas se puntuarán con +1.0, mientras que las respondidas de manera incorrecta se puntuarán con -0.25. Las no contestadas no tendrán influencia ni positiva ni negativa en la nota.

Pregunta 1: Indica cual de las siguientes declaraciones es válida para el método main:

- a. public static void main(String args[]);
- b. static public void main(String);
- c. public static void main(String);
- d. public static int main(String args[]);

Pregunta 2: Indica cual de las siguientes afirmaciones es correcta:

- a. Los métodos de modificación no cambian el estado de un objeto.
- b. Las sentencias de asignación almacenan el valor representado por el lado derecho de la sentencia en una variable nombrada a la izquierda.
- c. El alcance de una variable define la sección de un método en la que la variable puede ser accedida.
- d. Los métodos de acceso devuelven información sobre el estado de una instancia.

Pregunta 3: Indica cual de las siguientes afirmaciones es correcta:

- a. Un depurador es una herramienta de software que ayuda a examinar cómo compila una aplicación.
- b. Una llamada a método interno consiste en que los métodos pueden llamar a otros métodos de la misma clase como parte de su implementación.
- c. Una llamada a método externo consiste en que los métodos pueden llamar a métodos de otras clases abstractas usando la notación de punto.
- d. Los objetos pueden crear otros objetos usando el operador “instanceof”.

Pregunta 4: Supongamos que queremos implementar una Agenda, ¿cuál sería la salida del siguiente código?

```
public class Agenda {  
  
    public static void main(String argv[]){  
        Agenda agenda = new Agenda();  
    }  
  
    protected Agenda(){  
        for(int i=0; i<10; i++){  
            System.out.println(i);  
        }  
    }  
}
```

- a. Error de Compilación ya que los constructores no pueden ser declarados como “protected”.
- b. Error en tiempo de ejecución ya que los constructores no pueden ser declarados como “protected”.
- c. Compilación correcta y salida de los dígitos de 0 a 10.
- d. Compilación correcta y salida de los dígitos de 0 a 9.

Pregunta 5: Indica cual de las siguientes afirmaciones es correcta:

- a. El lenguaje Java tiene tres tipos de ciclo: while, while-do y for.
- b. Un ciclo while es similar en su estructura y propósito que el ciclo for-each.
- c. El tipo de la variable de ciclo no tiene porqué ser el mismo que el tipo del elemento declarado para la colección que estamos recorriendo con un ciclo.
- d. Un índice es un objeto que proporciona funcionalidad para recorrer todos los elementos de una colección.

Pregunta 6: Dado el siguiente fragmento de código, indique cuál de las siguientes afirmaciones es el resultado de su ejecución:

```
if(" Problema ".trim().toLowerCase() == "problema")
    System.out.println("Igual");
else
    System.out.println("No Igual");
```

- a. El código provocará un error de compilación.
- b. El código provocará un error en tiempo de ejecución.
- c. El código compilará e imprimirá “Igual”.
- d. El código compilará e imprimirá “No Igual”.

Pregunta 7: Indica cual de las siguientes afirmaciones es correcta:

- a. La prueba es la actividad de descubrir si una pieza de código produce el comportamiento pretendido.
- b. Una aserción es una expresión que establece una condición que esperamos que resulte verdadera.
- c. Un seguimiento es la actividad de trabajar a través de un segmento de código línea por línea, mientras se observan cambios de estado y otros comportamientos de la aplicación.
- d. Todas las respuestas anteriores son correctas.

Pregunta 8: Indica cual de las siguientes afirmaciones es correcta:

- a. El acoplamiento describe la conectividad de los propios objetos de una clase
- b. Un sistema débilmente acoplado se caracteriza por la imposibilidad de modificar una de sus clases sin tener que realizar cambios en ninguna otra
- c. Un encapsulamiento apropiado en las clases reduce el acoplamiento
- d. El término acoplamiento describe cuánto se ajusta una unidad de código a una tarea lógica o a una entidad

Pregunta 9: Basado en el ejemplo de la Base de Datos de CDs y DVDs visto en la asignatura en el capítulo 8, ¿cuál sería la salida del siguiente código?

```
public class BaseDeDatos {

    public final void metodoAregarElemento(){
        System.out.println("Aregar Elemento");
    }
}

public class BaseDeDatosDeMusica {

    public static void main(String argv[ ]){
        BaseDeDatos db = new BaseDeDatos();
```

```

        db.metodoAgregarElemento();
    }
}

```

- a. Error en tiempo de compilación indicando que una clase con métodos finales deben ser declarada también como final.
- b. Error en tiempo de compilación indicando que no se puede heredar de una clase con métodos finales.
- c. Error en tiempo de ejecución indicando que BaseDeDatos no ha sido definida como final.
- d. Éxito en la compilación y salida “ Agregar Elemento”.

Pregunta 10: Indica cual de las siguientes afirmaciones es correcta:

- a. La declaración de un campo o de un método como “protected” permite el acceso directo al mismo desde las subclases (solo directas).
- b. Las llamadas a métodos en Java permite que la misma llamada a un método en diferentes momentos pueda invocar diferentes métodos, dependiendo del tipo dinámico del parámetro de retorno a la hora de hacer la invocación.
- c. La llamada a “super” en un determinado método (que no sea un constructor) tiene que ocurrir en su primera sentencia dentro de dicho método.
- d. Ninguna de las anteriores.

Pregunta 11: Indica cual de las siguientes afirmaciones es correcta:

- a. Todos los métodos de una interfaz son abstractos.
- b. Las interfaces no contienen ningún constructor.
- c. En una interfaz sólo se permiten los campos constantes.
- d. Todas las respuestas anteriores son correctas.

Pregunta 12: Dado un visor de imágenes, ¿Cuál sería la salida del siguiente código?

```

import java.awt.*;
public class Pulsador extends Frame{
    public static void main(String argv[]){
        Pulsador MiPulsador=new Pulsador();
    }

    Pulsador(){
        Button BotonHola=new Button("Hola");
        Button BotosAdios=new Button("Adios");
        add(BotonHola);
        add(BotosAdios);
        setSize(300,300);
        setVisible(true);
    }
}

```

- a. Dos botones uno al lado del otro ocupando todo el marco, “Hola” en la izquierda y “Adios” en la derecha.
- b. Dos botones uno encima del otro diciendo, “Hola” arriba y “Adios” abajo.
- c. Un solo botón ocupando el marco entero diciendo “Hola”.
- d. Un solo botón ocupando el marco entero diciendo “Adios”.

Pregunta 13: Teniendo en cuenta el modelo en cascada presente en la construcción del software, indica cual de las siguientes fases NO pertenece al desarrollo de software:

- a. Análisis del problema.
- b. Prueba Unitaria.
- c. Prueba Secuencial.
- d. Entrega del sistema al cliente.

Pregunta 14: ¿Cuál sería la salida del siguiente código?

```
int i=1;
switch (i) {
case 0:
System.out.print("cero ");
break;
case 1:
System.out.print("uno ");
case 2:
System.out.print("dos ");
break;
default:
System.out.print("otro ");
}
```

- a. uno
- b. uno otro
- c. uno dos
- d. uno dos otro

Pregunta 15: Indica cual de las siguientes afirmaciones es correcta:

- a. El proceso de autoboxing se lleva a cabo automáticamente cuando se usa un valor de un tipo no primitivo en un contexto que requiere un tipo objeto.
- b. Los objetos subtipo pueden usarse cada vez que se espera un supertipo. Esto se conoce como supertipación.
- c. Las clases que están vinculadas mediante una relación de herencia forman una jerarquía de herencia.
- d. Todas las respuestas anteriores son falsas.

PARTE PRÁCTICA [6,5 PUNTOS]:

La práctica del presente curso ha sido una versión del legendario arcade “Pac-Man”. A continuación se muestra la propuesta del juego tal y como se solicitaba para la práctica del curso.

- 1- El juego constará de un solo nivel donde el jugador deberá comer todos los puntos de la pantalla.
- 2- El jugador controlará a Pac-Man y dispondrá de 1 vida.
- 3- Los fantasmas serán controlados por el ordenador teniendo en cuenta el comportamiento diferente de cada uno.
- 4- Pac-Man podrá moverse (Utilizando las flechas del teclado) arriba (Tecla Up), abajo (Tecla Down), izquierda (Tecla Left) y derecha (Tecla Right). Así mismo podrá pausar el juego pulsando la tecla “P”.
- 5- El área de movimiento permitido para Pac-Man y los fantasmas será el mapa del único nivel disponible.

- 6- Será necesario comprobar que tanto Pac-Man como los fantasmas no superen los límites del mapa.
 - 7- Los caminos del mapa solo permiten el paso de un individuo al mismo tiempo, por tanto habrá que tener en cuenta las colisiones.
 - 8- Los fantasmas deben implementar comportamientos diferentes:
 - a. Blinky, el fantasma rojo, buscará colisionar con Pac-Man. Para acercarse a Pac-Man calculará la distancia (por ejemplo medido en filas y columnas) e intentará primero acercarse verticalmente y luego horizontalmente.
 - b. Pinky. Buscará colisionar con Pac-Man. Para acercarse a Pac-Man calculará la distancia (por ejemplo medido en filas y columnas) e intentará primero acercarse horizontalmente y luego verticalmente.
 - c. Clyde. Él no persigue a Pac-Man, si no que deambula sin una ruta específica.
 - 9- Se deberán de detectar dos tipos de colisiones.
 - a. Las colisiones entre Pac-Man y los fantasmas, lo que supondrá la pérdida de una vida o el final del juego en caso de no disponer de más vidas.
 - b. Las colisiones entre los fantasmas, que supondrá un cambio de dirección en los fantasmas involucrados.
 - 10- Habrá cuatro puntos más grandes de lo normal situados cerca de las esquinas del laberinto y proporcionarán a Pac-Man la habilidad temporal (5 segundos) de comerse a los fantasmas (todos ellos se vuelven azules mientras Pac-Man tiene esa habilidad). Después de haber sido tragados, los fantasmas se regeneran cada uno en una esquina del laberinto.
 - 11- Será necesario implementar un contador con los puntos obtenidos en cada momento, teniendo en cuenta los objetos comidos. Un punto pequeño supone 10 puntos. Comer un fantasma 100 puntos.
 - 12- Si el jugador finaliza el nivel del juego deberá aparecer un mensaje de felicitación y se volvería a mostrar la página inicial.
- a) **[2 puntos]** Diseñar utilizando un paradigma orientado a objetos, los elementos necesarios para la aplicación explicada de la práctica durante el curso. Es necesario identificar la estructura y las relaciones de herencia (mediante el uso de un diagrama de clases) y de uso de las clases necesarias para almacenar y gestionar esta información. Debe hacerse uso de los mecanismos de herencia siempre que sea posible. Se valorará un buen diseño que favorezca la reutilización de código y facilite su mantenimiento.
- b) **[1,5 puntos]** Implementa la clase FantasmaMinky. Este fantasma buscará colisionar con Pac-Man. Para acercarse a Pac-Man rodeará los obstáculos al contrario de las manecillas del reloj y aumentará su velocidad después de que un cierto número de puntos sean comidos (por ejemplo cada 25 puntos aumentará su velocidad). Especifica sus atributos y métodos y justifica las decisiones de implementación que creas importantes así como los cambios en las clases que creas necesario.
- c) **[1,5 puntos]** Implementa un método que permita la existencia de cuatro puntos más grandes de lo normal presentes en las esquinas del laberinto y que proporcionarían a Pac-Man la habilidad temporal (10 segundos) de comerse a los fantasmas. Los fantasmas se volverían azules durante este periodo de tiempo y después, en caso de haber sido comidos aparecerían de nuevo cada uno por una esquina diferente.
- d) **[1,5 puntos]** Indique los cambios que serían necesarios en el diseño y programa para permitir que hubiera un nuevo tipo de Fantasma denominado "Invlnky" que tuviera

velocidad constante, tuviera un movimiento errático sin la intención de colisionar con PacMan y que además tuviera dos tipos de colisiones:

- a. En caso de colisionar con los otros Fantasmas, tanto “InvlNky” como el otro fantasma involucrado, cambiarían de sentido alejándose uno del otro.
- b. En el caso de PacMan, no habría colisión sino que para PacMan sería como un objeto invisible que no le causa ningún tipo de daño ni cambio en su comportamiento.

PARTE TEÓRICA - TEST [2,5 PUNTOS]:

Sólo una de las respuestas es válida. Las respuestas correctas se puntuarán con +1.0, mientras que las respondidas de manera incorrecta se puntuarán con -0.25. Las no contestadas no tendrán influencia ni positiva ni negativa en la nota.

Pregunta 1: Un método de acceso o selector:

- a. Habitualmente devuelve void.
- b. Devuelve siempre información sobre el estado de un objeto.
- c. Devuelve siempre un objeto de la clase Object.
- d. Permite acceder al constructor de la clase que lo define.

Pregunta 2: Un método de modificación o mutador:

- a. Habitualmente devuelve void.
- b. Devuelve siempre información sobre el estado de un objeto.
- c. Permite modificar el estado únicamente de los campos públicos de la clase.
- d. Permite acceder al constructor de la clase que lo define.

Pregunta 3: Dada la siguiente definición de clase:

```
public class MusicOrganizer
{
    private int valor;

    public MusicOrganizer (int n)
    {
        valor=n;
    }

    public int calcular()
    {
        int resultado=1;
        int numero=valor;

        if (valor>=1)
        {
            while(numero>=1){
                resultado*=numero;
                numero--;
            }
            return resultado;
        }
        else{
            return 1;
        }
    }

    public static void main (String [] args){
        MusicOrganizer t=new MusicOrganizer(4);
        int v=t.calcular();
```

```

        System.out.println(v);
    }
}

```

El resultado de su ejecución sería:

- a. 24
- b. 6
- c. 120
- d. 0

Pregunta 4: Cual de las siguientes afirmaciones es falsa:

- a. La documentación de la librería de clases de Java muestra detalles acerca de todas las clases de la librería.
- b. La interfaz de una clase describe lo que una clase hace y cómo se puede utilizar sin mostrar su implementación.
- c. Los modificadores de acceso definen la visibilidad sólo de los campos.
- d. Las clases pueden tener campos. Estos se conocen con el nombre de variables de clase o variables estáticas.

Pregunta 5: Indique cual de las siguientes afirmaciones no es correcta respecto al uso de la herencia en JAVA:

- a. Evita el tener que declarar constructores
- b. Evita la duplicación de código
- c. Facilita la reutilización del código
- d. Facilita la ampliabilidad y mantenimiento del código

Pregunta 6: Se ha visto en la asignatura una clase MessagePost que hereda de otra Post. Si ambas clases tuvieran la siguiente estructura:

```

class Post {
    public String mensaje = "En Post";

    public void enviarMensaje(){
        System.out.println(mensaje);
    }
}

public class MessagePost extends Post {
    public String mensaje = "En MessagePost";

    public void enviarMensaje(){
        System.out.println(mensaje);
    }

    public static void main(String args[]) {
        MessagePost p = new MessagePost();
        Post post=p;
        post.enviarMensaje();
    }
}

```

Cual sería el resultado de ejecutar el método main:

- a. En Post
- b. En MessagePost En Post
- c. En MessagePost
- d. En MessagePost En MessagePost

Pregunta 7: Las pruebas de regresión se definen como:

- La ejecución de las pruebas pasadas previamente para asegurarse de que la nueva versión aún las pasa.
- La ejecución de pruebas automatizadas aleatorias sobre los distintos valores que puede recibir la clase evaluada.
- La aplicación sistemática del conjunto de casos de prueba base que se definieron justo al comenzar con el desarrollo de la aplicación y que no varían nunca a lo largo de éste.
- El conjunto de pruebas negativas necesarias para demostrar que la clase evaluada falla.

Pregunta 8: En una simulación de los zorros y los conejos se puede definir una clase abstracta Animal. En una versión modificada de la simulación el código podría ser:

```
import java.util.List;
abstract class Animal {
    String nombre = "Animal";

    abstract public void metodo(List<Animal> newAnimals);
}

class Zorro extends Animal {
    String nombre = "Zorro";

    public String nombreAnimal(){
        return(nombre);
    }

    public void metodo(List<Animal> newAnimals){

        System.out.println("Animal");
    }
}

public class ZorrosConejos {
    public static void main(String[] args) {
        Animal z = new Zorro();
        System.out.println(z.nombreAnimal());
    }
}
```

Cual sería el resultado de ejecutar el método main:

- Animal
- Zorro
- Un error de compilación
- Un error en tiempo de ejecución

Pregunta 9: Dado el código de la clase MusicOrganizer. ¿Cuál sería el resultado de la ejecución del método main?

```
import java.util.ArrayList;

public class MusicOrganizer
{

    private ArrayList<String> files;

    public MusicOrganizer()
    {files = new ArrayList<String>();}
}
```

```

public void addFile(String filename)
{files.add(filename);}

public int getNumberOfFiles()
{return files.size();}

public void listFile(int index)
{
    if(index >= 0 && index < files.size()) {
        String filename = files.get(index);
        System.out.println(filename);
    }
}

public void removeFile(int index)
{
    if(index >= 0 && index < files.size()) {
        files.remove(index);
    }
}

public class Test
{
    public static void main(String[] args) {
        MusicOrganizer mo=new MusicOrganizer();

        mo.addFile("Disco 1");
        mo.addFile("Disco 2");
        mo.addFile("Disco 3");

        mo.listFile(1);
        mo.removeFile(1);
        mo.listFile(1);
    }
}

```

- a. Disco 2 null
- b. Error en tiempo de ejecución
- c. Disco 1 Disco 2
- d. Disco 2 Disco 3

Pregunta 10: Dado el siguiente fragmento de código, podemos afirmar que la salida del programa:

```

import java.util.Random;

public class Test
{
    public static void main(String[] args) {

        Random generadorAleatorios;
        generadorAleatorios=new Random();

        for(int n=0;n<=100;n++){
            System.out.println(generadorAleatorios.nextInt(n+1));
        }
    }
}

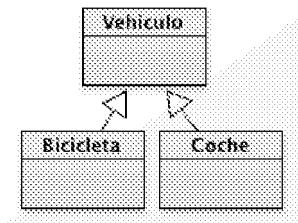
```

- a. Generará valores aleatorios entre 0 y n en cada vuelta del bucle.
- b. Generará valores aleatorios entre 0 y n+1 en cada vuelta del bucle.
- c. Generará valores aleatorios entre 1 y n en cada vuelta del bucle.
- d. Generará valores aleatorios entre 1 y n+1 en cada vuelta del bucle.

Pregunta 11: Indique cual de las siguientes afirmaciones es falsa en relación al desarrollo de la práctica obligatoria:

- El método actionPerformed es el encargado de actualizar las posiciones y el estado del juego en cada intervalo de tiempo.
- La vista del juego se implementa sobre un JPanel sobreescribiendo su método paint().
- Para detectar las pulsaciones de teclado en el juego podemos hacer uso de la clase abstracta KeyAdapter.
- El método actionPerformed hay que implementarlo en el modelo, nunca en el controlador.

Pregunta 12: Dada la siguiente jerarquía de clases:



Y la siguiente inicialización de objetos:

```
public static void main(String[] args) {
    Vehiculo v1=new Coche();
    Vehiculo v2=new Vehiculo();
    Bicicleta b=new Bicicleta();
    Coche c=new Coche();
}
```

¿Cuáles de las siguientes asignaciones son correctas?

- c=(Coche)v1;
 - c=(Coche)v2;
 - b=(Bicicleta) c;
 - b=v2;
-
- 1
 - 1 y 4
 - 1,2 y 4
 - 4

Pregunta 13: Sea una nueva definición de las clases Post y MessagePost:

```
public class Post
{
    public String toString(){
        return "Mensaje 2 ";
    }
}

public class MessagePost extends Post
{
    public String toString(){
        return "Mensaje 1 "+super.toString();
    }
}
```

```

    }

public class Test
{
    public static void main(String args[])
    {
        Post p = new MessagePost();
        System.out.println(p);
    }
}

```

¿Qué se mostrará por pantalla al ejecutar el método main de la clase Test?

- a. Error de Compilación
- b. Mensaje 2 Mensaje 1
- c. Mensaje 1 Mensaje2
- d. Mensaje 2

Pregunta 14: En el código libreta de direcciones (address book) explicado en el libro de texto se hace uso de las aserciones.

```

public void removeDetails(String key)
{
    if(key == null){
        throw new IllegalArgumentException("Null      key      passed      to
removeDetails. ");
    }
    if(keyInUse(key)) {
        ContactDetails details = book.get(key);
        book.remove(details.getName());
        book.remove(details.getPhone());
        numberOfEntries--;
    }
    assert !keyInUse(key);
    assert consistentSize() : "Inconsistent book size";
}

```

Indique cual de las siguientes afirmaciones es cierta:

- a. Se muestra un error AssertionError en el caso en el que el método keyInUse devuelva como resultado falso.
- b. Se muestra el mensaje “Inconsistent book size” en el caso en el que el método consistentSize devuelva como resultado falso.
- c. Se muestra el mensaje “Inconsistent book size” en el caso en el que el método consistentSize devuelva como resultado verdadero.
- d. Se lanza una excepción derivada de la clase Exception en el caso en el que el método keyInUse devuelva como resultado verdadero.

Pregunta 15: Indique cual de las siguientes afirmaciones relativas al uso de excepciones en Java es falsa:

- a. Las excepciones comprobadas están pensadas para aquellos casos en los que el cliente debería esperar que una operación pueda fallar.
- b. Las excepciones no comprobadas están pensadas para aquellos casos que nunca deberían fallar durante la operación normal.
- c. Las excepciones no comprobadas heredan de la clase Error.
- d. Las excepciones comprobadas heredan de la clase Exception.

PARTE PRÁCTICA [6,5 PUNTOS]:

La práctica del presente curso ha sido una versión del legendario arcade “Pac-Man”. A

continuación se muestra la propuesta del juego tal y como se solicitaba para la práctica del curso:

1. El juego constará de un solo nivel donde el jugador deberá comer todos los puntos de la pantalla.
 2. El jugador controlará a Pac-Man y dispondrá de 1 vida.
 3. Los fantasmas serán controlados por el ordenador teniendo en cuenta el comportamiento diferente de cada uno.
 4. Pac-Man podrá moverse (Utilizando las flechas del teclado) arriba (Tecla Up), abajo (Tecla Down), izquierda (Tecla Left) y derecha (Tecla Right). Así mismo podrá pausar el juego pulsando la tecla "P".
 5. El área de movimiento permitido para Pac-Man y los fantasmas será el mapa del único nivel disponible.
 6. Será necesario comprobar que tanto Pac-Man como los fantasmas no superen los límites del mapa.
 7. Los caminos del mapa solo permiten el paso de un individuo al mismo tiempo, por tanto habrá que tener en cuenta las colisiones.
 8. Los fantasmas deben implementar comportamientos diferentes:
 - a. Blinky, el fantasma rojo, buscará colisionar con Pac-Man. Para acercarse a Pac-Man calculará la distancia (por ejemplo medido en filas y columnas) e intentará primero acercarse verticalmente y luego horizontalmente.
 - b. Pinky. Buscará colisionar con Pac-Man. Para acercarse a Pac-Man calculará la distancia (por ejemplo medido en filas y columnas) e intentará primero acercarse horizontalmente y luego verticalmente.
 - c. Clyde. Él no persigue a Pac-Man, si no que deambula sin una ruta específica.
 9. Se deberán de detectar dos tipos de colisiones.
 - a. Las colisiones entre Pac-Man y los fantasmas, lo que supondrá la pérdida de una vida o el final del juego en caso de no disponer de más vidas.
 - b. Las colisiones entre los fantasmas, que supondrá un cambio de dirección en los fantasmas involucrados.
 10. Habrá cuatro puntos más grandes de lo normal situados cerca de las esquinas del laberinto y proporcionarán a Pac-Man la habilidad temporal (5 segundos) de comerse a los fantasmas (todos ellos se vuelven azules mientras Pac-Man tiene esa habilidad). Después de haber sido tragados, los fantasmas se regeneran en "casa de fantasmas".
 11. Será necesario implementar un contador con los puntos obtenidos en cada momento, teniendo en cuenta los objetos comidos. Un punto pequeño supone 10 puntos. Comer un fantasma 100 puntos.
 12. Si el jugador finaliza el nivel del juego deberá aparecer un mensaje de felicitación y se volvería a mostrar la página inicial.
- a) **[2 puntos]** Diseñar utilizando un paradigma orientado a objetos, los elementos necesarios para la aplicación explicada de la práctica durante el curso. Es necesario identificar la estructura y las relaciones de herencia y de uso de las clases necesarias para almacenar y gestionar esta información. Debe hacerse uso de los mecanismos de herencia siempre que sea posible. Se valorará un buen diseño que favorezca la reutilización de código y facilite su mantenimiento.
- b) **[1,5 puntos]** Implementa la clase FantasmaClyde. Especifica sus atributos y métodos y justifica las decisiones de implementación que creas importantes. Recuerda que este fantasma deambula sin una ruta específica por el laberinto.

- c) **[1,5 puntos]** Implementa los métodos necesarios para gestionar el cambio de estado tanto en Pac-Man como en los fantasmas cuando éste pasa por encima de un punto grande. Es necesario indicar a qué clases pertenece cada uno de los métodos implementados.
- d) **[1,5 puntos]** Indique los cambios que serían necesarios en el diseño y programa para permitir la existencia de “premios” (e.g. cerezas, fresas, naranjas, manzanas, etc) en cada nivel del juego que permitan a Pac-Man aumentar su puntuación al pasar por encima de ellos.

PARTE TEÓRICA - TEST [2,5 PUNTOS]:

Solo una de las respuestas es válida. Las respuestas correctas se puntuarán con +1.0, mientras que las respondidas de manera incorrecta se puntuarán con -0.25. Las no contestadas no tendrán influencia ni positiva ni negativa en la nota.

Pregunta 1: Indica cual de las siguientes afirmaciones es correcta:

- a. Los métodos pueden devolver información de algún objeto mediante un valor de retorno.
- b. Los métodos siempre tienen parámetros con los que obtener la información necesaria.
- c. A partir de una clase tan solo se puede crear un solo objeto.
- d. El estado de los objetos se representa mediante los métodos implementados.

Pregunta 2: Dado el siguiente fragmento de código,

```
int estudiante = 1;
boolean[] matriculas = new boolean[3];
boolean aprobado = matriculas [estudiante];
```

Indica cual de las siguientes afirmaciones es correcta en relación al valor de la variable aprobado.

- a. aprobado tiene el valor 0
- b. aprobado tiene el valor null
- c. aprobado tiene el valor false
- d. Se produce una excepción y aprobado no posee ningún valor

Pregunta 3: Indica cual de las siguientes afirmaciones es correcta:

- a. Los campos también son conocidos como variables de estado.
- b. El alcance de una variable define la sección de código desde donde la variable puede ser declarada.
- c. Los constructores permiten que cada objeto sea preparado adecuadamente cuando es creado.
- d. El tiempo de vida de una variable describe el número de veces que es utilizada en un método.

Pregunta 4: Supongamos que queremos implementar una Agenda, ¿cuál sería la salida del siguiente código?

```
public class Agenda {

    public static void main(String argv[]) {
        Agenda agenda = new Agenda();
    }

    protected Agenda(){
        for(int i=0; i<10; i++){
            System.out.println(i);
        }
    }
}
```

- a. Error de Compilación ya que los constructores no pueden ser declarados como “protected”.
- b. Error en tiempo de ejecución ya que los constructores no pueden ser declarados como “protected”.
- c. Compilación correcta y salida de los dígitos de 0 a 10.
- d. Compilación correcta y salida de los dígitos de 0 a 9.

Pregunta 5: Indica cual de las siguientes afirmaciones es correcta:

- a. Los campos se definen dentro de los constructores y de los métodos.
- b. Los campos se usan para almacenar datos que nunca persisten durante la vida del objeto.
- c. Los campos tienen un tiempo de vida que perdura después de terminar el objeto.
- d. La accesibilidad de los campos se extiende a toda clase y por este motivo pueden usarse dentro de cualquier constructor o método de clase en la que estén definidos.

Pregunta 6: Indica cual de las siguientes afirmaciones es correcta:

- a. El lenguaje Java tiene tres variantes del ciclo for : for-each, for y for-do.
- b. Un ciclo while es similar en su estructura y propósito que el ciclo for-each.
- c. El tipo de la variable de ciclo no tiene porqué ser el mismo que el tipo del elemento declarado para la colección que estamos recorriendo con un ciclo.
- d. Un índice es un objeto que proporciona funcionalidad para recorrer todos los elementos de una colección.

Pregunta 7: Indica cual de las siguientes afirmaciones es correcta:

- a. La prueba es la actividad de descubrir si una pieza de código produce el comportamiento pretendido.
- b. Una aserción es una expresión que establece una condición que esperamos que resulte verdadera.
- c. Un seguimiento es la actividad de trabajar a través de un segmento de código línea por línea, mientras se observan cambios de estado y otros comportamientos de la aplicación.
- d. Todas las respuestas anteriores son correctas.

Pregunta 8: Indica cual de las siguientes afirmaciones es correcta:

- a. Un objeto de tipo String puede ser modificado una vez que está creado, por tanto no es un ejemplo de objeto inmutable
- b. La clase String tiene un método de nombre trim que permite modificar caracteres en cualquier posición de una cadena
- c. Las cadenas de texto de tipo String solamente se pueden comparar mediante el operador “==”
- d. Un objeto es inmutable si su contenido o su estado no puede ser cambiado una vez que se ha creado

Pregunta 9: Basado en el ejemplo de la Base de Datos de CDs y DVDs visto en la asignatura en el capítulo 8, ¿cuál sería la salida del siguiente código?

```
public class BaseDeDatos {  
    public final void metodoAregarElemento() {  
        System.out.println("Agregar Elemento");  
    }  
}  
  
public class BaseDeDatosDeMusica {  
    public static void main(String argv[]) {  
        BaseDeDatos db = new BaseDeDatos();  
        db.metodoAregarElemento();  
    }  
}
```

- a. Error en tiempo de compilación indicando que una clase con métodos finales deben ser declarada también como final.
- b. Error en tiempo de compilación indicando que no se puede heredar de una clase con métodos finales.
- c. Error en tiempo de ejecución indicando que BaseDeDatos no ha sido definida como final.
- d. Éxito en la compilación y salida “Agregar Elemento”.

Pregunta 10: Indica cual de las siguientes afirmaciones es correcta:

- a. El término acoplamiento describe cuánto se ajusta una unidad de código a una tarea lógica o a una entidad
- b. El acoplamiento describe la conectividad de los propios objetos de una clase
- c. Un encapsulamiento apropiado en las clases reduce el acoplamiento
- d. Un sistema debilmente acoplado se caracteriza por la imposibilidad de modificar una de sus clases sin tener que realizar cambios en ninguna otra

Pregunta 11: Dado el siguiente fragmento de código que pretende mostrar un ejemplo de sobrescritura:

```
class Examen {  
    private float pregunta = 1.0f ;  
    protected float getNota () {return pregunta;}  
}  
  
class Test extends Examen {  
    private float nota = 2.0f;  
    //Insertar código aquí  
}
```

Indique cual de las siguientes opciones completaría el código anterior para dar lugar a un ejemplo correcto de sobrescritura:

- a. public float getNota (float valor) { return valor;}
- b. public float getNota () { return nota;}
- c. float getNota () { return nota;}
- d. float double getNota () { return nota;}

Pregunta 12: Indica cual de las siguientes afirmaciones es correcta:

- a. Una IGU se construye mediante visores que se ubican en la pantalla.
- b. La distribución de los componentes en la pantalla se lleva a cabo mediante gestores de disposición.
- c. Los componentes se ubican en una ventana agregándolos a la barra de estado o al panel agregador.
- d. Un objeto puede escuchar los eventos de los componentes implementando una interfaz interpretadora de eventos.

Pregunta 13: En el siguiente fragmento de código hemos definido la ejecución de cinco bloques. Estos bloques se ejecutarán dependiendo de las excepciones que se produzcan en cada caso.

```
// Bloque1
try{
    // Bloque2
} catch (ArithmeticException e) {
    // Bloque3
} finally{
    // Bloque4
}
// Bloque5
```

Indique cual de las siguientes afirmaciones es correcta:

- a. El Bloque4 no se ejecutará si se produce una excepción de tipo aritmético en el Bloque2
- b. El Bloque4 se ejecutará antes de que la excepción producida por un acceso a un objeto nulo (null) en el Bloque2 se propague hacia arriba
- c. El Bloque4 no se ejecutará si se produce un acceso a un objeto nulo (null) en el Bloque2
- d. El Bloque4 se ejecutará antes que el Bloque3 si se produce una excepción de tipo aritmético en el Bloque2

Pregunta 14: Indica cual de las siguientes afirmaciones es correcta:

- a. Una superclase es una clase que es implementada por otra.
- b. Una subclase es una clase que implementa a otro objeto.
- c. La herencia nos permite heredar pero no reutilizar en un nuevo contexto clases que fueron escritas previamente.
- d. Las clases que están vinculadas mediante una relación de herencia forman una jerarquía de herencia.

Pregunta 15: Indica cual de las siguientes afirmaciones es correcta:

- a. La interfaz de una clase describe lo que hace la clase y cómo puede usarse pudiendo mostrar parte de su implementación.
- b. Un mapa es una colección que almacena entradas de ternas de valores llave/valor/posición.
- c. La documentación de una clase debe ser suficientemente detallada como para que otros programadores puedan usar la clase sin necesidad de leer su implementación.
- d. Los modificadores de acceso definen las restricciones de uso de un objeto para determinados métodos, constructores o campos.

PARTE PRÁCTICA [6,5 PUNTOS]:

La práctica del presente curso ha sido una terminal punto de venta (por sus siglas, TPV) que ha servido para estudiar y practicar los mecanismos de la Programación Orientada a Objetos.

Definición de TPV y Características

Según la Wikipedia (www.wikipedia.org), un terminal punto de venta (cuyo acrónimo es TPV hace referencia al dispositivo y tecnologías que ayudan en la tarea de gestión de un establecimiento comercial de venta al público que puede contar con sistemas informáticos especializados mediante una interfaz accesible para los vendedores.

Los TPV permiten la creación e impresión del tique de venta mediante las referencias de productos, realizan diversas operaciones durante todo el proceso de venta, así como cambios en el inventario. También generan diversos reportes que ayudan en la gestión del negocio. Los TPV se componen de una parte hardware (dispositivos físicos) y otra software (sistema operativo y programa de gestión).

En nuestro caso concreto, el hardware será un ordenador tipo PC o similar y nuestro software será una aplicación desarrollada en Java que se ejecutará sobre dicho equipo.

Funcionalidades

Los TPV permiten la implementación desde labores simples de gestión de una venta, hasta operaciones más complejas como es la gestión de almacén o inventario, gestión de facturación o gestión de clientes. En esta práctica, se propondrá diferentes funcionalidades para el sistema de gestión del TPV:

- Llevar un control de diferentes elementos que existen en nuestro establecimiento. Así, los productos habrán de estar identificados en el sistema por, al menos, los siguientes datos: código descriptivo (por ejemplo, el código de barras), descripción, precio unitario sin IVA, IVA aplicable, precio unitario con IVA, cantidad disponible en stock.
- El sistema debe permitir dar de alta nuevos productos, dar de baja productos existentes así como modificar los datos del mismo.
- Realizar la importación y/o exportación de los productos a/desde ficheros (u otro método similar que el alumno considere en su lugar).
- Llevar un control de las diferentes ventas que se producen. Así, el sistema deberá llevar un control de tickets generados, de modo que cada ticket se considerará una venta. Cada ticket tiene que tener un código de identificador único. Una forma de generar un código único podría ser de la forma AAAAMMDDHHMM, donde AAAA es el año en curso, MM el mes en que se genera la venta, DD el día de la venta, HHMM las horas y minutos en las que se inicia la venta. Asumiremos que sólo hay un TPV, por lo que no procede que haya dos ventas simultáneas.
- La venta consistirá en la inclusión de varios productos en una lista, generándose una línea por cada producto vendido. Cada línea mostrará, al menos, el código del producto, la descripción del producto, la cantidad de unidades vendidas, el precio unitario con IVA, el IVA que se le aplica y el importe total de la venta de ese producto según el número de unidades vendidas.
- El proceso de venta implicará automáticamente un proceso de actualización del inventario. De este modo, si se introduce un código que no pertenece a ningún producto, o si se introduce un producto que no existe en stock (o más unidades de las existentes), el programa deberá mostrar los errores correspondientes.
- El sistema deberá permitir también introducir un producto a vender en el ticket haciendo una búsqueda por la descripción, además de con el código que lo identifica.
- Realizar la importación y/o exportación de los diferentes tickets de ventas a/desde ficheros (u otro método similar que el alumno considere en su lugar).
- Llevar un control de los diferentes clientes que trabajan con el establecimiento comercial. Así, los

clientes habrán de estar identificados en el sistema por, al menos, los siguientes datos: código identificativo del cliente, NIF o CIF, nombre y apellidos / razón social, domicilio, fecha de alta en el sistema.

- El sistema debe permitir dar de alta nuevos clientes, dar de baja clientes existentes así como modificar los datos de los mismos.
 - Realizar la importación y/o exportación de los clientes a/desde ficheros (u otro método similar que el alumno considere en su lugar).
 - Permitir generar facturas a partir de un conjunto de tickets. Puede generar facturas agrupando diferentes tickets siempre y cuando pertenezcan al mismo cliente y se han realizado dentro del mismo periodo fiscal (es decir, dentro del mismo año). La información que irá en cada factura deberá ser, al menos, la siguiente: número de la factura (identificador único), CIF del vendedor, razón social del vendedor, fecha de emisión de la factura, datos del cliente (los indicados con anterioridad, excepto la fecha de alta en el sistema), listado de los diferentes productos vendidos (especificando para cada producto, el ticket en el que se encuentra, su cantidad vendida e importe total) así como suma del total de la venta (valor total de la factura).
 - Realizar la importación y/o exportación de las facturas a/desde ficheros (u otro método similar que el alumno considere en su lugar).
 - Generación de listados: se deberá implementar, al menos, la emisión de tres listados, a saber: ventas realizadas en un intervalo de tiempo determinado agrupadas estas ventas por clientes, ventas realizadas en un intervalo de tiempo determinado a un cliente y ranking de productos más vendidos en un intervalo de tiempo determinado.
- a) **[1,0 puntos]** Diseñar utilizando un paradigma orientado a objetos, los elementos necesarios para la aplicación explicada de la práctica durante el curso. Es necesario identificar la estructura y las relaciones de herencia (mediante el uso de un diagrama de clases) y de uso de las clases necesarias para almacenar y gestionar esta información. Debe hacerse uso de los mecanismos de herencia siempre que sea posible. Se valorará un buen diseño que favorezca la reutilización de código y facilite su mantenimiento.
- b) **[1,5 puntos]** Implementa la funcionalidad que permite dar de alta nuevos clientes, dar de baja clientes existentes así como modificar los datos de los mismos. Justifique las opciones y decisiones que se tomen.
- c) **[1,5 puntos]** Implementa la funcionalidad que permite la generación de listados: se deberá implementar, al menos, la emisión de tres listados, a saber: ventas realizadas en un intervalo de tiempo determinado agrupadas estas ventas por clientes, ventas realizadas en un intervalo de tiempo determinado a un cliente y ranking de productos más vendidos en un intervalo de tiempo determinado. Justifique las opciones y decisiones que se tomen.
- d) **[2,5 puntos]** Para la siguiente versión del software se desea añadir la figura del proveedor. De cada proveedor se debe tener un listado de los productos que sirve, así como su precio, que podría actualizarse manualmente después de la última venta. El sistema debería consultar el inventario después de cada venta y por debajo de un mínimo de productos realizar un pedido al proveedor más barato, de manera automática. El número de productos pedido, podrá ser configurable para que el vendedor establezca un valor por cada producto. ¿Qué cambios serían necesarios en el diseño para adaptar esta nueva funcionalidad? Implemente el método (o métodos) que permita esta nueva funcionalidad.

PARTE TEÓRICA - TEST [2,5 PUNTOS]:

Solo una de las respuestas es válida. Las respuestas correctas se puntuarán con +1.0, mientras que las respondidas de manera incorrecta se puntuarán con -0.25. Las no contestadas no tendrán influencia ni positiva ni negativa en la nota.

Pregunta 1: ¿Qué significa el siguiente fragmento de código Java?:

```
int uno() { return 1; }
```

- a. Hay un método “int uno” que no recibe ningún parámetro de entrada y devuelve el valor 1.
- b. Hay una variable “int” cuyo valor es “uno() { return 1; }”
- c. Hay un método “uno” que no recibe ningún parámetro de entrada y devuelve un entero cuyo valor es 1.
- d. El fragmento no representa un fragmento de código legal en Java.

Pregunta 2: Para lograr que una clase entre en el depurador en BlueJ a hacer una instancia en BlueJ, ¿qué hay que hacer con el código fuente?:

- a. Compilarlo de nuevo con la opción Debug activado.
- b. Meter un punto de ruptura.
- c. Lanzar directamente el depurador.
- d. Se hace automáticamente al encontrar un error en el código.

Pregunta 3: Dado el siguiente fragmento de código en Java:

```
int m, n;
public void f() {
    m = (m + 2) % n;
    System.out.print(m + " ");
}
public void g() {
    int i = 0; m = 0; n = 8;
    while(i++<n) {
        f();
    }
}
```

¿Cuál es el resultado de ejecutar g()?

- a. 0 2 4 6 0 2 4 6
- b. 2 4 6 2 4 6 2 4
- c. 2 4 6 0 2 4 6 0 2 4
- d. 2 4 6 0 2 4 6 0

Pregunta 4: Según el libro de la asignatura, ¿cómo se llama el código fuente de una clase?:

- a. La implementación de la clase.
- b. La interfaz de la clase.
- c. Los métodos de la clase.
- d. La visibilidad de la clase.

Pregunta 5: Para captar el evento de dar en un botón en Java, ¿que interfaz hay que implementar?:

- a. public class DemoBoton extends JPanel implements WindowListener
- b. public class DemoBoton extends JPanel implements EventListener
- c. public class DemoBoton extends JPanel implements ButtonListener
- d. public class DemoBoton extends JPanel implements ActionListener

Pregunta 6: Según el libro de la asignatura, la duplicación de código es un síntoma de:

- a. Buena cohesión.
- b. Mala cohesión.
- c. La solución inevitable de un problema complejo.
- d. Mal encapsulamiento.

Pregunta 7: Según el libro de la asignatura, ¿qué significa “prueba de unidades”?:

- a. Una prueba completa de la aplicación.
- b. Una prueba sistemática de un método en concreto.
- c. Una prueba de las partes individuales de la aplicación.
- d. Una depuración completa de la aplicación.

Pregunta 8: Dado el siguiente fragmento de código de Java:

```
class I {}  
class J extends I {}  
class K extends J {}  
public void hh () {  
    J j = new J();  
    boolean b1 = j instanceof K;  
    boolean b2 = j instanceof J;  
    boolean b3 = j instanceof I;  
    boolean b4 = j instanceof Object;  
    System.out.println(b1 + " " + b2 + " " + b3 + " " + b4);  
}
```

¿Cuál sería la salida del método hh?

- a. false true true true
- b. true true true true
- c. false true true false
- d. false false true true

Pregunta 9: Un método abstracto se declara de la siguiente forma:

- a. abstract String ss(){}
b. abstract String ss();
c. abstract String ss(){return void};
d. abstract ss();

Pregunta 10: Si existe un **ArrayList<String>** as, ¿cómo se consigue un iterador sobre as?

- a. Iterator it = as.getIterator();
- b. Iterator it = as.iterator();
- c. Iterator it = new Iterator(as);
- d. Iterator it = as.nextIterator();

Pregunta 11: Si las clases J y K heredan de la clase I y la clase L hereda de la clase J, entonces (indica la respuesta que daría un error de compilación):

- a. I i = new K();
- b. J j = new K();
- c. J j = new L();
- d. K k = new I();

Pregunta 12: En la práctica hay que leer información desde un archivo. ¿Cómo se puede generar una excepción si no se encuentra el archivo?:

- a. throw new FileNotFoundException("Archivo no encontrado");
- b. throw new NoFileNotFoundException("Archivo no encontrado");
- c. throw new FileException("Archivo no encontrado");
- d. throw new NullPointerException("Archivo no encontrado");

Pregunta 13: Dado el siguiente fragmento de código en Java:

```
String s1 = 0 + "5"; // (1)
String s2 = "0" + 5; // (2)
String s3 = 0 + 5 + ""; // (3)
String s4 = java.lang.Integer.toString(0) + 5; // (4)
```

¿Cuáles son formas **válidas** de construir una cadena?:

- a. 1, 2 y 4
- b. 1, 2 y 3
- c. 4
- d. Todas las formas son válidas.

Pregunta 14: Según el libro de la asignatura, las instrucciones condicionales llevan a cabo (con la excepción del switch):

- a. una de dos acciones posibles.
- b. una de varias acciones posibles.
- c. las dos acciones posibles.
- d. dos o más de las acciones posibles.

Pregunta 15: ¿Cuál es la diferencia entre declarar una variable de clase **private** y **protected**?:

- a. private: Acceso solo dentro de la clase, protected: Acceso desde la clase y sus hijos.
- b. private: Acceso desde la clase y sus hijos, protected: Acceso solo dentro de la clase.
- c. private: Acceso solo dentro del paquete, protected: Acceso desde la clase y sus hijos.
- d. Se pueden aplicar private y protected solamente a la declaración de clases y no a variables.

PARTE PRÁCTICA [6,5 PUNTOS]:

La práctica del presente curso ha sido una terminal punto de venta (por sus siglas, TPV) que ha servido para estudiar y practicar los mecanismos de la Programación Orientada a Objetos.

Definición de TPV y Características

Según la Wikipedia (www.wikipedia.org), un terminal punto de venta (cuyo acrónimo es TPV hace referencia al dispositivo y tecnologías que ayudan en la tarea de gestión de un establecimiento comercial de venta al público que puede contar con sistemas informáticos especializados mediante una interfaz accesible para los vendedores.

Los TPV permiten la creación e impresión del tique de venta mediante las referencias de productos, realizan diversas operaciones durante todo el proceso de venta, así como cambios en el inventario. También generan diversos reportes que ayudan en la gestión del negocio. Los TPV se componen de una parte hardware (dispositivos físicos) y otra software (sistema operativo y programa de gestión).

En nuestro caso concreto, el hardware será un ordenador tipo PC o similar y nuestro software será una aplicación desarrollada en Java que se ejecutará sobre dicho equipo.

Funcionalidades

Los TPV permiten la implementación desde labores simples de gestión de una venta, hasta operaciones más complejas como es la gestión de almacén o inventario, gestión de facturación o gestión de clientes. En esta práctica, se propondrá diferentes funcionalidades para el sistema de gestión del TPV:

- Llevar un control de diferentes elementos que existen en nuestro establecimiento. Así, los productos habrán de estar identificados en el sistema por, al menos, los siguientes datos: código descriptivo (por ejemplo, el código de barras), descripción, precio unitario sin IVA, IVA aplicable, precio unitario con IVA, cantidad disponible en stock.
- El sistema debe permitir dar de alta nuevos productos, dar de baja productos existentes así como modificar los datos del mismo.
- Realizar la importación y/o exportación de los productos a/desde ficheros (u otro método similar que el alumno considere en su lugar).
- Llevar un control de las diferentes ventas que se producen. Así, el sistema deberá llevar un control de tickets generados, de modo que cada ticket se considerará una venta. Cada ticket tiene que tener un código de identificador único. Una forma de generar un código único podría ser de la forma AAAAMMDDHHMM, donde AAAA es el año en curso, MM el mes en que se genera la venta, DD el día de la venta, HHMM las horas y minutos en las que se inicia la venta. Asumiremos que sólo hay un TPV, por lo que no procede que haya dos ventas simultáneas.
- La venta consistirá en la inclusión de varios productos en una lista, generándose una línea por cada producto vendido. Cada línea mostrará, al menos, el código del producto, la descripción del producto, la cantidad de unidades vendidas, el precio unitario con IVA, el IVA que se le aplica y el importe total de la venta de ese producto según el número de unidades vendidas.
- El proceso de venta implicará automáticamente un proceso de actualización del inventario tal y como se ha definido en el Nivel 2. De este modo, si se introduce un código que no pertenece a ningún producto, o si se introduce un producto que no existe en stock (o más unidades de las existentes), el programa deberá mostrar los errores correspondientes.
- El sistema deberá permitir también introducir un producto a vender en el ticket haciendo una búsqueda por la descripción, además de con el código que lo identifica.
- Realizar la importación y/o exportación de los diferentes tickets de ventas a/desde ficheros (u otro método similar que el alumno considere en su lugar).
- Llevar un control de los diferentes clientes que trabajan con el establecimiento comercial. Así, los

clientes habrán de estar identificados en el sistema por, al menos, los siguientes datos: código identificativo del cliente, NIF o CIF, nombre y apellidos / razón social, domicilio, fecha de alta en el sistema.

- El sistema debe permitir dar de alta nuevos clientes, dar de baja clientes existentes así como modificar los datos de los mismos.
 - Realizar la importación y/o exportación de los clientes a/desde ficheros (u otro método similar que el alumno considere en su lugar).
 - Permitir generar facturas a partir de un conjunto de tickets. Puede generar facturas agrupando diferentes tickets siempre y cuando pertenezcan al mismo cliente y se han realizado dentro del mismo periodo fiscal (es decir, dentro del mismo año). La información que irá en cada factura deberá ser, al menos, la siguiente: número de la factura (identificador único), CIF del vendedor, razón social del vendedor, fecha de emisión de la factura, datos del cliente (los indicados con anterioridad, excepto la fecha de alta en el sistema), listado de los diferentes productos vendidos (especificando para cada producto, el ticket en el que se encuentra, su cantidad vendida e importe total) así como suma del total de la venta (valor total de la factura).
 - Realizar la importación y/o exportación de las facturas a/desde ficheros (u otro método similar que el alumno considere en su lugar).
 - Generación de listados: se deberá implementar, al menos, la emisión de tres listados, a saber: ventas realizadas en un intervalo de tiempo determinado agrupadas estas ventas por clientes, ventas realizadas en un intervalo de tiempo determinado a un cliente y ranking de productos más vendidos en un intervalo de tiempo determinado.
- a) **[1,0 puntos]** Diseñar utilizando un paradigma orientado a objetos, los elementos necesarios para la aplicación explicada de la práctica durante el curso. Es necesario identificar la estructura y las relaciones de herencia (mediante el uso de un diagrama de clases) y de uso de las clases necesarias para almacenar y gestionar esta información. Debe hacerse uso de los mecanismos de herencia siempre que sea posible. Se valorará un buen diseño que favorezca la reutilización de código y facilite su mantenimiento.
- b) **[2,0 puntos]** Implementa la funcionalidad de generar facturas agrupando diferentes tickets siempre y cuando pertenezcan al mismo cliente y se han realizado dentro del mismo periodo fiscal (es decir, dentro del mismo año). La información que irá en cada factura deberá ser, al menos, la siguiente: número de la factura (identificador único), CIF del vendedor, razón social del vendedor, fecha de emisión de la factura, datos del cliente (los indicados con anterioridad, excepto la fecha de alta en el sistema), listado de los diferentes productos vendidos (especificando para cada producto, el ticket en el que se encuentra, su cantidad vendida e importe total) así como suma del total de la venta (valor total de la factura).
- c) **[1,0 punto]** Implementa un método que dé una notificación cuando se venda el último ejemplar de un producto en el almacén para que se pueda pedir más.
- d) **[2,5 puntos]** Para la siguiente versión del software se quiere incluir un catálogo de los productos para que un cliente pueda mirarlo en pantallas dentro de la tienda y comprar los productos directamente desde el catálogo. Se trataría de una nueva interfaz sobre el TPV que proporcionaría algunas de las funciones (aquellas relevantes para el cliente) directamente al cliente. ¿Qué cambios serían necesarios en el diseño para incluir esta nueva funcionalidad? Implementa los cambios necesarios que permitan esta nueva funcionalidad.

**UNIVERSIDAD NACIONAL DE EDUCACIÓN A DISTANCIA – ESCUELA TÉCNICA SUPERIOR DE
INGENIERÍA INFORMÁTICA**

**71901072 – PROGRAMACIÓN ORIENTADA A OBJETOS (GRADO EN INGENIERÍA INFORMÁTICA /
TECNOLOGÍAS DE LA INFORMACIÓN)**

**JUNIO/SEPTIEMBRE 2015 – MODELO C – NO ESTÁ PERMITIDO EL USO DE MATERIAL
ADICIONAL**

PARTE TEÓRICA - TEST [2,5 PUNTOS]:

Solo una de las respuestas es válida. Las respuestas correctas se puntuarán con +1.0, mientras que las respondidas de manera incorrecta se puntuarán con -0.25. Las no contestadas no tendrán influencia ni positiva ni negativa en la nota.

Pregunta 1: Según el texto de la bibliografía básica de la asignatura, los constructores ...

- a. Almacenan datos de manera persistente dentro de un objeto.
- b. Implementan el comportamiento de un objeto.
- c. Son responsables de garantizar que un objeto se configure apropiadamente en el momento de usarlo por primera vez, siempre y cuando haya sido creado previamente.
- d. Ninguna de las anteriores.

Pregunta 2: En BlueJ, ¿qué nos permite experimentar con expresiones Java?

- a. El Pad Code
- b. El Patch Code
- c. El Pan Code
- d. Ninguna de las anteriores.

Pregunta 3: Por modularización entendemos ...

- a. El proceso de dividir un todo en partes laxamente definidas que puedan construirse y examinarse en conjunto y que interactúen de formas bien definidas.
- b. El proceso de dividir un todo en partes bien definidas que puedan construirse y examinarse en conjunto y que interactúen de formas bien definidas.
- c. El proceso de dividir un todo en partes bien definidas que puedan construirse y examinarse por separado y que interactúen de formas bien definidas.
- d. Ninguna de las anteriores.

Pregunta 4: Queremos compilar el siguiente código que se puede encontrar en el texto base de la asignatura y que ha sido modificado convenientemente. ¿Cuál es el resultado que obtenemos al compilar?

```
1  public class MailItem
2  {
3      private static String from;
4      private String to;
5      private String message;
6
7      public static MailItem (String from, String to, String m)
8      {
9          this.from = from;
10         this.to = to;
11         this.message = m;
12     }
13 }
```

- a. El código compila sin errores
- b. Produce un error de compilación en la línea 3.
- c. Produce un error de compilación en la línea 11.
- d. Ninguna de las anteriores.

Pregunta 5: Queremos compilar el siguiente código que se puede encontrar en el texto base de la asignatura y que ha sido modificado convenientemente. Al compilar, BlueJ nos da error de compilación. ¿Qué deberemos cambiar para que el programa compile?

```

1 import java.util.ArrayList;
2
3 public class MusicOrganizer
4 {
5     private ArrayList <String> files;
6     private MusicPlayer player;
7
8     public MusicOrganizer ()
9     {
10         files = new ArrayList <String> ();
11         player = new MusicPlayer();
12     }
13
14     public void startPlayingFile (int index)
15     {
16         String filename = files.get(index);
17         player.startPlaying (filename);
18     }
19
20     public void stopPlaying ()
21     {
22         player.stop();
23     }
24 }
```

- a. Definir la clase MusicPlayer convenientemente, con al menos los métodos stopPlaying() y startPlayingFile (int index), e importarla (si fuese necesario) mediante la instrucción import MusicPlayer;
- b. Modificar la línea 10 para que quede así: files = new ArrayList <String> (0);
- c. Modificar la línea 6 para que quede así: public MusicPlayer player;
- d. Hay que aplicar los cambios indicados en a, b y c, puesto que si alguno no se aplicase, el código no compilaría.

Pregunta 6: Queremos compilar el siguiente código que se puede encontrar en el texto base de la asignatura. Al compilar, BlueJ podría darnos un error en tiempo de compilación y/o un error en tiempo de ejecución. ¿Cómo deberemos proceder para que el código compile y se ejecute correctamente?

```

1 public class Ejemplo
2 {
3     public static Vehicle v;
4     public static Car c;
5     public static Bicycle b;
6
7     public static void main ()
8     {
9         c = new Car();
10        v = c;
```

```

11         b = (Bicycle) c;
12         b = (Bicycle) v;
13         System.out.println("Funciona");
14     }
15 }
```

- a. Con independencia de cómo definamos las clases Vehicle, Car y Bicycle, siempre nos dará un error en tiempo de ejecución en la línea 12.
- b. Con independencia de cómo definamos las clases Vehicle, Car y Bicycle, siempre nos dará un error en tiempo de compilación en la línea 11.
- c. Si definimos que Car extends Vehicle y que Bicycle extends Car, conseguimos eliminar todos los errores del código y muestra el mensaje por pantalla "Funciona".
- d. Si definimos que Vehicle extends Bicycle y que Car extends Vehicle, conseguimos eliminar todos los errores del código y muestra el mensaje por pantalla "Funciona".

Pregunta 7: Queremos compilar el siguiente código que se puede encontrar en el texto base de la asignatura. ¿Qué ocurre al compilarlo con BlueJ?

```

1 import java.util.List;
2 interface Actor
3 {
4     void act (List <Actor> newActors);
5     boolean isActive();
6 }
```

- a. Compila, no proporcionando ningún error en tiempo de compilación.
- b. Compila, no proporcionando ningún error en tiempo de compilación, del mismo modo que también compilaría si prescindieramos de la línea 1.
- c. No compila. Hay que modificar la línea 2 quedando de la siguiente manera: public interface Actor
- d. No compila. Hay que modificar la línea 4 quedando de la siguiente manera: public void act (List <Actor> newActors);

Pregunta 8: Según el texto de la bibliografía básica de la asignatura ...

- a. El tipo estático de una variable v y el tipo dinámico de una variable v coinciden siempre.
- b. El tipo estático de una variable v se modifica automáticamente al modificar el tipo dinámico de la variable.
- c. El tipo estático siempre coincide con el tipo declarado en la instrucción de declaración de la variable.
- d. Ninguna de las anteriores.

Pregunta 9: Según el texto de la bibliografía básica de la asignatura, con respecto a la igualdad de referencias podemos afirmar ...

- a. La igualdad de referencia tiene en cuenta el contenido de los objetos.
- b. No es posible usar la igualdad de referencias para realizar comparaciones de cadenas de caracteres.
- c. El método equals heredado de la clase Object no permite comprobar que existe una igualdad de referencias.
- d. Ninguna de las anteriores.

Pregunta 10: Queremos compilar el siguiente código que se puede encontrar en el texto base de la asignatura y que hemos modificado. ¿Qué ocurre al compilarlo con BlueJ?

```
1 import java.util.List;
2
3 public abstract class Animal
4 {
5     public void act (List<Animal> newAnimals, char animals) {}
6     public static void act (List<Animal> newAnimals, int animals);
7     abstract public static void act (List<Animal> newAnimals, String animals);
8 }
```

- a. Las líneas 5 y 6 provocan errores de compilación.
- b. Las líneas 5 y 7 provocan errores de compilación.
- c. Las líneas 6 y 7 provocan errores de compilación.
- d. Las líneas 5, 6 y 7 provocan errores de compilación.

Pregunta 11: Queremos compilar el siguiente código que se puede encontrar en el texto base de la asignatura y que hemos modificado. ¿Qué ocurre al compilarlo con BlueJ?

```
1 public class Ejemplo
2 {
3     public static void main (String []args)
4     {
5         int[] numeros = new int[] {1,2,3};
6
7         System.out.print(numeros);
8         System.out.print(numeros.length);
9         System.out.print(numeros.last);
10        System.out.print(numeros.first);
11    }
12 }
```

- a. Se produce un error de compilación en las líneas 8, 9 y 10
- b. Se produce un error de compilación en las líneas 9 y 10.
- c. Se produce un error de compilación en línea 10.
- d. Ninguna de las anteriores.

Pregunta 12: ¿Cuál de las siguientes opciones permite modificar una cadena declarada como String input;?

- a. input.toUpperCase();
- b. input.trim();
- c. input.startsWith("hola");
- d. Ninguna de las anteriores.

Pregunta 13: En relación a los conceptos de acoplamiento y cohesión, podemos afirmar ...

- a. Un alto grado de acoplamiento implica necesariamente un alto grado de cohesión.
- b. Un bajo grado de acoplamiento no implica necesariamente un alto grado de cohesión.
- c. En un diseño de clases perseguimos un bajo grado de cohesión y un bajo acoplamiento
- d. Ninguna de las anteriores.

Pregunta 14: Queremos compilar el siguiente código que se puede encontrar en el texto base de la asignatura y que hemos modificado. El código compila sin causar ningún error de compilación, pero no muestra nada por pantalla. ¿Qué falta por añadir para que el código muestre algo por pantalla al crear un objeto de la clase ImageViewer dentro del entorno de BlueJ?

```
1 import java.awt.*;
2 import java.awt.event.*;
3 import javax.swing.*;
4
5 public class ImageViewer extends JFrame
6 {
7     public ImageViewer()
8     {
9         super("ImageViewer");
10        makeFrame();
11    }
12
13    private void makeFrame()
14    {
15        Container contentPane = getContentPane();
16        JLabel label = new JLabel("I am a label.");
17        contentPane.add(label);
18    }
19
20 }
21 }
```

- a. Añadir la instrucción `this.pack()` en la línea 11.
- b. Añadir la instrucción `pack()` en la línea 19.
- c. No hay que añadir nada. Se puede ver el texto “I am a label” en pantalla.
- d. Ninguna de las anteriores

Pregunta 15: Queremos compilar el siguiente código que se puede encontrar en el texto base de la asignatura y que ha sido modificado convenientemente. Se produce un error de compilación. ¿Qué línea es la que contiene un error, tal que si la modificamos convenientemente, el código compila y no provoca ningún error de compilación en BlueJ?

```
1 import java.io.*;
2
3 public class Ejemplo
4 {
5     public static void main () throws IOException
6     {
7         String filename = new String ("EJEMPLO");
8         try {
9             throw IOException ();
10        }
11        catch (Exception e) {
12            System.out.println("Unable to save to "+ filename);
13        }
14    }
15 }
```

- a. El error está en la línea 5
- b. El error está en la línea 7
- c. El error está en la línea 9
- d. El error está en la línea 11

PARTE PRÁCTICA [6,5 PUNTOS]:

La práctica del presente curso ha sido una terminal punto de venta (por sus siglas, TPV) que ha servido para estudiar y practicar los mecanismos de la Programación Orientada a Objetos.

Definición de TPV y Características

Según la Wikipedia (www.wikipedia.org), un terminal punto de venta (cuyo acrónimo es TPV hace referencia al dispositivo y tecnologías que ayudan en la tarea de gestión de un establecimiento comercial de venta al público que puede contar con sistemas informáticos especializados mediante una interfaz accesible para los vendedores.

Los TPV permiten la creación e impresión del tique de venta mediante las referencias de productos, realizan diversas operaciones durante todo el proceso de venta, así como cambios en el inventario. También generan diversos reportes que ayudan en la gestión del negocio. Los TPV se componen de una parte hardware (dispositivos físicos) y otra software (sistema operativo y programa de gestión).

En nuestro caso concreto, el hardware será un ordenador tipo PC o similar y nuestro software será una aplicación desarrollada en Java que se ejecutará sobre dicho equipo.

Funcionalidades

Los TPV permiten la implementación desde labores simples de gestión de una venta, hasta operaciones más complejas como es la gestión de almacén o inventario, gestión de facturación o gestión de clientes. En esta práctica, se propondrá diferentes funcionalidades para el sistema de gestión del TPV:

- Llevar un control de diferentes elementos que existen en nuestro establecimiento. Así, los productos habrán de estar identificados en el sistema por, al menos, los siguientes datos: código descriptivo (por ejemplo, el código de barras), descripción, precio unitario sin IVA, IVA aplicable, precio unitario con IVA, cantidad disponible en stock. □
- El sistema debe permitir dar de alta nuevos productos, dar de baja productos existentes así como modificar los datos del mismo. □
- Realizar la importación y/o exportación de los productos a/desde ficheros (u otro método similar que el alumno considere en su lugar). □
- Llevar un control de las diferentes ventas que se producen. Así, el sistema deberá llevar un control de tickets generados, de modo que cada ticket se considerará una venta. Cada ticket tiene que tener un código de identificador único. Una forma de generar un código único podría ser de la forma AAAAMMDDHHMM, donde AAAA es el año en curso, MM el mes en que se genera la venta, DD el día de la venta, HHMM las horas y minutos en las que se inicia la venta. Asumiremos que sólo hay un TPV, por lo que no procede que haya dos ventas simultáneas. □
- La venta consistirá en la inclusión de varios productos en una lista, generándose una línea por cada producto vendido. Cada línea mostrará, al menos, el código del producto, la descripción del producto, la cantidad de unidades vendidas, el precio unitario con IVA, el IVA que se le aplica y el importe total de la venta de ese producto según el número de unidades vendidas. □
- El proceso de venta implicará automáticamente un proceso de actualización del inventario. De este modo, si se introduce un□ código que no pertenece a ningún producto, o si se introduce un producto que no existe en stock (o más unidades de las existentes), el programa deberá mostrar los errores correspondientes.
- El sistema deberá permitir también introducir un producto a vender en el ticket haciendo una

- búsqueda por la descripción, además de con el código que lo identifica.
- Realizar la importación y/o exportación de los diferentes tickets de ventas a/desde ficheros (u otro método similar que el alumno considere en su lugar).
 - Llevar un control de los diferentes clientes que trabajan con el establecimiento comercial. Así, los clientes habrán de estar identificados en el sistema por, al menos, los siguientes datos: código identificativo del cliente, NIF o CIF, nombre y apellidos / razón social, domicilio, fecha de alta en el sistema.
 - El sistema debe permitir dar de alta nuevos clientes, dar de baja clientes existentes así como modificar los datos de los mismos.
 - Realizar la importación y/o exportación de los clientes a/desde ficheros (u otro método similar que el alumno considere en su lugar).
 - Permitir generar facturas a partir de un conjunto de tickets. Puede generar facturas agrupando diferentes tickets siempre y cuando pertenezcan al mismo cliente y se han realizado dentro del mismo periodo fiscal (es decir, dentro del mismo año). La información que irá en cada factura deberá ser, al menos, la siguiente: número de la factura (identificador único), CIF del vendedor, razón social del vendedor, fecha de emisión de la factura, datos del cliente (los indicados con anterioridad, excepto la fecha de alta en el sistema), listado de los diferentes productos vendidos (especificando para cada producto, el ticket en el que se encuentra, su cantidad vendida e importe total) así como suma del total de la venta (valor total de la factura).
 - Realizar la importación y/o exportación de las facturas a/desde ficheros (u otro método similar que el alumno considere en su lugar).
 - Generación de listados: se deberá implementar, al menos, la emisión de tres listados, a saber: ventas realizadas en un intervalo de tiempo determinado agrupadas estas ventas por clientes, ventas realizadas en un intervalo de tiempo determinado a un cliente y ranking de productos más vendidos en un intervalo de tiempo determinado.
- a) **[1,0 puntos]** Diseñar utilizando un paradigma orientado a objetos, los elementos necesarios para la aplicación explicada de la práctica durante el curso. Es necesario identificar la estructura y las relaciones de herencia (mediante el uso de un diagrama de clases) y de uso de las clases necesarias para almacenar y gestionar esta información. Debe hacerse uso de los mecanismos de herencia siempre que sea posible. Se valorará un buen diseño que favorezca la reutilización de código y facilite su mantenimiento.
- b) **[1,0 puntos]** Implementa un método (o métodos) que permitan la importación (cargar al programa) de los diferentes tickets de ventas desde fichero (u otro método similar que el alumno considere en su lugar). Justifíquese las opciones y decisiones que se tomen.
- c) **[2,0 puntos]** Implementa un método (o métodos) que implementen el proceso de venta, junto con la actualización del inventario. De este modo, si se introduce un código que no pertenece a ningún producto, o si se introduce un producto que no existe en stock (o más unidades de las existentes), el programa deberá mostrar los errores correspondientes. Justifíquese las opciones y decisiones que se tomen.
- d) **[2,5 puntos]** Proporcione un método (o métodos) que permita mostrar por pantalla un formulario básico en **modo gráfico** que permita recoger los parámetros necesarios para dar de alta un nuevo cliente en el sistema. El método deberá comprobar si el cliente existe, y si existe, mostrar el correspondiente mensaje por pantalla. Si no existe, procederá a dar de alta al cliente. Justifíquese las opciones y decisiones que se tomen.

**UNIVERSIDAD NACIONAL DE EDUCACIÓN A DISTANCIA – ESCUELA TÉCNICA SUPERIOR DE
INGENIERÍA INFORMÁTICA**
**71901072 – PROGRAMACIÓN ORIENTADA A OBJETOS (GRADO EN INGENIERÍA INFORMÁTICA /
TECNOLOGÍAS DE LA INFORMACIÓN)**
**JUNIO/SEPTIEMBRE 2015 – MODELO D – NO ESTÁ PERMITIDO EL USO DE MATERIAL
ADICIONAL**

PARTE TEÓRICA - TEST [2,5 PUNTOS]:

Solo una de las respuestas es válida. Las respuestas correctas se puntuarán con +1.0, mientras que las respondidas de manera incorrecta se puntuarán con -0.25. Las no contestadas no tendrán influencia ni positiva ni negativa en la nota.

Pregunta 1: ¿Cuál es el resultado de compilar y ejecutar el siguiente código?

```
import java.util.*;
public class Test {
    public static void main (String[ ] Args) {
        ArrayList<String> lista=new ArrayList<String>();
        lista.add("uno");
        lista.add("dos");
        lista.add("tres");
        for(String valor:lista){
            System.out.print("Elimino "+valor+" - ");
            lista.remove(valor);
        }
    }
}
```

- a. Muestra en consola: "Elimino uno - "
- b. Muestra en consola: "Elimino uno - " y una excepción del tipo ConcurrentModificationException
- c. Muestra en consola: "Elimino uno – Elimino dos – Elimino tres - "
- d. Muestra en consola una excepción del tipo ConcurrentModificationException

Pregunta 2: ¿Cuál es el resultado de compilar y ejecutar el siguiente código?

```
import java.util.*;
public class Test {
    public static void main (String[ ] Args) {
        ArrayList<String> lista=new ArrayList<String>();
        lista.add("uno");
        lista.add("dos");
        lista.add("tres");
        Iterator<String> it=lista.iterator();
        while(it.hasNext()){
            String valor=it.next();
            System.out.print("Elimino "+valor+" - ");
            it.remove();
        }
    }
}
```

- a. Muestra en consola: "Elimino uno - "
- b. Muestra en consola: "Elimino uno - " y una excepción del tipo ConcurrentModificationException
- c. Muestra en consola: "Elimino uno – Elimino dos – Elimino tres - "
- d. Muestra en consola una excepción del tipo ConcurrentModificationException

Pregunta 3: ¿Cuál de las siguientes afirmaciones es cierta?

- a. El término acoplamiento describe lo bien que una unidad de código se corresponde con una tarea lógica o con una entidad.
- b. El término cohesión describe la interconexión de las clases.
- c. Se debe tender a un acoplamiento débil.
- d. Se debe tender a un acoplamiento fuerte.

Pregunta 4: ¿Cuál es el resultado de compilar y ejecutar el siguiente código?

```
public class Padre
{
    private int x;
    public Padre(int valor) { x = valor; }
}
public class Hijo extends Padre
{
    public Hijo(){}
}
public class Test {
    public static void main (String[ ] Args) {
        Hijo h=new Hijo();
        System.out.println("Clase instanciada");
    }
}
```

- a. Muestra en consola: "Clase instanciada "
- b. Error de compilación. Hay que poner super() en el constructor de la clase Hijo para acceder al constructor por defecto de la clase Padre.
- c. Error de compilación. Hay que escribir explícitamente el constructor Padre() en la clase Padre.
- d. Error de compilación. Hay que poner super() en el constructor de la clase Hijo para acceder al constructor por defecto de la clase Padre y hay que escribir explícitamente el constructor Padre() en la clase Padre.

Pregunta 5: ¿Cuál es el resultado de compilar y ejecutar el siguiente código?

```
public class Padre
{
    void metodoA(){
        System.out.println("Ejecuto el método A de la clase Padre");
    }
}
public class Hijo extends Padre
{
    void metodoA(){
        System.out.println("Ejecuto el método A de la clase Hijo");
    }
}
public class Test {
    public static void main (String[ ] Args) {
        Padre h=new Hijo();
        h.metodoA();
    }
}
```

- a. Muestra en consola: "Ejecuto el método A de la clase Hijo "
- b. Muestra en consola: "Ejecuto el método A de la clase Padre "
- c. Error de compilación. Tipos incompatibles.
- d. Ninguna de las anteriores.

Pregunta 6: ¿Cuál es el resultado de compilar y ejecutar el siguiente código?

```
public class Padre
{
    void metodoA(){
        System.out.println("Ejecuto el método A de la clase Padre");
    }
}
public class Hijo extends Padre
{
    void metodoA(){
        System.out.println("Ejecuto el método A de la clase Hijo");
    }
}
public class Test {
    public static void main(String[] args) {
        Hijo h=new Padre();
        if(h instanceof Hijo)
            System.out.println("Instancia de la clase Hijo");
        else
            System.out.println("Instancia de la clase padre");
    }
}
```

- a. Muestra en consola: "Instancia de la clase Hijo "
- b. Muestra en consola: "Instancia de la clase Padre"
- c. Error de compilación. Tipos incompatibles.
- d. Ninguna de las anteriores.

Pregunta 7: Indica cual de las siguientes afirmaciones es correcta.

- a. Declarar un campo o un método protegido (protected) permite acceder directamente a él desde las subclases directas o indirectas.
- b. Declarar un campo o un método protegido (protected) permite acceder directamente a él únicamente desde las subclases directas.
- c. Los miembros definidos como private en una subclase son accesibles para los objetos de otras clases.
- d. Los miembros definidos como private en una superclase son accesibles para los objetos de sus subclases.

Pregunta 8: ¿Cuál es el resultado de compilar y ejecutar el siguiente código?

```
public class Padre
{
    void metodoA(){
        System.out.println("Ejecuto el método A de la clase Padre");
    }
}
public class Hijo extends Padre
{
    void metodoA(){
        System.out.println("Ejecuto el método A de la clase Hijo");
    }
}
public class Test {
    public static void main(String[] args) {
        Hijo h=(Hijo) new Padre();
        if(h instanceof Hijo)
```

```

        System.out.println("Instancia de la clase Hijo");
    else
        System.out.println("Instancia de la clase padre");
    }

}

```

- a. Muestra en consola: "Instancia de la clase Hijo "
- b. Muestra en consola: "Instancia de la clase Padre"
- c. Error de compilación. Tipos incompatibles.
- d. Error en tiempo de ejecución. Lanza una excepción ClassCastException.

Pregunta 9: ¿Cuál es el resultado de compilar y ejecutar el siguiente código?

```

public class Padre
{
    void metodoA(){
        System.out.print("Ejecuto el método A de la clase Padre. ");
    }
}
public class Hijo extends Padre
{
    void metodoA(){
        System.out.print("Ejecuto el método A de la clase Hijo. ");
        super.metodoA();
    }
}
public class Test {
    public static void main(String[] args) {
        Padre p=new Hijo();
        p.metodoA();
    }
}

```

- a. Muestra en consola: "Ejecuto el método A de la clase Padre. "
- b. Muestra en consola: "Ejecuto el método A de la clase Hijo."
- c. Muestra en consola: "Ejecuto el método A de la clase Padre. Ejecuto el método A de la clase Hijo.
- d. Ninguna de las anteriores.

Pregunta 10: ¿Cuál es el resultado de compilar y ejecutar el siguiente código?

```

public abstract class Padre
{
    abstract void metodoA();
}
public class Hijo extends Padre
{
    void metodoA(){
        System.out.print("Método A de Hijo. ");
    }
}
public class Hija extends Padre
{
    void metodoA(){
        System.out.print("Método A de Hija. ");
    }
}
import java.util.*;
public class Test {
    public static void main(String[] args) {

```

```

        List<Padre> lista=new ArrayList<Padre>();
        lista.add(new Hijo());
        lista.add(new Hijo());
        lista.add(new Hija());
        for(Iterator<Padre> it=lista.iterator();it.hasNext();) {
            Padre p=it.next();
            p.metodoA();
        }
    }
}

```

- a. Muestra en consola: "Método A de Hijo. Método A de Hijo. Método A de Hija."
- b. Error de compilación. No se puede incluir un iterador dentro de un bucle for.
- c. Error de compilación. Las clases Hijo e Hija deben declararse como abstractas.
- d. Error en tiempo de ejecución. Es obligatorio hacer un cast en cada elemento extraído del ArrayList para invocar al método correcto

Pregunta 11: ¿Cuál es el resultado de compilar y ejecutar el siguiente código?

```

public abstract class Padre
{
    void metodoA() {
        this.metodoB();
    }
    abstract void metodoB();
}
public class Hijo extends Padre
{
    void metodoB() {
        System.out.print("Método B de Hijo. ");
    }
}
import java.util.*;
public class Test {
    public static void main(String[] args) {
        Hijo h=new Hijo();
        h.metodoA();
    }
}

```

- a. Muestra en consola: "Método B de Hijo. "
- b. Error de compilación. No se puede invocar un método abstracto desde la clase abstracta Padre
- c. Error de compilación. La clase abstracta Padre debe declarar todos sus métodos como abstractos.
- d. Ninguna de las anteriores.

Pregunta 12: ¿Cuál de las siguientes afirmaciones es falsa?

- a. Una interfaz en Java es una especificación de un tipo que no define implementación para alguno de sus métodos.
- b. Para que una subclase de una clase abstracta se transforme en concreta, debe proporcionar implementaciones para todos los métodos abstractos heredados.
- c. El objetivo de una clase abstracta es servir como una superclase de otras clases.
- d. Las llamadas a métodos de instancia no privados desde dentro de una superclase siempre se evalúan en el contexto más amplio del tipo dinámico del objeto.

Pregunta 13: ¿Cuál de las siguientes afirmaciones es falsa?

- a. Las clases internas anónimas son una estructura muy útil a la hora de implementar escuchas de sucesos.

- b. Una interfaz GUI se construye disponiendo componentes en pantalla. Los componentes se representan mediante objetos.
- c. Para definir la colocación de los componentes de una GUI se utilizan gestores de diseño gráfico.
- d. Un objeto puede escuchar los sucesos de los componentes extendiendo una clase abstracta de escucha de sucesos.

Pregunta 14: ¿Cuál de las siguientes afirmaciones es falsa?

- a. La serialización permite leer y escribir en una única operación objetos completos, pero no jerarquías de objetos.
- b. Una excepción no comprobada es un tipo de excepción cuyo uso no requiere ninguna comprobación por parte del compilador.
- c. Una excepción comprobada es un tipo de excepción cuyo uso requiere comprobaciones adicionales por parte del compilador.
- d. Una aserción es un enunciado de un hecho que debe ser cierto durante la ejecución del programa.

Pregunta 15: ¿Cuál de las siguientes afirmaciones es falsa?

- a. Las pruebas son la actividad consistente en averiguar si un fragmento de código presenta el comportamiento deseado.
- b. Un recorrido manual es la actividad consistente en analizar un segmento de código línea a línea mientras que se observan los cambios de estado y otros comportamientos de la aplicación.
- c. Una prueba negativa es una prueba de un caso que se espera que funcione correctamente y que finalmente no funciona.
- d. Si la condición definida en una aserción es falsa, decimos que la aserción ha fallado.

PARTE PRÁCTICA [6,5 PUNTOS]:

La práctica del presente curso ha sido una terminal punto de venta (por sus siglas, TPV) que ha servido para estudiar y practicar los mecanismos de la Programación Orientada a Objetos.

Definición de TPV y Características

Según la Wikipedia (www.wikipedia.org), un terminal punto de venta (cuyo acrónimo es TPV) hace referencia al dispositivo y tecnologías que ayudan en la tarea de gestión de un establecimiento comercial de venta al público que puede contar con sistemas informáticos especializados mediante una interfaz accesible para los vendedores.

Los TPV permiten la creación e impresión del tique de venta mediante las referencias de productos, realizan diversas operaciones durante todo el proceso de venta, así como cambios en el inventario. También generan diversos reportes que ayudan en la gestión del negocio. Los TPV se componen de una parte hardware (dispositivos físicos) y otra software (sistema operativo y programa de gestión).

En nuestro caso concreto, el hardware será un ordenador tipo PC o similar y nuestro software será una aplicación desarrollada en Java que se ejecutará sobre dicho equipo.

Funcionalidades

Los TPV permiten la implementación desde labores simples de gestión de una venta, hasta operaciones más complejas como es la gestión de almacén o inventario, gestión de facturación o gestión de clientes. En esta práctica, se propondrá diferentes funcionalidades para el sistema de gestión del TPV:

- Llevar un control de diferentes elementos que existen en nuestro establecimiento. Así, los

productos habrán de estar identificados en el sistema por, al menos, los siguientes datos: código descriptivo (por ejemplo, el código de barras), descripción, precio unitario sin IVA, IVA aplicable, precio unitario con IVA, cantidad disponible en stock.

- El sistema debe permitir dar de alta nuevos productos, dar de baja productos existentes así como modificar los datos del mismo.
- Realizar la importación y/o exportación de los productos a/desde ficheros (u otro método similar que el alumno considere en su lugar).
- Llevar un control de las diferentes ventas que se producen. Así, el sistema deberá llevar un control de tickets generados, de modo que cada ticket se considerará una venta. Cada ticket tiene que tener un código de identificador único. Una forma de generar un código único podría ser de la forma AAAAMMDDHHMM, donde AAAA es el año en curso, MM el mes en que se genera la venta, DD el día de la venta, HHMM las horas y minutos en las que se inicia la venta. Asumiremos que sólo hay un TPV, por lo que no procede que haya dos ventas simultáneas.
- La venta consistirá en la inclusión de varios productos en una lista, generándose una línea por cada producto vendido. Cada línea mostrará, al menos, el código del producto, la descripción del producto, la cantidad de unidades vendidas, el precio unitario con IVA, el IVA que se le aplica y el importe total de la venta de ese producto según el número de unidades vendidas.
- El proceso de venta implicará automáticamente un proceso de actualización del inventario. De este modo, si se introduce un código que no pertenece a ningún producto, o si se introduce un producto que no existe en stock (o más unidades de las existentes), el programa deberá mostrar los errores correspondientes.
- El sistema deberá permitir también introducir un producto a vender en el ticket haciendo una búsqueda por la descripción, además de con el código que lo identifica.
- Realizar la importación y/o exportación de los diferentes tickets de ventas a/desde ficheros (u otro método similar que el alumno considere en su lugar).
- Llevar un control de los diferentes clientes que trabajan con el establecimiento comercial. Así, los clientes habrán de estar identificados en el sistema por, al menos, los siguientes datos: código identificativo del cliente, NIF o CIF, nombre y apellidos / razón social, domicilio, fecha de alta en el sistema.
- El sistema debe permitir dar de alta nuevos clientes, dar de baja clientes existentes así como modificar los datos de los mismos.
- Realizar la importación y/o exportación de los clientes a/desde ficheros (u otro método similar que el alumno considere en su lugar).
- Permitir generar facturas a partir de un conjunto de tickets. Puede generar facturas agrupando diferentes tickets siempre y cuando pertenezcan al mismo cliente y se han realizado dentro del mismo periodo fiscal (es decir, dentro del mismo año). La información que irá en cada factura deberá ser, al menos, la siguiente: número de la factura (identificador único), CIF del vendedor, razón social del vendedor, fecha de emisión de la factura, datos del cliente (los indicados con anterioridad, excepto la fecha de alta en el sistema), listado de los diferentes productos vendidos (especificando para cada producto, el ticket en el que se encuentra, su cantidad vendida e importe total) así como suma del total de la venta (valor total de la factura).
- Realizar la importación y/o exportación de las facturas a/desde ficheros (u otro método similar que el alumno considere en su lugar).
- Generación de listados: se deberá implementar, al menos, la emisión de tres listados, a saber: ventas realizadas en un intervalo de tiempo determinado agrupadas estas ventas por clientes, ventas realizadas en un intervalo de tiempo determinado a un cliente y ranking de productos más vendidos en un intervalo de tiempo determinado.

- a) **[1,0 puntos]** Diseñar utilizando un paradigma orientado a objetos, los elementos necesarios para la aplicación explicada de la práctica durante el curso. Es necesario identificar la estructura y las relaciones de herencia (mediante el uso de un diagrama de clases) y de uso de las clases necesarias para almacenar y gestionar esta información. Debe hacerse uso de los mecanismos de herencia siempre que sea posible. Se valorará un buen diseño que favorezca la reutilización de código y facilite su mantenimiento.
- b) **[1,0 puntos]** Modifique aquellas clases que considere oportunas para incluir la funcionalidad de tener clientes VIP. A estos clientes se les aplicará un descuento del 10% en todos los productos.
- c) **[2,0 puntos]** Modifique los métodos de venta y generación de facturas para aplicar el descuento en el precio final de cada producto en el caso en el que el cliente sea VIP.
- d) **[2,5 puntos]** Proporcione un método (o métodos) que permitan mostrar por pantalla un listado, en **modo gráfico**, de los clientes VIPs ordenados de mayor a menor en función del gasto realizado.

**UNIVERSIDAD NACIONAL DE EDUCACIÓN A DISTANCIA – ESCUELA TÉCNICA SUPERIOR DE
INGENIERÍA INFORMÁTICA**
**71901072 – PROGRAMACIÓN ORIENTADA A OBJETOS (GRADO EN INGENIERÍA INFORMÁTICA /
TECNOLOGÍAS DE LA INFORMACIÓN)**
JUNIO 2016 – MODELO B – NO ESTÁ PERMITIDO EL USO DE MATERIAL ADICIONAL

PARTE TEÓRICA - TEST [2,5 PUNTOS]:

El test consta de 14 preguntas y 2 preguntas adicionales de reserva. Solo una de las respuestas es válida. Las respuestas correctas se puntuarán con +1.0, mientras que las respondidas de manera incorrecta se puntuarán con -0.25. Las no contestadas no tendrán influencia ni positiva ni negativa en la nota.

Las preguntas de reserva sólo tendrán utilidad en el caso de que alguna de las 14 preguntas iniciales del test sea anulada por cualquier circunstancia. Caso de ocurrir este hecho, si se produjera la anulación de alguna de las 14 preguntas iniciales, la primera pregunta de reserva sustituiría a la pregunta anulada. Caso de que una segunda pregunta de las 14 iniciales fuese anulada, entonces la segunda pregunta de reserva sustituiría a esta segunda pregunta anulada. En aquellos hipotéticos casos en los que se produjese la anulación de una tercera o sucesivas preguntas de las 14 iniciales, entonces sólo en ese caso, las preguntas tercera y sucesivas anuladas se considerarían como correctas (al no existir más preguntas de reserva que las sustituyan).

Pregunta 1: Según el texto de la bibliografía básica de la asignatura, un ArrayList tiene las siguientes características (indica la respuesta que **NO** es correcta):

- a. Puede incrementar su capacidad interna si es necesario.
- b. Mantiene su propio contador del número de elementos almacenados.
- c. Mantiene el orden de almacenamiento de los elementos.
- d. No permite la duplicación de objetos.

Pregunta 2: Según el texto de la bibliografía básica de la asignatura, ¿cuál de las siguientes clases **NO** es una colección?

- a. ArrayList
- b. HashSet
- c. HashMap
- d. ArraySet

Pregunta 3: La presencia de dos o más constructores en una clase se llama:

- a. Herencia
- b. Abstracción
- c. Sobrecarga
- d. Métodos internos

Pregunta 4: Dado el siguiente código:

```
1 import java.util.ArrayList;
2 class Persona {
3     private String nombre; private String telefono;
4     public Persona(String n, String t) {
5         nombre = n; telefono = t;
6     }
7     public String getNombre() {
8         return(nombre);
9     }
10 }
11 public class Prueba {
12     private ArrayList<Persona> amigos = new ArrayList<Persona>();
13
14     public Prueba() {
15         amigos.add(new Persona("Pepe","1234"));
16         amigos.add(new Persona("Sara","1236"));
17     }
}
```

```

18     public void borrarAmigo(String nombre) {
19         for (Persona p: amigos) {
20             if (p.getNombre().equals(nombre)) {
21                 amigos.remove(p);
22                 System.out.println("Amigo borrado");
23             }
24         }
25     }
26 }
```

¿Cuál es el resultado de compilar / llamar el método `borrarAmigo ("Sara")`?

- a. Se produce un error de compilación.
- b. Se produce una excepción durante la ejecución
- c. Se produce un aviso(warning) durante la ejecución
- d. Se imprime por pantalla el mensaje: Amigo borrado y no se genera error o excepción alguno

Pregunta 5: Según el texto de la bibliografía básica de la asignatura, el estado de un objeto se denomina:

- a. El conjunto de parámetros que recibe y devuelve un objeto.
- b. El conjunto de valores de todos los atributos que definen al objeto.
- c. El conjunto de campos y métodos que componen el objeto.
- d. Ninguna de las anteriores.

Pregunta 6: ¿Cómo se activan por primera vez las herramientas de prueba de JUnit en BlueJ?

- a. No es necesario, ya vienen activadas.
- b. Con el botón derecho, seleccionando Activar (*Activate*) JUnit
- c. A través de la pestaña Miscelánea (*Miscellaneous*) del cuadro de diálogo Preferencias (*Preferences*).
- d. A través del menú.

Pregunta 7: Según el texto de la bibliografía básica de la asignatura, ¿qué es una aserción?:

- a. Una expresión que establece una condición que esperamos que sea cierta.
- b. Una expresión que resume la función de un método.
- c. Una instrucción que prueba la validez de una función.
- d. Ninguna de las respuestas anteriores.

Pregunta 8: Si simplificamos el ejemplo PhotoPost del libro de la asignatura de la siguiente manera:

```

1  public class Post {
2      private String usuario;
3
4      public Post(){}
5      public Post(String autor) {
6          usuario = autor;
7      }
8  }
9  class PhotoPost extends Post {
10     private String file;
11
12     public PhotoPost(String autor, String file) {
13         XXX
14         this.file = file;
15     }
16 }
```

¿Qué tendremos que añadir a la línea 13, en vez de XXX, para asignar el valor de autor a variable usuario en la clase Post?:

- a. super();
- b. usuario = autor;
- c. super(autor);
- d. super.usuario = autor;

Pregunta 9: Según el texto de la bibliografía básica de la asignatura, caracterizan los campos, constructores y métodos de la siguiente forma:

1. ... implementan el comportamiento de un objeto.
2. ... almacenan datos de manera persistente dentro de un objeto.
3. ... son responsables de garantizar que un objeto se configure apropiadamente a crearlo por primera vez.

¿Qué definición corresponde con que término?

- a. Campos = 1, Constructores = 2, Métodos = 3.
- b. Campos = 2, Constructores = 1, Métodos = 3.
- c. Campos = 2, Constructores = 3, Métodos = 1.
- d. Campos = 3, Constructores = 2, Métodos = 1.

Pregunta 10: Dado el siguiente fragmento de código del ejemplo del libro *ImageViewer*:

```

1  private void makeMenuBar(JFrame frame) {
2      final int SHORTCUT_MASK =
3          Toolkit.getDefaultToolkit().getMenuShortcutKeyMask();
4
5      JMenuBar menubar = new JMenuBar();
6      frame.setJMenuBar(menubar);
7
8      JMenu menu;
9      JMenuItem item;
10
11     // generar menú de preferencias
12     XXX
...

```

Si se quisiera añadir una nueva entrada en la barra de menús con el nombre Preferencias, ¿que habrá que añadir en la línea 13 en vez de XXX?:

- a. menu = new JMenuItem("Preferencias"); menubar.add(menu);
- b. menu = new JMenu("Preferencias"); frame.add(menu);
- c. menu = (Jmenu)new JMenuItem("Preferencias"); menubar.add(menu);
- d. menu = new JMenu("Preferencias"); menubar.add(menu);

Pregunta 11: Según el texto de la bibliografía básica de la asignatura, para crear una representación de un objeto en forma de String tenemos que implementar el método:

- a. getString()
- b. toString()
- c. printString()
- d. parseString()

Pregunta 12: Se quiere proporcionar dos constructores en la clase del reloj ClockDisplay del libro para iniciar el reloj de dos formas diferentes (fijando las horas y los minutos y fijando los minutos y los segundos):

```

1  public ClockDisplay(int hour, int minute)
2  {
3      hours = new NumberDisplay(24);
4      minutes = new NumberDisplay(60);
5      seconds = new NumberDisplay(60);
6      setTime(hour, minute, second);
7      this.hour = hour;
8      this.minute = minute;
9      this.second = 0;
10 }
11
12 public ClockDisplay(int minute, int second)
13 {
14     hours = new NumberDisplay(24);
15     minutes = new NumberDisplay(60);
16     seconds = new NumberDisplay(60);
17     setTime(hour, minute, second);
18     this.hour = 0;
19     this.minute = minute;

```

```
20         this.second = second;
21     }
```

¿Cuál es el resultado de compilar / llamar al constructor con los valores 2 y 15?

- a. Generamos un objeto ClockDisplay con la hora 02:15:00
- b. Generamos un objeto ClockDisplay con la hora 00:02:15
- c. Tanto la respuesta a como la b son posibles.
- d. Se produce un error de compilación en la línea 12.

Pregunta 13: Para definir una nueva clase de excepción, como se ha hecho en el ejemplo *AddressBook* del libro, se hace de la siguiente forma:

```
1  public class NoMatchingDetailsException extends XXX {
2      private String key;
3
4      public NoMatchingDetailsException(String key) {
5          this.key = key;
6      }
...

```

¿Qué habrá que añadir en la línea 13 en vez de XXX para declarar correctamente la nueva clase?:

- a. RunException
- b. Exception
- c. IOException
- d. java.util.Exception

Pregunta 14: Si se quiere insertar el siguiente método en la clase de la máquina expendedora de billetes, ¿cuál es el resultado de compilar / ejecutar el método con un valor de coste de 50?

```
1  public void probarDinero(int dinero)
2  {
3      if (dinero = 50) {
4          System.out.println("No se admiten billetes de 50€.");
5      }
6      else {
7          System.out.println("Impresión de billete en curso.");
8      }
9 }
```

- a. Se produce un error de compilación en la línea 3.
- b. Se produce un error de ejecución en la línea 4.
- c. Se imprime por pantalla el mensaje: No se admiten billetes de 50€.
- d. Se imprime por pantalla el mensaje: Impresión de billete en curso.

Reserva 1: Supongamos que reescribimos una parte de la simulación de los zorros y los conejos del libro de la forma que se muestra a continuación:

```
1  import java.util.List;
2  public abstract class Animal {
3
4      public Animal(){}
5
6      abstract public void act(List<Animal> newAnimals);
7      protected void isAlive(){}
8      protected void setDead(){}
9  }
10
11  class Rabbit extends Animal {
12
13      XXX
14      private void incrementAge(){}
15      private void giveBirth(List<Animal> newRabbits){}
16  }
```

¿Qué tendremos que añadir a la línea 13, en vez de XXX, para que el código no genere un error de

compilación?:

- a. public Rabbit();
- b. public void act(List<Animal> newRabbits) {}
- c. public void act(List<Animal> newRabbits);
- d. El código se compilará sin ningún error de todas formas.

Reserva 2: Según el texto de la bibliografía básica de la asignatura, ¿cómo se usa *instanceof* para averiguar si el tipo dinámico de un objeto (miobj) es de una cierta clase (MiClase)?

- a. MiClase.instanceof(miobj)
- b. java.lang.Object.instanceof(miobj, MiClase)
- c. miobj instanceof MiClase
- d. Object.instanceof(miobj, MiClase)

PARTE PRÁCTICA [6,5 PUNTOS]:

La Práctica del presente curso va a consistir en diseñar e implementar un sistema integrado de gestión de una biblioteca (a partir de ahora, SIGB). Han existido versiones sencillas de estos sistemas incluso antes de la existencia de los computadores, donde se almacenaba información sobre los materiales de la biblioteca (por aquél entonces, libros, revistas, periódicos) en fichas en formato papel, guardadas en cajones clasificados. El primer paso hacia la informatización de estos sistemas, hacia lo que hoy en día es un SIGB, tuvo lugar en 1936 en la Universidad de Texas, donde la información sobre los libros estaba representada en tarjetas perforadas y cargada en su computador central. Con el avance de la informática a lo largo de los años, el manejo de los recursos de una biblioteca, a través de sistemas integrados de gestión, se ha podido llevar a cabo con una amplia gama de dispositivos, desde terminales tontas hasta teléfonos móviles.

Funcionalidades

Un SIGB proporciona las siguientes funcionalidades:

- Añadir nuevos materiales a la colección de la biblioteca (rellenando los datos de un formulario). Cada tipo de material debería tener su propia colección (libros, revistas, periódicos, audio, video, etc.).
- Borrar materiales de la colección.
- Realizar búsquedas sencillas sobre los materiales.
- Gestionar suscripciones a revistas y periódicos.
- Gestión de usuarios: altas, bajas, generación de tarjetas, historiales de préstamo, control de acceso (diferenciar entre dos perfiles: usuarios y bibliotecarios).
- Realización básica de Préstamos: prestar un material si está disponible en la biblioteca, asignar fechas de devolución.
- Producir listados de préstamos según el tipo de material.
- Realizar búsquedas flexibles sobre los materiales en la biblioteca combinando varios campos de búsqueda.
- Control de préstamos: número máximo de ítems de préstamo (6 por usuario, independiente de tipo de material), emisión de avisos de materiales fuera de plazo, gestión de multas, etc.
- Producir listados de los materiales prestados.
- Realizar búsquedas flexibles sobre los materiales en varias bibliotecas a la vez combinando varios campos de búsqueda.
- Préstamos entre bibliotecas: poder solicitar materiales a otras bibliotecas y procesar las solicitudes de otras bibliotecas. El procesamiento de dichas solicitudes se lleva a cabo usando archivos de solicitud de la siguiente manera:
 - Preparar y exportar una lista de solicitudes de materiales que se quiere hacer a una biblioteca. Se prepara la lista usando un formulario para identificar el nombre de la biblioteca, el nombre del libro, el autor y el nombre de esta biblioteca. Una vez

terminado, se guardar la lista en un archivo de texto. No es necesario en esta práctica preocuparse de cómo se enviaría el archivo a otras bibliotecas.

- Importar y procesar un archivo de solicitudes para materiales proveniente de otra biblioteca. Se debe actualizar el estatus de cada material para marcarse como prestado, pero en vez del identificador del usuario debería aparecer el identificador de la biblioteca.
 - Control de reservas: poder reservar un material si está ya prestado, gestión de avisos (al usuario con el material que convendría devolverlo porque hay alguien esperando y al usuario con la reserva cuando el material ya esta devuelto).
- a) **[1 punto]** Diseñar utilizando el paradigma orientado a objetos, los elementos necesarios para la aplicación explicada de la práctica durante el curso. Es necesario identificar la estructura y las relaciones de herencia (mediante el uso de un diagrama de clases) y de uso de las clases necesarias para almacenar y gestionar esta información. Debe hacerse uso de los mecanismos de herencia siempre que sea posible. Se valorará un buen diseño que favorezca la reutilización de código y facilite su mantenimiento.
- b) **[1 punto]** Implementar el método que permite producir listados de préstamos según el tipo de material. Justifíquese las opciones y decisiones que se tomen.
- c) **[2 puntos]** Implementar la funcionalidad de la preparación y exportación de una lista de solicitudes que se quiere hacer a una biblioteca como parte del préstamo entre bibliotecas. Justifíquese las opciones y decisiones que se tomen.
- d) **[2,5 puntos]** Se quiere añadir una cafetería a la biblioteca para que los clientes puedan tomar algo mientras miran los libros, revistas o periódicos. Se quiere utilizar el sistema de gestión de la biblioteca para gestionar la compra y venta de los productos de la cafetería. Indique los cambios que serían necesarios en el diseño y la implementación para permitir esa nueva funcionalidad.

**UNIVERSIDAD NACIONAL DE EDUCACIÓN A DISTANCIA – ESCUELA TÉCNICA SUPERIOR DE
INGENIERÍA INFORMÁTICA**
**71901072 – PROGRAMACIÓN ORIENTADA A OBJETOS (GRADO EN INGENIERÍA INFORMÁTICA /
TECNOLOGÍAS DE LA INFORMACIÓN)**
JUNIO 2016 – MODELO C – NO ESTÁ PERMITIDO EL USO DE MATERIAL ADICIONAL

PARTE TEÓRICA - TEST [2,5 PUNTOS]:

El test consta de 14 preguntas y 2 preguntas adicionales de reserva. Solo una de las respuestas es válida. Las respuestas correctas se puntuarán con +1.0, mientras que las respondidas de manera incorrecta se puntuarán con -0.25. Las no contestadas no tendrán influencia ni positiva ni negativa en la nota.

Las preguntas de reserva sólo tendrán utilidad en el caso de que alguna de las 14 preguntas iniciales del test sea anulada por cualquier circunstancia. Caso de ocurrir este hecho, si se produjera la anulación de alguna de las 14 preguntas iniciales, la primera pregunta de reserva sustituiría a la pregunta anulada. Caso de que una segunda pregunta de las 14 iniciales fuese anulada, entonces la segunda pregunta de reserva sustituiría a esta segunda pregunta anulada. En aquellos hipotéticos casos en los que se produjese la anulación de una tercera o sucesivas preguntas de las 14 iniciales, entonces sólo en ese caso, las preguntas tercera y sucesivas anuladas se considerarían como correctas (al no existir más preguntas de reserva que las sustituyan).

Pregunta 1: ¿De qué forma podemos invocar un método en el IDE BlueJ?

- a. Haciendo clic con el botón derecho del ratón en un objeto, y seleccionando en el menú emergente el método correspondiente.
- b. Haciendo clic con el botón derecho del ratón en una clase, y seleccionando en el menú emergente el método correspondiente.
- c. Haciendo clic con el botón derecho del ratón en un objeto o clase indistintamente, y seleccionando en el menú emergente el método correspondiente.
- d. Ninguna de las anteriores.

Pregunta 2: Según el texto de la bibliografía básica de la asignatura, ¿cuál de las siguientes afirmaciones **NO** es correcta?

- a. Los campos almacenan datos de manera no persistente dentro de un objeto.
- b. Los constructores son responsables de garantizar que un objeto se configure apropiadamente en el momento de crearlo por primera vez.
- c. Los métodos implementan el comportamiento de un objeto; proporcionan su funcionalidad.
- d. Ninguna de las anteriores.

Pregunta 3: Queremos compilar el siguiente código que se puede encontrar en el texto base de la asignatura y que ha sido modificado convenientemente. ¿Cuál es el resultado que obtenemos al compilar?

```
1  public class Prueba
2  {
3      public static void main (String args[])
4      {
5          String cadena1 = "ejemPLO";
6          String cadena2 = "EJEMPLO";
7          cadena1.toUpperCase();
8
9          if (cadena1.equals(cadena2))
10         {
11             System.out.println("Son iguales");
12         }
13     else
14     {
15         System.out.println("Son diferentes");
16     }
}
```

```
17      }
18  }
```

- a. Se produce una excepción y la ejecución falla.
- b. Se imprime por pantalla el mensaje: Son iguales.
- c. Se imprime por pantalla el mensaje: Son diferentes.
- d. Ninguna de las anteriores.

Pregunta 4: Según el texto de la bibliografía básica de la asignatura, ¿qué debilitaría la encapsulación?

- a. Emplear el acceso protegido a los métodos de una clase.
- b. Emplear el acceso protegido a los campos de una clase.
- c. Emplear el acceso protegido a los constructores de una clase.
- d. Ninguna de las anteriores.

Pregunta 5: Supongamos que reescribimos el ejemplo BouncingBall del libro de la forma en que se muestra a continuación:

```
1  public class BouncingBall {
2      int n;
3      public static void main (String args [])
4      {
5
6          if (n!=0)
7          {
8              n = n + 1;
9              System.out.println("El número es " + n);
10         }
11     }
12 }
```

El programa no compila. ¿Qué podemos cambiar para que funcione correctamente?

- a. Sustituir la línea 2 por la siguiente: int n = 0;
- b. Insertando en la línea 5 lo siguiente: n = 0;
- c. Sustituir la línea 3 por lo siguiente: public static void main ()
- d. Ninguna de las anteriores.

Pregunta 6: Sea el siguiente código modificado de la clase MusicOrganizer mostrada en el libro base:

```
1  import java.util.*;
2  public class MusicOrganizer {
3      public static void main (String args [])
4      {
5          ArrayList <String> a = new ArrayList (5);
6          for (int i=0; i<=5; i++)
7          {
8              a.add("Hola");
9          }
10         System.out.println("Funciona");
11     }
12 }
```

La compilación produce un warning. ¿Cómo podemos resolver ese problema?

- a. Sustituyendo la línea 4 por: ArrayList a = new ArrayList (5);
- b. Sustituyendo la línea 4 por: ArrayList a = new ArrayList <String> (5);
- c. Sustituyendo la línea 4 por: ArrayList <String> a = new ArrayList <String> (5);
- d. Tanto (a) como (c) resuelven el problema.

Pregunta 7: Queremos compilar el siguiente código que se puede encontrar en el texto base de la asignatura,

convenientemente modificado. ¿Qué ocurre al compilarlo con BlueJ?

```
1 import java.util.List;
2 public abstract class Animal
3 {
4     private boolean alive;
5     private String field;
6     private String location;
7
8     public abstract Animal(String field, String location)
9     {
10         alive = true;
11         this.field = field;
12         this.location = location;
13     }
14
15     abstract public void act(List<Animal> newAnimals);
16 }
```

- a. Compila, no proporcionando ningún error en tiempo de compilación.
- b. No compila. Se soluciona sustituyendo la línea 15 por la siguiente: public void act(List<Animal> newAnimals);
- c. No compila. Se soluciona sustituyendo la línea 15 por la siguiente: public abstract void act(List<Animal> newAnimals);
- d. Ninguna de las anteriores.

Pregunta 8: Según el texto de la bibliografía básica de la asignatura, las subclases de Error suelen estar reservadas para ...

- a. Los errores del sistema en tiempo de ejecución.
- b. Los errores del sistema en tiempo de compilación.
- c. Los errores de programación en tiempo de compilación.
- d. Ninguna de las anteriores.

Pregunta 9: Según el texto de la bibliografía básica de la asignatura, ¿cuál es el método imprescindible y que ha de implementarse siempre de la interfaz Serializable cuando queremos implementar la serialización?

- a. El método IOWrite.
- b. El método InputOutputWrite.
- c. El método WriteOutput.
- d. Ninguno de los anteriores.

Pregunta 10: Queremos compilar el siguiente código que se puede encontrar en el texto base de la asignatura y que ha sido convenientemente modificado. El programa compila sin problemas pero no muestra por pantalla el texto “Ejemplo de texto”. ¿Qué tendríamos que añadir / modificar para se mostrase?

```
1 import java.awt.*;
2 import java.awt.event.*;
3 import javax.swing.*;
4
5 public class ImageViewer
6 {
7     private JFrame frame;
8     public ImageViewer(){
9         makeFrame();
10    }
11}
```

```

12     private void makeFrame() {
13         frame = new JFrame("ImageViewer");
14         Container contentPane = frame.getContentPane();
15         JLabel label = new JLabel("Ejemplo de texto");
16         contentPane.add(label);
17         frame.setVisible(true);
18     }
19 }
```

- a. Añadir entre las líneas 9 y 10 lo siguiente: `frame.setVisible(true);` y eliminar la línea 17.
- b. Añadir entre las líneas 15 y 16 lo siguiente: `frame.pack();`
- c. Añadir entre las líneas 16 y 17 lo siguiente: `frame.pack();`
- d. Sustituir la línea 17 por lo siguiente: `frame.pack();`

Pregunta 11: Según el texto de la bibliografía básica de la asignatura, ¿qué define el término acoplamiento?

- a. La bondad de la correspondencia entre una unidad de código y una tarea lógica o entidad.
- b. La interconexión existente entre clases, buscando un acoplamiento lo menor posible.
- c. La capacidad de un objeto de comportarse como otra clase de la cual proviene.
- d. Ninguna de las anteriores.

Pregunta 12: Sea el siguiente fragmento de código modificado de la clase `MailItem` mostrada en el libro de texto:

```

1  public class MailItem {
2      static String from;
3      static String to;
4      static String message;
5      int number;
6
7      public static void main (String args[]) {
8          MailItem m = new MailItem("Hola", "Adios", "Luego", 3);
9          System.out.println("Funciona");
10     }
11
12     public MailItem (String from, String to, String message, int number) {
13         this.from = from;
14         this.to = to;
15         this.message = message;
16         this.number = number;
17     }
18 }
```

¿Cuál es el resultado de ejecutar el código?

- a. Se produce un error de compilación
- b. Se produce un error de ejecución en la línea 9.
- c. Se produce un error de ejecución en la línea 16.
- d. Ninguna de las anteriores.

Pregunta 13: Según el texto de la bibliografía básica de la asignatura, ¿qué podemos afirmar sobre las pruebas de regresión?

- a. La modificación de software acarrea con mucha facilidad errores adicionales de software.
- b. Las pruebas de regresión sobre un módulo determinado tras haberse hecho una modificación del código pueden obviarse si no se realizan cambios en ese módulo.
- c. Los marcos de regresión permiten automatizar las pruebas de regresión.
- d. Si no se automatizan, es más probable que las pruebas de regresión se lleven a cabo.

Pregunta 14: Según el texto de la bibliografía básica de la asignatura, ¿qué podemos afirmar sobre el concepto

de sustitución?

- a. Pueden utilizarse objetos de un supertipo en cualquier lugar en el que se espera objetos de un subtipo.
- b. Permite crear objetos de un clase que es abstracta.
- c. Permite que una variable almacena objetos de diferentes tipos (en concreto, del tipo declarado o de cualquier supertipo del tipo declarado).
- d. Ninguna de las anteriores

RESERVA 1: Si una clase B extiende una clase abstracta A que tiene un método abstracto `met`, ¿qué podemos afirmar?

- a. Que necesariamente B es abstracta.
- b. Que si B implementa el método `met`, entonces seguro que B no es abstracta.
- c. Que no se pueden crear instancias de A.
- d. Que puedo crear instancias de A.

RESERVA 2: Según el texto de la bibliografía básica de la asignatura, ¿qué puede usarse para generar la descripción de las interfaces de las clases a partir del código fuente?

- a. JDK
- b. JUnit
- c. Code Pad
- d. Ninguna de las anteriores

PARTE PRÁCTICA [6,5 PUNTOS]:

La Práctica del presente curso es diseño e implementación de un sistema integrado de gestión de una biblioteca (a partir de ahora, SIGB). En general, las funciones que tienen un SIGB son varias según el perfil de su usuario (que va desde el usuario de la biblioteca hasta su director) e incluyen las siguientes:

- Adquisiciones: la compra de materiales (libros en diferentes formatos, audiolibros, CDs de música, películas en DVD, etc.), gestión de compras, facturación, etc.
- Catalogar: la clasificación e indexación de los materiales de la biblioteca.
- Préstamos: prestar los materiales a los usuarios (tanto en papel como en otros formatos), reservas de materiales ya en préstamos, control de préstamos (emisión de avisos de materiales fuera de plazo), gestión de multas.
- Suscripciones: gestión de las suscripciones a revistas y periódicos.
- Catálogo en línea u OPAC (del inglés Online Public Access Catalog): interfaz pública a los servicios de la biblioteca (búsquedas, gestión de préstamos, etc.).
- Gestión de usuarios: altas, bajas, generación de tarjetas, historiales.

Funcionalidades

Los SIGB permiten la implementación desde labores simples de gestión de una alta de usuario, hasta operaciones más complejas como es la gestión de préstamos o inventario. En esta práctica, se propondrán diferentes funcionalidades para el sistema de gestión bibliotecaria:

- Añadir nuevos materiales a la colección de la biblioteca (rellenando los datos de un formulario). Cada tipo de material debería tener su propia colección (libros, revistas, periódicos, audio, video, etc.).
- Borrar materiales de la colección.
- Realizar búsquedas sencillas sobre los materiales.
- Gestionar suscripciones a revistas y periódicos.
- Gestión de usuarios: altas, bajas, generación de tarjetas, historiales de préstamo, control de acceso (diferenciar entre dos perfiles: usuarios y bibliotecarios).
- Realización básica de Préstamos: prestar un material si está disponible en la biblioteca, asignar fechas de devolución.

- Producir listados de préstamos según el tipo de material.
 - Realizar búsquedas flexibles sobre los materiales en la biblioteca combinando varios campos de búsqueda.
 - Control de préstamos: número máximo de ítems de préstamo (6 por usuario, independiente de tipo de material), emisión de avisos de materiales fuera de plazo, gestión de multas, etc.
 - Producir listados de los materiales prestados.
 - Realizar búsquedas flexibles sobre los materiales en varias bibliotecas a la vez combinando varios campos de búsqueda.
 - Préstamos entre bibliotecas: poder solicitar materiales a otras bibliotecas y procesar las solicitudes de otras bibliotecas. El procesamiento de dichas solicitudes se lleva a cabo usando archivos de solicitud de la siguiente manera:
 - Preparar y exportar una lista de solicitudes de materiales que se quiere hacer a una biblioteca. Se prepara la lista usando un formulario para identificar el nombre de la biblioteca, el nombre del libro, el autor y el nombre de esta biblioteca. Una vez terminado, se guardar la lista en un archivo de texto. No es necesario en esta práctica preocuparse de cómo se enviaría el archivo a otras bibliotecas.
 - Importar y procesar un archivo de solicitudes para materiales proveniente de otra biblioteca. Se debe actualizar el estatus de cada material para marcarse como prestado, pero en vez del identificador del usuario debería aparecer el identificador de la biblioteca.
 - Control de reservas: poder reservar un material si está ya prestado, gestión de avisos (al usuario con el material que convendría devolverlo porque hay alguien esperando y al usuario con la reserva cuando el material ya está devuelto).
- a) **[1,0 puntos]** Diseñar utilizando un paradigma orientado a objetos, los elementos necesarios para la aplicación explicada de la práctica durante el curso. Es necesario identificar la estructura y las relaciones de herencia (mediante el uso de un diagrama de clases) y de uso de las clases necesarias para almacenar y gestionar esta información. Debe hacerse uso de los mecanismos de herencia siempre que sea posible. Se valorará un buen diseño que favorezca la reutilización de código y facilite su mantenimiento.
- b) **[1,5 puntos]** Implementa un método (o métodos) que permitan la importación (cargar al programa) de los diferentes usuarios que hay en el sistema de gestión de biblioteca. Justifíquese las opciones y decisiones que se tomen.
- c) **[3,0 puntos]** Implementa un método (o métodos) que implementen el proceso devolución de un préstamo. Deberá tenerse en cuenta las implicaciones que un préstamo puede acarrear: sanciones (si se entrega fuera de plazo), actualización de ficheros que contienen los préstamos (indicar la solución que se plantea en este caso), actualización de las reservas que haya sobre ese libro, etc. Justifíquese las opciones y decisiones que se tomen.
- d) **[1,0 puntos]** Proporcione un método (o métodos) que permita mostrar por pantalla un formulario básico en **modo gráfico** que permita generar las estadísticas de los préstamos que se encuentran almacenados en un fichero. La pantalla permitirá elegir entre dos listados: libros más prestados, usuarios más activos (con mayor número de préstamos). Pedirá un rango de fechas y aplicará ese criterio a la hora de buscar los contenidos en los ficheros. Los mostrará por pantalla de mayor a menor. Se pide expresamente la parte gráfica. El objetivo es ver el conocimiento y destreza en el uso de las librerías Swing y/o AWT. No desarrolle código asociado a la funcionalidad del préstamo. Justifíquese las opciones y decisiones que se tomen.

PARTE TEÓRICA - TEST [2,5 PUNTOS]:

El test consta de 14 preguntas y 2 preguntas adicionales de reserva. Solo una de las respuestas es válida. Las respuestas correctas se puntuarán con +1.0, mientras que las respondidas de manera incorrecta se puntuarán con -0.25. Las no contestadas no tendrán influencia ni positiva ni negativa en la nota.

Las preguntas de reserva sólo tendrán utilidad en el caso de que alguna de las 14 preguntas iniciales del test sea anulada por cualquier circunstancia. Caso de ocurrir este hecho, si se produjera la anulación de alguna de las 14 preguntas iniciales, la primera pregunta de reserva sustituiría a la pregunta anulada. Caso de que una segunda pregunta de las 14 iniciales fuese anulada, entonces la segunda pregunta de reserva sustituiría a esta segunda pregunta anulada. En aquellos hipotéticos casos en los que se produjese la anulación de una tercera o sucesivas preguntas de las 14 iniciales, entonces sólo en ese caso, las preguntas tercera y sucesivas anuladas se considerarían como correctas (al no existir más preguntas de reserva que las sustituyan).

Pregunta 1: Indica cual de las siguientes afirmaciones es correcta:

- a. La firma está formada por los parámetros de un método y proporciona la información necesaria para invocarlo.
- b. La firma es el encabezado de un método y puede tener parámetros para proporcionar información adicional para realizar una tarea.
- c. La firma es el encabezado de un método y proporciona la información necesaria para invocarlo.
- d. La firma es el nombre de un método y puede tener parámetros para proporcionar información adicional para realizar una tarea.

Pregunta 2: Indique el orden seguido en los ejemplos del texto de la bibliografía básica de la asignatura en cuanto a la parte interna de una clase:

```
public class NombreClase
{
    PARTE INTERNA DE UNA CLASE
}
```

- a. Constructores, Métodos y Campos
- b. Métodos, Constructores y Campos
- c. Campos, Constructores y Métodos
- d. Campos, Métodos y Constructores

Pregunta 3: Dado el siguiente fragmento de código:

```
int A = 9;
float B = 3.3F;
char C = 'w';

System.out.println(A + B > 12);
System.out.println(A >= 8 && C != 'w');
System.out.println((C == 'c') || ((A + B) == 12));
```

Indica cual será la salida por pantalla (cada valor en una línea diferente):

- a. True true false
- b. True false false
- c. True false true
- d. False false false

Pregunta 4: Indica cual de las siguientes afirmaciones es correcta en relación a que tipo de bucle se debe utilizar:

- a. Si tenemos un bucle que no está relacionado con colecciones habrá que elegir el bucle for-each
- b. El bucle for es preferible si, al principio del bucle, no sabemos cuantas veces tenemos que ejecutarlo.
- c. Si necesitamos iterar a través de todos los elementos de una colección, el bucle for-each es casi siempre la opción más elegante.
- d. El bucle for-each es adecuado cuando nos hace falta utilizar de manera explícita el contador del bucle.

Pregunta 5: Indica cual de las siguientes afirmaciones es correcta:

- a. Un objeto de tipo String puede ser modificado una vez que está creado, por tanto no es un ejemplo de objeto inmutable
- b. Un objeto es inmutable si su contenido o su estado no puede ser cambiado una vez que se ha creado
- c. La clase String tiene un método de nombre trim que permite modificar caracteres en cualquier posición de una cadena
- d. Como regla general, las cadenas de texto de tipo String se suelen comparar mediante el operador ==

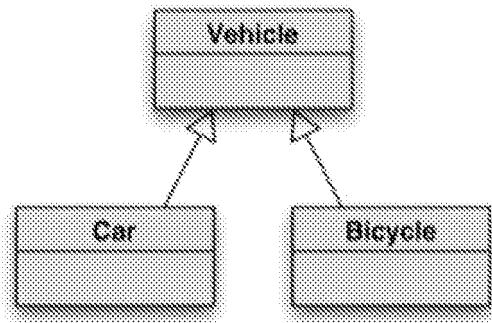
Pregunta 6: Indica cual de las siguientes afirmaciones es correcta:

- a. Una clase cohesionada representa una entidad bien acoplada.
- b. Un método cohesionado será responsable de varias tareas bien definidas.
- c. La segunda mayor ventaja de la cohesión es que ofrece un mayor potencial de utilización.
- d. Las dos más importantes maneras en que una alta cohesión beneficia a un diseño son el acoplamiento y la encapsulación.

Pregunta 7: Indica cual de las siguientes afirmaciones es correcta:

- a. Un seguimiento es la actividad de trabajar a través de un segmento de código línea por línea, mientras se observan cambios de estado y otros comportamientos de la aplicación.
- b. Un banco de pruebas es un conjunto de objetos en un estado indefinido que sirven como base para realizar pruebas de unidades.
- c. La aserción es la actividad de descubrir si una pieza de código produce el comportamiento pretendido.
- d. Una prueba es una expresión que establece una condición que esperamos que resulte verdadera.

Pregunta 8: Dada la siguiente jerarquía de herencia:



Indica cual de las siguientes asignaciones es correcta:

- a. Car c1 = new Vehicle();
- b. Car c2 = new Vehicle();
- c. Vehicle v1 = new Car();
- d. Todas las asignaciones anteriores son correctas.

Pregunta 9: Dado el siguiente código:

```
public class testJunio {  
    public void setVar (int a, int b, float c) {  
    }  
}
```

Y los siguientes métodos:

1. private void setVar (int a, float c, int b) { }
2. protected void setVar (int a, int b, float c) { }
3. public int setVar (int a, float c, int b) {return a;}
4. public int setVar (int a, int b, float c) {return a;}
5. protected float setVar (int a, int b, float c) {return c;}

Indique qué métodos permiten una sobrecarga del método setVar de manera correcta:

- a. 3 y 5
- b. 3 y 4
- c. 1 y 2
- d. 1 y 3

Pregunta 10: Indique cual de las siguientes opciones declarará un método en una clase que fuerza a una subclase a implementarlo:

- a. static void methoda (double d1) {}
- b. public native double methoda();
- c. abstract public void methoda();
- d. protected void methoda (double d1){}

Pregunta 11: Indica cual de las siguientes afirmaciones es correcta en relación a un marco de Swing (JFrame):

- a. Está compuesto de tres partes: la barra de título, una barra de menú opcional y los cuadros de diálogo.
- b. Está compuesto de tres partes: la barra de título, el panel de contenido y el gestor de los bordes.
- c. Está compuesto de tres partes: la barra de título, una barra de menú opcional y el panel de contenido.
- d. Está compuesto de cuatro partes: la barra de título, una barra de menú opcional, el panel de contenido y el gestor de los bordes.

Pregunta 12: Indique cual de las siguientes afirmaciones es correcta:

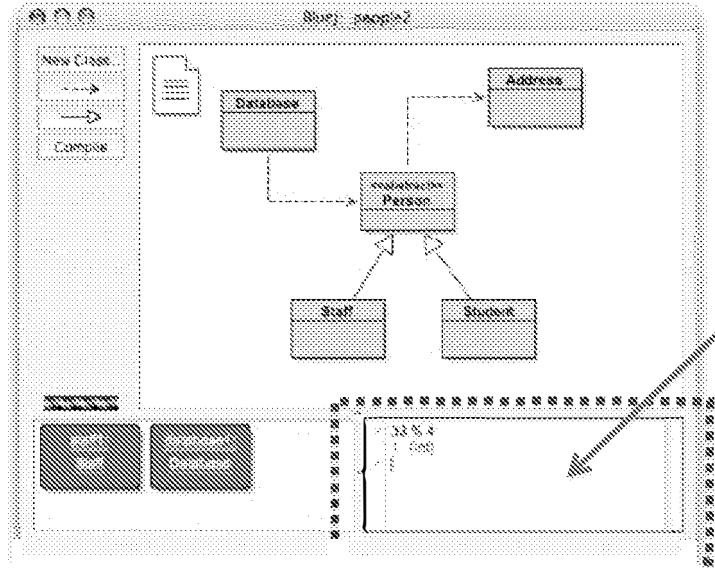
Todas las excepciones no comprobadas son subclases de

- a. Exception
- b. RunTimeException
- c. Throwable
- d. Error

Pregunta 13: Teniendo en cuenta el modelo en cascada presente en la construcción del software, indica cual de las siguientes fases NO pertenece al desarrollo de software:

- a. Análisis del problema.
- b. Prueba Unitaria.
- c. Prueba Secuencial.
- d. Entrega del sistema al cliente.

Pregunta 14: La siguiente imagen corresponde a un pantallazo de la aplicación BlueJ. En la parte inferior derecha hay un componente rectangular de BlueJ que en su interior contiene la expresión $3 \% 4$ que a su vez está rodeada por un rectángulo intermitente. ¿Qué componente o herramienta es esta?



- a. Code Add
- b. Code Exp
- c. Code Pad
- d. Code Area

Pregunta R1: Indica cual de las siguientes afirmaciones es correcta:

- a. Un mapa es una colección que almacena pares llave/valor como entradas.
- b. Un mapa es una colección que almacena tríos llave/índice/valor como entradas.
- c. Un mapa es una colección que almacena pares índice/valor como entradas.
- d. Un mapa es una colección que almacena tríos índice/posición/valor como entradas.

Pregunta R2: Indique el resultado de ejecutar el siguiente código que se muestra a continuación:

```
public class test {  
    public static void add3 (Integer i) {  
        int val = i.intValue();  
        val += 3;  
        i = new Integer (val);  
    }  
  
    public static void main (String args[]) {  
        Integer i = new Integer (0);  
        add3 (i);  
        System.out.println (i.intValue ( ) );  
    }  
}
```

- a. El programa indicará un fallo en tiempo de compilación
- b. El programa imprime por pantalla el valor 0
- c. El programa imprime por pantalla el valor 3
- d. El programa lanzará una excepción en la línea 3 (int val = i.intValue();)

PARTE PRÁCTICA [6,5 PUNTOS]:

La Práctica del presente curso es diseño e implementación de un sistema integrado de gestión de una biblioteca (a partir de ahora, SIGB). En general, las funciones que tienen un SIGB son varias según el perfil de su usuario (que va desde el usuario de la biblioteca hasta su director) e incluyen las siguientes:

- Adquisiciones: la compra de materiales (libros en diferentes formatos, audiolibros, CDs de música, películas en DVD, etc.), gestión de compras, facturación, etc.
- Catalogar: la clasificación e indexación de los materiales de la biblioteca.
- Préstamos: prestar los materiales a los usuarios (tanto en papel como en otros formatos), reservas de materiales ya en préstamos, control de préstamos (emisión de avisos de materiales fuera de plazo), gestión de multas.
- Suscripciones: gestión de las suscripciones a revistas y periódicos.
- Catálogo en línea u OPAC (del inglés Online Public Access Catalog): interfaz pública a los servicios de la biblioteca (búsquedas, gestión de prestamos, etc.).
- Gestión de usuarios: altas, bajas, generación de tarjetas, historiales.

Funcionalidades

Los SIGB permiten la implementación desde labores simples de gestión de una alta de usuario, hasta operaciones más complejas como es la gestión de préstamos o inventario. En esta práctica, se propondrán diferentes funcionalidades para el sistema de gestión bibliotecaria:

- Añadir nuevos materiales a la colección de la biblioteca (rellenando los datos de un formulario). Cada tipo de material debería tener su propia colección (libros, revistas, periódicos, audio, video, etc.).
- Borrar materiales de la colección.
- Realizar búsquedas sencillas sobre los materiales.
- Gestionar suscripciones a revistas y periódicos.
- Gestión de usuarios: altas, bajas, generación de tarjetas, historiales de préstamo, control de acceso (diferenciar entre dos perfiles: usuarios y bibliotecarios).
- Realización básica de Préstamos: prestar un material si está disponible en la biblioteca, asignar fechas de devolución.
- Producir listados de préstamos según el tipo de material.
- Realizar búsquedas flexibles sobre los materiales en la biblioteca combinando varios campos de búsqueda.
- Control de préstamos: número máximo de ítems de préstamo (6 por usuario, independiente de tipo de material), emisión de avisos de materiales fuera de plazo, gestión de multas, etc.
- Producir listados de los materiales prestados.
- Realizar búsquedas flexibles sobre los materiales en varias bibliotecas a la vez combinando varios campos de búsqueda.
- Préstamos entre bibliotecas: poder solicitar materiales a otras bibliotecas y procesar las solicitudes de otras bibliotecas. El procesamiento de dichas solicitudes se lleva a cabo usando archivos de solicitud de la siguiente manera:
 - Preparar y exportar una lista de solicitudes de materiales que se quiere hacer a una biblioteca. Se prepara la lista usando un formulario para identificar el nombre de la biblioteca, el nombre del libro, el autor y el nombre de esta biblioteca. Una vez terminado, se guardar la lista en un archivo de texto. No es necesario en esta práctica preocuparse de cómo se enviaría el archivo a otras bibliotecas.
 - Importar y procesar un archivo de solicitudes para materiales proveniente de otra biblioteca. Se debe actualizar el estatus de cada material para marcarse como prestado, pero en vez del identificador del usuario debería aparecer el identificador de la biblioteca.
- Control de reservas: poder reservar un material si está ya prestado, gestión de avisos (al usuario con el material que convendría devolverlo porque hay alguien esperando y al usuario con la reserva cuando el material ya está devuelto).

- a) **[1,0 puntos]** Diseñar utilizando un paradigma orientado a objetos, los elementos necesarios para la aplicación explicada de la práctica durante el curso. Es necesario identificar la estructura y las relaciones de herencia (mediante el uso de un diagrama de clases) y de uso de las clases necesarias para almacenar y gestionar esta información. Debe hacerse uso de los mecanismos de herencia siempre que sea posible. Se valorará un buen diseño que favorezca la reutilización de código y facilite su mantenimiento.
- b) **[2,0 puntos]** Implementa un método (o métodos) que permitan añadir nuevos materiales a la colección de la biblioteca (rellenando los datos de un formulario). Cada tipo de material debería tener su propia colección (libros, revistas, periódicos, audio, video, etc.).
- c) **[2,0 puntos]** Implementa un método (o métodos) que permitan la importación y procesamiento de un archivo de solicitudes para materiales proveniente de otra biblioteca (préstamos entre bibliotecas). Se debe actualizar el estatus de cada material para marcarse como prestado, pero en vez del identificador del usuario debería aparecer el identificador de la biblioteca.
- d) **[1,5 puntos]** Para la siguiente versión del software se desea añadir la figura de la Editorial. De cada editorial se debe tener un listado de los libros y audiolibros que proporciona, así como su precio, que podría actualizarse manualmente después de la última venta. El sistema debería consultar el inventario a final de año, y de los libros más prestados (top 10%), realizar un pedido a la Editorial correspondiente de un 20% más de libros. El porcentaje de libros más prestados y el porcentaje de libros pedidos a las Editoriales, podrán ser configurables dependiendo del presupuesto de la biblioteca. ¿Qué cambios serían necesarios en el diseño para adaptar esta nueva funcionalidad? Implemente el método (o métodos) que permita esta nueva funcionalidad.

**UNIVERSIDAD NACIONAL DE EDUCACIÓN A DISTANCIA – ESCUELA TÉCNICA SUPERIOR DE
INGENIERÍA INFORMÁTICA**
**71901072 – PROGRAMACIÓN ORIENTADA A OBJETOS (GRADO EN INGENIERÍA INFORMÁTICA /
TECNOLOGÍAS DE LA INFORMACIÓN)**
SEPTIEMBRE 2016 – MODELO D – NO ESTÁ PERMITIDO EL USO DE MATERIAL ADICIONAL

PARTE TEÓRICA - TEST [2,5 PUNTOS]:

El test consta de 14 preguntas y 2 preguntas adicionales de reserva. Solo una de las respuestas es válida. Las respuestas correctas se puntuarán con +1.0, mientras que las respondidas de manera incorrecta se puntuarán con -0.25. Las no contestadas no tendrán influencia ni positiva ni negativa en la nota.

Las preguntas de reserva sólo tendrán utilidad en el caso de que alguna de las 14 preguntas iniciales del test sea anulada por cualquier circunstancia. Caso de ocurrir este hecho, si se produjera la anulación de alguna de las 14 preguntas iniciales, la primera pregunta de reserva sustituiría a la pregunta anulada. Caso de que una segunda pregunta de las 14 iniciales fuese anulada, entonces la segunda pregunta de reserva sustituiría a esta segunda pregunta anulada. En aquellos hipotéticos casos en los que se produjese la anulación de una tercera o sucesivas preguntas de las 14 iniciales, entonces sólo en ese caso, las preguntas tercera y sucesivas anuladas se considerarían como correctas (al no existir más preguntas de reserva que las sustituyan).

Pregunta 1: ¿Cuál es el resultado de ejecutar el siguiente código?

```
1. public abstract class ClaseA {  
2.     public final void metodo1() {  
3.         System.out.println("ClaseA");  
4.     }  
5.  
6.     public static void main(String [] args) {  
7.         ClaseA obj = new ClaseB();  
8.         obj.metodo1();  
9.     }  
10. }  
11.  
12. class ClaseB extends ClaseA {  
13.     public void metodo1() {  
14.         System.out.println("ClaseB");  
15.     }  
16. }
```

- a. ClaseA
- b. ClaseB
- c. Error de compilación en la línea 7
- d. Error de compilación en la línea 13

Pregunta 2: De acuerdo a la bibliografía básica, el que un campo o miembro público de una clase sea estático implica que:

- a. Puedo acceder y modificar su valor sólo a través de un objeto.
- b. Puedo acceder y modificar su valor sin necesidad de haber instanciado objeto alguno.
- c. Todos los objetos tienen una copia de la variable.
- d. Es una variable global y se puede usar directamente en cualquier lugar sin hacer referencia a la clase correspondiente y sin instanciar objeto alguno.

Pregunta 3: Según el texto de la bibliografía básica de la asignatura, indique cuál de las siguientes afirmaciones es incorrecta:

- a. Las clases pueden ser abstractas.
- b. Las diagramas de clases muestran las clases de una aplicación y la relación entre ellas.
- c. Las clases deben contener al menos un miembro de clase o campo.
- d. Las clases pueden contener métodos.

Pregunta 4: Dada la siguiente definición de clases:

```
1. public class ClaseA {  
2.     protected void metodo1() {  
3.         System.out.print("Entro en el método desde ClaseA");  
4.     }  
5. }  
6.  
7. class ClaseB extends ClaseA {  
8.     public void metodo1() {  
9.         System.out.print("Entro en el método desde ClaseB");  
10.    }  
11. }
```

¿Qué salida obtendremos al ejecutar el siguiente código dentro de un método *main* implementado en cualquiera de las clases?

```
15. ClaseA p = new ClaseB();  
16. p.metodo1();
```

- a. Entro en el método desde Clase A
- b. Entro en el método desde Clase B
- c. Error de compilación en la línea 15
- d. Error de compilación en la línea 8

Pregunta 5: Dado el siguiente código indicar cuál de las afirmaciones es correcta. Suponga que las importaciones de librerías correspondientes se han realizado.

```
6.     Set < Integer > conjunto = new HashSet < Integer > ();  
7.     conjunto.add(new Integer(86));  
8.     conjunto.add(75);  
9.     conjunto.add(new Integer(86));  
10.    conjunto.add(null);  
11.    conjunto.add(309);  
12.    Iterator i = conjunto.iterator();  
13.    while(i.hasNext()) {  
14.        System.out.print(i.next());  
15.    }
```

- a. El código compila sin errores
- b. La salida que se muestra es 8675null309
- c. La línea 6 produce un error de compilación
- d. La línea 12 produce un error de compilación

Pregunta 6: ¿Cuál es el resultado de ejecutar el siguiente código?

```
6.     List lista = new ArrayList();
7.     lista.add("AA");
8.     lista.add("BB");
9.     lista.add(8);
10.    for(int n=0;n<lista.size();n++) {
11.        System.out.print(lista.get(n));
12.    }
```

a. AABB
b. AABB8
c. Error de compilación en la línea 6
d. Error de compilación en la línea 9

Pregunta 7: Según el texto de la bibliografía básica de la asignatura, indique cuál de las siguientes afirmaciones es correcta:

- a. Una clase abstracta no puede implementar ninguna interface.
- b. Si una clase tiene sólo alguno de sus métodos abstractos, no es necesario declararla como abstracta.
- c. Una clase abstracta no puede extender otra clase que no sea abstracta.
- d. Una clase que hereda de una clase abstracta tiene que implementar todos los métodos abstractos para no ser abstracta.

Pregunta 8: Según el texto de la bibliografía básica de la asignatura, el alcance de una variable:

- a. Define el valor máximo que puede almacenar dicha variable.
- b. Define qué métodos de otras clases pueden acceder a la variable.
- c. Define qué clases externas pueden utilizar dicha variable.
- d. Ninguna de las anteriores.

Pregunta 9: ¿Cuál es el resultado de ejecutar el siguiente código?

```
10.    String [] array= {"AA", "BB", "CC"};
11.    for(int n = 0; n < array.length; n++) {
12.        System.out.print(array[n]);
13.    }
14.    System.out.print(n);
```

a. AABBCC
b. AAB
c. AABB
d. El código no compila

Pregunta 10: ¿Cuál es el resultado de ejecutar el siguiente código?

```
1. public class ClaseA{
2.     private String miembrol;
3.     private boolean miembro2;
4.
5.     public static void main(String [] args ) {
6.         ClaseA obj = new ClaseA();
7.         if(!obj.miembrol) {
8.             System.out.println("Miembrol = " + obj.miembrol);
9.         }
10.    }
11.}
```

- a. Error de compilación en la línea 6
- b. Error de compilación en la línea 7
- c. Error de compilación en la línea 8
- d. Miembro1 = null

Pregunta 11: ¿Cuál es el resultado de ejecutar el siguiente código?

```

6. String cadena1 = "Examen";
7. String cadena2 = new String(cadena1);
8. if(cadena1 == cadena2) {
9.     System.out.println("cadena1 == cadena2");
10. }
11. if(cadena1.equals(cadena2)) {
12.     System.out.println("cadena1.equals(cadena2)");
13. }
```

- a. No se muestra nada
- b. cadena1 == cadena2
- c. cadena1.equals(cadena2)
- d. B y C son ciertas

Pregunta 12: Según el texto de la bibliografía básica de la asignatura, indique cuál de las siguientes afirmaciones es correcta:

- a. El acoplamiento describe el encapsulamiento de las clases.
- b. El encapsulamiento apropiado en las clases reduce su cohesión.
- c. El encapsulamiento apropiado en las clases reduce su acoplamiento.
- d. La cohesión de una unidad de código refleja su acoplamiento.

Pregunta 13: ¿Cuál es el resultado de ejecutar el siguiente código?

```

1. class ClaseA {
2.     public float metodo1(double d) {
3.         System.out.println("ClaseA");
4.         return 1.0F;
5.     }
6. }
7.
8. public class ClaseB extends ClaseA {
9.     public double metodo1(double d) {
10.         System.out.println("ClaseB");
11.         return 1.0;
12.     }
13.
14. public static void main(String [] args) {
15.     new ClaseB().metodo1(0.0);
16. }
17. }
```

- a. ClaseA
- b. ClaseB
- c. 1.0
- d. El código no compila

Pregunta 14: De acuerdo a la bibliografía básica, ¿qué significa instanciar una clase?

- a. Duplicar una clase.
- b. Heredar de una clase.
- c. Crear un objeto a partir de una clase.
- d. Conectar dos clases entre sí.

RESERVA 1: Respecto a las excepciones en Java, ¿Cuál de las siguientes afirmaciones es correcta?

- a. Todas las subclases de la clase estándar de Java RunTimeException son excepciones comprobadas.
- b. Todas las subclases de la clase estándar de Java Exception son excepciones comprobadas.
- c. Error es una subclase directa de Throwable, mientras que Exception es una subclase directa de Error.
- d. Tanto Error como Exception son subclases directas de Throwable.

RESERVA 2: ¿Cuál de las siguientes afirmaciones es correcta?

- a. En Java no se permite la herencia múltiple de clases, ni tampoco la implementación múltiple de interfaces.
- b. En Java se permite la herencia múltiple de clases, pero no la implementación múltiple de interfaces.
- c. En Java no se permite la herencia múltiple de clases, pero sí la implementación múltiple de interfaces.
- d. En Java se permite la herencia múltiple de clases, y también la implementación múltiple de interfaces.

PARTE PRÁCTICA [6,5 PUNTOS]:

La Práctica del presente curso es diseño e implementación de un sistema integrado de gestión de una biblioteca (a partir de ahora, SIGB). En general, las funciones que tienen un SIGB son varias según el perfil de su usuario (que va desde el usuario de la biblioteca hasta su director) e incluyen las siguientes:

- Adquisiciones: la compra de materiales (libros en diferentes formatos, audiolibros, CDs de música, películas en DVD, etc.), gestión de compras, facturación, etc.
- Catalogar: la clasificación e indexación de los materiales de la biblioteca.
- Préstamos: prestar los materiales a los usuarios (tanto en papel como en otros formatos), reservas de materiales ya en préstamos, control de préstamos (emisión de avisos de materiales fuera de plazo), gestión de multas.
- Suscripciones: gestión de las suscripciones a revistas y periódicos.
- Catálogo en línea u OPAC (del inglés Online Public Access Catalog): interfaz pública a los servicios de la biblioteca (búsquedas, gestión de préstamos, etc.).
- Gestión de usuarios: altas, bajas, generación de tarjetas, historiales.

Funcionalidades

Los SIGB permiten la implementación desde labores simples de gestión de una alta de usuario, hasta operaciones más complejas como es la gestión de préstamos o inventario. En esta práctica, se propondrán diferentes funcionalidades para el sistema de gestión bibliotecaria:

- Añadir nuevos materiales a la colección de la biblioteca (rellenando los datos de un formulario). Cada tipo de material debería tener su propia colección (libros, revistas, periódicos, audio, video, etc.).
- Borrar materiales de la colección.
- Realizar búsquedas sencillas sobre los materiales.
- Gestionar suscripciones a revistas y periódicos.
- Gestión de usuarios: altas, bajas, generación de tarjetas, historiales de préstamo, control de acceso (diferenciar entre dos perfiles: usuarios y bibliotecarios).

- Realización básica de Préstamos: prestar un material si está disponible en la biblioteca, asignar fechas de devolución.
 - Producir listados de préstamos según el tipo de material.
 - Realizar búsquedas flexibles sobre los materiales en la biblioteca combinando varios campos de búsqueda.
 - Control de préstamos: número máximo de ítems de préstamo (6 por usuario, independiente de tipo de material), emisión de avisos de materiales fuera de plazo, gestión de multas, etc.
 - Producir listados de los materiales prestados.
 - Realizar búsquedas flexibles sobre los materiales en varias bibliotecas a la vez combinando varios campos de búsqueda.
 - Préstamos entre bibliotecas: poder solicitar materiales a otras bibliotecas y procesar las solicitudes de otras bibliotecas. El procesamiento de dichas solicitudes se lleva a cabo usando archivos de solicitud de la siguiente manera:
 - Preparar y exportar una lista de solicitudes de materiales que se quiere hacer a una biblioteca. Se prepara la lista usando un formulario para identificar el nombre de la biblioteca, el nombre del libro, el autor y el nombre de esta biblioteca. Una vez terminado, se guardar la lista en un archivo de texto. No es necesario en esta práctica preocuparse de cómo se enviaría el archivo a otras bibliotecas.
 - Importar y procesar un archivo de solicitudes para materiales proveniente de otra biblioteca. Se debe actualizar el estatus de cada material para marcarse como prestado, pero en vez del identificador del usuario debería aparecer el identificador de la biblioteca.
 - Control de reservas: poder reservar un material si está ya prestado, gestión de avisos (al usuario con el material que convendría devolverlo porque hay alguien esperando y al usuario con la reserva cuando el material ya está devuelto).
- a) **[1 punto]** Diseñar utilizando un paradigma orientado a objetos, los elementos necesarios para la aplicación explicada de la práctica durante el curso. Es necesario identificar la estructura y las relaciones de herencia (mediante el uso de un diagrama de clases) y de uso de las clases necesarias para almacenar y gestionar esta información. Debe hacerse uso de los mecanismos de herencia siempre que sea posible. Se valorará un buen diseño que favorezca la reutilización de código y facilite su mantenimiento.
- b) **[1,5 puntos]** Se desea incluir en la biblioteca un nuevo tipo de recurso, los videojuegos para diferentes consolas (PS3, PS4, Xbox 360, Xbox o Nintendo). Indica los cambios que habría que realizar en el sistema de gestión de préstamos, así como la estructura de clases resultante.
- c) **[2 puntos]** Implementar los métodos de préstamo y devolución de videojuegos teniendo en cuenta, en el caso de las devoluciones, las implicaciones que dichos préstamos pueden acarrear: sanciones (si se entrega fuera de plazo), actualización de ficheros que contienen los préstamos (indicar la solución que se plantea en este caso), actualización de las reservas que haya sobre ese videojuego, etc. Justifíquese las opciones y decisiones que se tomen.
- d) **[2 puntos]** Proporcione un método (o métodos) que permita mostrar por pantalla un formulario básico en **modo gráfico** que permita generar las estadísticas de los préstamos de videojuegos que se encuentran almacenados en un fichero. La pantalla permitirá elegir entre dos listados: videojuegos más prestados, usuarios más “jugones” (con mayor número de préstamos). Pedirá un rango de fechas y aplicará ese criterio a la hora de buscar los contenidos en los ficheros. Los mostrará por pantalla de mayor a menor. Justifíquese las opciones y decisiones que se tomen.

PARTE TEÓRICA - TEST [2,5 PUNTOS]:

Solo una de las respuestas es válida. Las respuestas correctas se puntuarán con +1.0, mientras que las respondidas de manera incorrecta se puntuarán con -0.25. Las no contestadas no tendrán influencia ni positiva ni negativa en la nota.

Las preguntas de reserva sólo tendrán utilidad en el caso de que alguna de las 14 preguntas iniciales del test sea anulada por cualquier circunstancia. Caso de ocurrir este hecho, si se produjera la anulación de alguna de las 14 preguntas iniciales, la primera pregunta de reserva sustituiría a la pregunta anulada. Caso de que una segunda pregunta de las 14 iniciales fuese anulada, entonces la segunda pregunta de reserva sustituiría a esta segunda pregunta anulada. En aquellos hipotéticos casos en los que se produjese la anulación de una tercera o sucesivas preguntas de las 14 iniciales, entonces sólo en ese caso, las preguntas tercera y sucesivas anuladas se considerarían como correctas (al no existir más preguntas de reserva que las sustituyan).

Pregunta 1: Dado el siguiente código, ¿cuál de las afirmaciones es cierta?

```
public class A {  
    public int a;  
    private int b;  
    private int c;  
    public void metodo(int variable, int b1, int c1) {  
        this.a = variable;  
        this.b = b1;  
        this.c = c1;  
    }  
}
```

- a. La clase A está totalmente encapsulada.
- b. El miembro de clase b hace que la clase no esté totalmente encapsulada.
- c. El método metodo hace que la clase no esté totalmente encapsulada.
- d. El miembro de clase a hace que la clase no esté totalmente encapsulada.

Pregunta 2: Dado el siguiente código, indicar cuál es la respuesta correcta.

```
public class A {  
    public static void main(String[] args) {  
        String s;  
        Boolean b1 = true;  
        Boolean b2 = false;  
        if((b2 = false) | (21%3) > 2) s += "x";  
        if(b1 && (b2 = true)) s += "y";  
        if(b2 == true) s += "z";  
        System.out.println(s);  
    }  
}
```

- a. Error de compilación.
- b. "y" aparece en la salida del sistema.
- c. "yz" aparece en la salida del sistema.
- d. "x" aparece en la salida del sistema.

Pregunta 3: Según el texto de la bibliografía básica de la asignatura, ¿cuál de las siguientes afirmaciones es correcta?

- a. En Java no se permite la herencia múltiple de clases, ni tampoco la implementación múltiple de interfaces.
- b. En Java se permite la herencia múltiple de clases, pero no la implementación múltiple de interfaces.
- c. En Java se permite la herencia múltiple de clases, y también la implementación múltiple de interfaces.
- d. Ninguna de las anteriores afirmaciones es correcta.

Pregunta 4: ¿Qué sentencia deberíamos insertar en el siguiente código para que éste compile y se ejecute correctamente?

```
import java.util.*;  
class A {}  
interface B{}  
  
class C extends A implements B{}  
  
public class D {  
    ArrayList<B> metodo() {  
        // INSERTAR EL CODIGO AQUÍ  
    }  
}
```

- a. return new ArrayList();
- b. return new ArrayList<C>();
- c. return new ArrayList<A>();
- d. return new ArrayList<Object>();

Pregunta 5: Según el texto de la bibliografía básica de la asignatura, un método de modificación o mutador:

- a. Habitualmente devuelve void.
- b. Devuelve siempre información sobre el estado de un objeto.
- c. Permite modificar el estado únicamente de los campos públicos de la clase.
- d. Permite acceder al constructor de la clase que lo define.

Pregunta 6: Dado el siguiente código, ¿qué se mostrará en la salida del sistema?

```
class A {  
    static String s = "";  
    public static void main(String[] args) {  
        try {  
            throw new Exception();  
        } catch (Exception e){  
            try {  
                try {  
                    throw new Exception();  
                } catch (Exception ex) { s += "esto "; }  
                catch (Exception x) { s += "es "; }  
                finally { s += "una "; }  
            }  
            finally { s += "excepcion "; }  
            System.out.println(s);  
        }  
    }  
}
```

- a. esto es
- b. Error de compilación
- c. esto es una excepción
- d. esto una excepción

Pregunta 7: Dada la siguiente definición de clase:

```
public class A {  
    private int num;  
  
    public A (int n) {  
        num=n;  
    }  
  
    public int metodo() {  
        int resultado=10;  
        int numero=num;  
  
        if (num>=1) {  
            while(numero>=1){  
                resultado*=numero;  
                numero=numero-2;  
            }  
            return resultado;  
        }  
        else {  
            return 1;  
        }  
    }  
  
    public static void main (String [] args) {  
        A t=new A(4);  
        int v=t.metodo();  
        System.out.println(v);  
    }  
}
```

El resultado de la ejecución del método main sería:

- a. 80
- b. 30
- c. 20
- d. Bucle infinito

Pregunta 8: Según el texto de la bibliografía básica de la asignatura, ¿cuál de las siguientes afirmaciones es cierta?

- a. “A extiende a B” es correcto si y sólo si A es una clase y B es una interfaz.
- b. “A extiende a B” es correcto si y sólo si A es una interfaz y B es una clase.
- c. “A extiende a B” es correcto si A y B son ambas o bien clases o bien interfaces.
- d. “A extiende a B” es correcto para todas las combinaciones de A y B siendo clases y/o interfaces.

Pregunta 9: Dado el siguiente código:

```
public class A extends B {  
    public static void main(String[] args) {  
        int a = 2;  
        System.out.println(metodo(a, 1));  
    }  
}  
  
class B {  
    int metodo(int x, int y) { return x + y; }  
}
```

¿Cuál de las afirmaciones es cierta?

- a. 3
- b. Error en tiempo de ejecución.
- c. Error de compilación en la línea System.out.println(metodo(a, 1));
- d. Error de compilación en la definición de la clase B

Pregunta 10: Dado el siguiente código:

```
class A {  
    public String mensaje = "En Clase A";  
  
    public void mensaje() {  
        System.out.println(mensaje);  
    }  
}  
  
public class B extends A {  
    public String mensaje = "En Clase B";  
  
    public void mensaje(){  
        System.out.println(mensaje);  
    }  
  
    public static void main(String args[]) {  
        B obj1 = new A();  
        B obj2=obj1;  
        obj2.mensaje();  
    }  
}
```

Cuál sería el resultado de ejecutar el método main:

- a. En Clase A
- b. En Clase B
- c. Error de Compilación en la línea B obj1 = new A();
- d. En Clase A En Clase B

Pregunta 11: Según el texto de la bibliografía básica de la asignatura, las pruebas de regresión se definen como:

- a. La ejecución de pruebas automatizadas aleatorias sobre los distintos valores que puede recibir la clase evaluada.
- b. La aplicación sistemática del conjunto de casos de prueba base que se definieron justo al comenzar con el desarrollo de la aplicación y que no varían nunca a lo largo de éste.
- c. La ejecución de las pruebas pasadas previamente para asegurarse de que la nueva versión aún las pasa.
- d. El conjunto de pruebas negativas necesarias para demostrar que la clase evaluada falla.

Pregunta 12: Según el texto de la bibliografía básica de la asignatura, ¿cuál de las siguientes afirmaciones es incorrecta?:

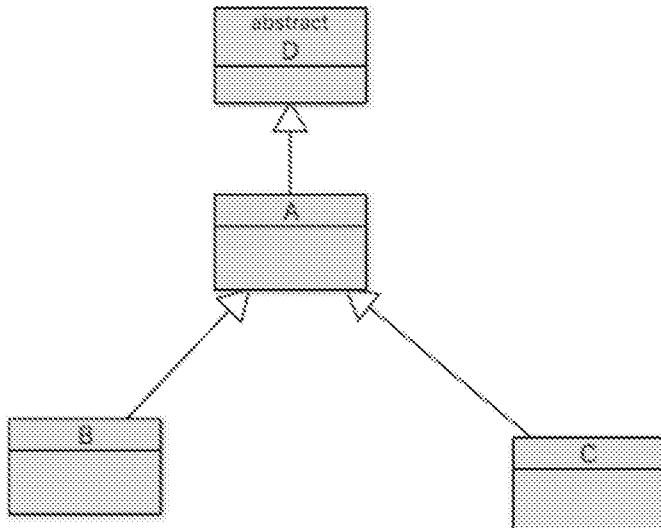
- a. Las clases pueden ser abstractas.
- b. Los diagramas de clases muestran las clases de una aplicación y la relación entre ellas.
- c. Las clases deben definir de manera explícita su constructor.
- d. Las clases pueden contener métodos.

Pregunta 13: Según el texto de la bibliografía básica de la asignatura, ¿cuál de las siguientes afirmaciones es correcta?:

- a. El lenguaje Java tiene tres variantes del bucle for: for-each, for y for-do.
- b. Un bucle while es similar en su estructura y propósito al bucle for-each.
- c. El tipo de la variable de un bucle no tiene porqué ser el mismo que el tipo del elemento declarado para la colección que estamos recorriendo con el bucle.
- d. Un índice es un objeto que proporciona funcionalidad para recorrer todos los elementos de una colección.

Pregunta 14:

Dada la siguiente jerarquía de clases:



Y la siguiente inicialización de objetos:

```

public static void main(String[] args) {
    D a=new A();
    A b=new C();
    A c=new A();
    B d=new B();
    C e=new C();
}
  
```

¿Cuáles de las siguientes asignaciones son correctas?

- 1- e=(C)b;
- 2- e=(C)c;
- 3- b=a;
- 4- b=(A)a;
- 5- d=(B)e;
- a. 1, 2, 3, 4, 5
- b. 1, 3, 5
- c. 1, 2,3
- d. 1, 2, 4

RESERVA 1: Según el texto de la bibliografía básica de la asignatura, un método de acceso o selector:

- a. Habitualmente devuelve void.
- b. Devuelve siempre información sobre el estado de un objeto.
- c. Devuelve siempre un objeto de la clase Object.
- d. Permite acceder al constructor de la clase que lo define.

RESERVA 2: Dado el siguiente código:

```

abstract class A {
    String nombre = "Clase A";

    abstract public int metodo(int valor);
}

class B extends A {
    String nombre = "Clase B";

    public String nombre(){
        return(nombre);
    }

    public int metodo(int valor){
        return valor;
    }
}

public class Test {
    public static void main(String[] args) {
        A obj = new B();
        System.out.println(obj.metodo(2));
    }
}
  
```

Cuál sería el resultado de ejecutar el método main():

- a. Clase A
- b. Error en tiempo de ejecución
- c. 2
- d. Error de compilación

PARTE PRÁCTICA [6,5 PUNTOS]:

La práctica del presente curso consiste en diseñar e implementar un sistema integrado de gestión de un taller de vehículos. Todos los dueños de algún tipo de vehículo (moto, coche, furgoneta, etc.) tienen experiencia en llevar su vehículo al taller para algún tipo de revisión y/o reparación. Puede que sea una puesta a punto antes de las vacaciones o algún viaje largo, o debido a un problema concreto o ruido que hace, o alguna fuga de líquido, etc. Hace años, el responsable de un taller no haría más que apuntar en un libro una breve descripción del vehículo, el motivo por el cual ha acudido al taller su dueño y algún número de contacto. Hoy en día, debido en parte a la competencia entre talleres y al deseo de aportar servicios de calidad para poder fidelizar al cliente, los talleres usan sistemas de gestión para todo el proceso de recepción, procesado y facturación de cada vehículo que pasa por el taller.

En general, las funciones que tienen un sistema de gestión de un taller de coches son varias:

- Recepción del vehículo: al entrar un vehículo en el taller hay que generar una ficha con los datos (si ya no forma parte del registro histórico del taller) más importantes (marca, modelo, matrícula, cliente, etc.), dejar constancia del motivo de la visita (problema mecánico, revisión, etc.) e imprimir el formulario de autorización que el dueño tiene que firmar para autorizar el trabajo.
- Asignación de los trabajos a los mecánicos: a medida que vayan terminando trabajos, el jefe del taller les va asignando nuevos vehículos.
- Procesado de los vehículos: un mecánico, al terminar la reparación de un vehículo, acude al sistema para ver los siguientes trabajos que le corresponden. Según la información del sistema, tiene que averiguar el problema, llevar a cabo el trabajo correspondiente y dejar constancia del proceso realizado en el sistema para que el comercial pueda informar al cliente de que su vehículo está listo para recoger. Una vez que el mecánico termine con un coche, en el sistema le aparece reflejado el siguiente vehículo asignado. Si por el motivo que sea (por ejemplo, falta de piezas o que un cliente no haya dado su autorización a realizar una reparación debido al coste), un mecánico deja un trabajo en un estado sin completar, anota en el sistema el motivo y pasa al siguiente trabajo.
- Gestión de usuarios: altas, bajas, modificaciones de las personas que figuran en el sistema (miembros del taller [jefe, mecánico, comercial] y clientes). La primera vez que acude un cliente al taller hay que darle de alta en el sistema.
- Gestión de clientes por parte del comercial: comunicar a los clientes el precio de una reparación, informarles de que su vehículo está listo para recoger, presentar ofertas especiales (por ejemplo, revisión antes de la inspección técnica del vehículo [ITV] o puesta a punto para las vacaciones, revisión de los neumáticos, frenos).

Se pide realizar las siguientes tareas:

- a) **[1,0 puntos]** Diseñar utilizando un paradigma orientado a objetos, los elementos necesarios para la aplicación explicada de la práctica durante el curso. Es necesario identificar la estructura y las relaciones de herencia (mediante el uso de un diagrama de clases) y de uso de las clases necesarias para almacenar y gestionar esta información. Debe hacerse uso de los mecanismos de herencia siempre que sea posible. Se valorará un buen diseño que favorezca la reutilización de código y facilite su mantenimiento.
- b) **[2,5 puntos]** Implementar un método (o métodos) que permitan visualizar un listado de los vehículos reparados en una fecha concreta ordenados de mayor a menos según el precio de la reparación. Justifíquense las opciones y decisiones que se tomen.
- c) **[2,0 puntos]** Implementar un método (o métodos) que permita asignar a los usuarios ofertas en función del coste de las reparaciones realizadas. De este modo, los usuarios que hayan gastado en sus últimas reparaciones más de 1.000 euros recibirán un descuento de 100 euros en su próxima reparación, los que hayan gastado más de 2.000 euros recibirán un descuento de 200 euros y los que hayan gastado más de 3.000 euros recibirán un descuento de 300 euros. Una vez aplicado el descuento, las reparaciones tenidas en cuenta para la oferta no deberán volver a ser tenidas en cuenta. Justifíquense las opciones y decisiones que se tomen.
- d) **[1,0 puntos]** Se quiere utilizar el sistema de gestión del taller para gestionar una pequeña tienda de accesorios para el automóvil similar a las tiendas que tienen las grandes cadenas de reparación de automóviles. Indique los cambios que serían necesarios en el diseño y la implementación para permitir esa nueva funcionalidad.

PARTE TEÓRICA - TEST [2,5 PUNTOS]:

Solo una de las respuestas es válida. Las respuestas correctas se puntuarán con +1.0, mientras que las respondidas de manera incorrecta se puntuarán con -0.25. Las no contestadas no tendrán influencia ni positiva ni negativa en la nota.

Las preguntas de reserva sólo tendrán utilidad en el caso de que alguna de las 14 preguntas iniciales del test sea anulada por cualquier circunstancia. Caso de ocurrir este hecho, si se produjera la anulación de alguna de las 14 preguntas iniciales, la primera pregunta de reserva sustituiría a la pregunta anulada. Caso de que una segunda pregunta de las 14 iniciales fuese anulada, entonces la segunda pregunta de reserva sustituiría a esta segunda pregunta anulada. En aquellos hipotéticos casos en los que se produjese la anulación de una tercera o sucesivas preguntas de las 14 iniciales, entonces sólo en ese caso, las preguntas tercera y sucesivas anuladas se considerarían como correctas (al no existir más preguntas de reserva que las sustituyan).

Pregunta 1: Según el texto de la bibliografía básica de la asignatura, ¿qué es el *estado* de un objeto?

- a. La instancia de una clase.
- b. El conjunto de valores de todos los métodos que definen a un objeto.
- c. El conjunto de valores de todos los atributos que definen a un objeto.
- d. Ninguna de las anteriores.

Pregunta 2: ¿A qué tipo pertenecen los siguientes campos?

```
private boolean listo;  
private Alumno alumno;  
private String nombre;
```

- a. boolean, Alumno, String.
- b. private, private, private.
- c. Boolean, alumno, String.
- d. listo, alumno, nombre.

Pregunta 3: Según el texto de la bibliografía básica de la asignatura, ¿qué almacenan las variables declaradas a partir de una clase?

- a. Objetos.
- b. Referencias a objetos.
- c. Copias de objetos.
- d. Ninguna de las anteriores.

Pregunta 4: Si llamamos al método `toUpperCase()` de una instancia (`nombre`) de la clase `String` de la siguiente forma:

```
nombre.toUpperCase();
```

¿Cuál sería el resultado de compilar y/o ejecutar el código?

- a. El código generaría un error de compilación.
- b. El código generaría un error de ejecución.
- c. El código convertiría la cadena `nombre` en mayúsculas.
- d. Ninguna de las anteriores.

Pregunta 5: En el siguiente fragmento de código, ¿cómo se crea la instancia del ArrayList archivos en la línea 8?

```
1 import java.util.ArrayList;  
2  
3 public class MusicOrganizer {  
4     private ArrayList<String> archivos;  
5  
6     public MusicOrganizer()  
7     {  
8  
9     }  
10 }
```

- a. archivos = new ArrayList;
- b. archivos = new ArrayList(String)();
- c. archivos = new ArrayList<String>();
- d. archivos = new ArrayList<String>;

Pregunta 6: Según el texto de la bibliografía básica de la asignatura, ¿Cómo se llama la idea de que cada clase debe ser responsable de gestionar sus propios datos?:

- a. El acoplamiento.
- b. La cohesión.
- c. El diseño dirigido por responsabilidad.
- d. Ninguna de las anteriores.

Pregunta 7: JUnit nos permite comprobar que un método devuelve un valor concreto. ¿Cómo se haría esa prueba para comprobar que el método comprobarPrecio() de una instancia (ventas) de una clase devuelva el valor de 500?

- a. assertEquals(500, ventas.comprobarPrecio());
- b. assertEquials(500, ventas.comprobarPrecio());
- c. assertMethod(500, ventas.comprobarPrecio());
- d. Ninguna de las anteriores.

Pregunta 8: Según el texto de la bibliografía básica de la asignatura, las ventajas de la herencia incluyen (indica la respuesta **incorrecta**):

- a. Facilita la duplicación de código.
- b. Facilita la reutilización de código.
- c. Facilita el mantenimiento de código.
- d. Facilita la ampliabilidad de código.

Pregunta 9: Según el texto de la bibliografía básica de la asignatura, para averiguar la clase de un objeto, se usa el operador:

- a. super
- b. instanceof
- c. instanceOf
- d. Ninguna de las anteriores.

Pregunta 10: Según el texto de la bibliografía básica de la asignatura, un mapa es una colección que almacena (indique la afirmación correcta):

- a. pares llave/valor como entradas.
- b. trios llave/índice/valor como entradas.
- c. pares índice/valor como entradas.
- d. trios índice/posición/valor como entradas.

Pregunta 11: ¿Cómo se podría añadir un menú nuevo a una barra de menús (barraMenus) en Java?

- a. JMenu archivoMenu = new JMenu("Archivo"); barraMenu.add(archivosMenu);
- b. JMenu archivoMenu = new JMenu("Archivo"); barraMenu.addMenu(archivosMenu);
- c. JMenuItem archivoMenu = new JMenuItem("Archivo"); barraMenu.add(archivosMenu);
- d. Ninguna de las anteriores

Pregunta 12: Respecto a File y Path, podemos afirmar lo siguiente:

- a. La interfaz File permite a un programa consultar los detalles relativos a un archivo externo, de una forma independiente del sistema de archivos concreto sobre el que se esté ejecutando el programa.
- b. La interfaz Path dispone por sí misma de sendos métodos exists y canRead.
- c. La interfaz Files proporciona un gran número de métodos estáticos para consultar los atributos de un objeto Path.
- d. Ninguna de las anteriores.

Pregunta 13: Dado el siguiente código:

```
1 public class PruebaExcepciones {  
2     public PruebaExcepciones(){}
3     public void lanzarExcepcion(int prueba)
4     {
5         if(prueba == 1){
6             throw new IllegalArgumentException("Excepción lanzada");
7         }
8         System.out.println("Prueba superada");
9     }
10 }
```

¿Cuál es el resultado de compilar / llamar el método lanzarExcepcion(1)?

- a. El problema se compila y se ejecuta generando una excepción: java.lang.IllegalArgumentException.
- b. El problema se compila y se ejecuta generando la salida Prueba superada.
- c. Se produce un error de compilación en la línea 1.
- d. Se produce un error de compilación en la línea 2.

Pregunta 14: Como parte del proceso de diseño de un programa orientado a objetos se pueden usar las tarjetas CRC. ¿Qué significa CR?

- a. Collaborators/Responsibilities/Collaboration.
- b. Class/Responsibilities/Class.
- c. Class/Responsibilities/Collaborators.
- d. Ninguna de las anteriores.

RESERVA 1: En BlueJ, con un proyecto abierto, ¿cómo se puede acceder al código fuente de una clase? (Indica la respuesta incorrecta)

- a. Haciendo clic con el botón izquierdo dos veces sobre una clase.
- b. Haciendo clic con el botón derecho y seleccionando "Abrir Editor".
- c. Seleccionando la clase e ir al menú Herramientas (Tools) y seleccionando la entrada "Abrir Editor".
- d. Todas las anteriores.

RESERVA 2: Sobre los tipos estático y dinámico de una variable se puede decir, según el texto de la bibliografía básica de la asignatura:

- a. El tipo estático de una variable es el tipo tal como está declarado en el código fuente. El tipo dinámico de una variable es el tipo del objeto que está almacenado actualmente.
- b. El tipo dinámico de una variable es el tipo tal como está declarado en el código fuente. El tipo estático de una variable es el tipo del objeto que está almacenado actualmente.
- c. Tanto el tipo estático como el tipo dinámico de una variable son el tipo tal como está declarado en el código fuente.
- d. Tanto el tipo estático como el tipo dinámico de una variable son el tipo del objeto que está almacenado actualmente.

PARTE PRÁCTICA [6,5 PUNTOS]:

La práctica del presente curso consiste en diseñar e implementar un sistema integrado de gestión de un taller de vehículos. Todos los dueños de algún tipo de vehículo (moto, coche, furgoneta, etc.) tienen experiencia en llevar su vehículo al taller para algún tipo de revisión y/o reparación. Puede que sea una puesta a punto antes de las vacaciones o algún viaje largo, o debido a un problema concreto o ruido que hace, o alguna fuga de líquido, etc. Hace años, el responsable de un taller no haría más que apuntar en un libro una breve descripción del vehículo, el motivo por el cual ha acudido al taller su dueño y algún número de contacto. Hoy en día, debido en parte a la competencia entre talleres y al deseo de aportar servicios de calidad para poder fidelizar al cliente, los talleres usan sistemas de gestión para todo el proceso de recepción, procesado y facturación de cada vehículo que pasa por el taller.

En general, las funciones que tienen un sistema de gestión de un taller de coches son varias:

- Recepción del vehículo: al entrar un vehículo en el taller hay que generar una ficha con los datos (si ya no forma parte del registro histórico del taller) más importantes (marca, modelo, matrícula, cliente, etc.), dejar constancia del motivo de la visita (problema mecánico, revisión, etc.) e imprimir el formulario de autorización que el dueño tiene que firmar para autorizar el trabajo.
- Asignación de los trabajos a los mecánicos: a medida que vayan terminando trabajos, el jefe del taller les va asignando nuevos vehículos.
- Procesado de los vehículos: un mecánico, al terminar la reparación de un vehículo, acude al sistema para ver los siguientes trabajos que le corresponden. Según la información del sistema, tiene que averiguar el problema, llevar a cabo el trabajo correspondiente y dejar constancia del proceso realizado en el sistema para que el comercial pueda informar al cliente de que su vehículo está listo para recoger. Una vez que el mecánico termine con un coche, en el sistema le aparece reflejado el siguiente vehículo asignado. Si por el motivo que sea (por ejemplo, falta de piezas o que un cliente no haya dado su autorización a realizar una reparación debido al coste), un mecánico deja un trabajo en un estado sin completar, anota en el sistema el motivo y pasa al siguiente trabajo.
- Gestión de usuarios: altas, bajas, modificaciones de las personas que figuran en el sistema (miembros del taller [jefe, mecánico, comercial] y clientes). La primera vez que acude un cliente al taller hay que darle de alta en el sistema.
- Gestión de clientes por parte del comercial: comunicar a los clientes el precio de una reparación, informarles de que su vehículo está listo para recoger, presentar ofertas especiales (por ejemplo, revisión antes de la inspección técnica del vehículo [ITV] o puesta a punto para las vacaciones, revisión de los neumáticos, frenos).

Se pide realizar las siguientes tareas:

- a) **[1,0 punto]** Diseñar utilizando un paradigma orientado a objetos, los elementos necesarios para la aplicación explicada de la práctica durante el curso. Es necesario identificar la estructura y las relaciones de herencia (mediante el uso de un diagrama de clases) y de uso de las clases necesarias para almacenar y gestionar esta información. Debe hacerse uso de los mecanismos de herencia siempre que sea posible. Se valorará un buen diseño que favorezca la reutilización de código y facilite su mantenimiento.
- b) **[1,5 puntos]** Implementar un método (o métodos) que permitan gestionar las diferentes ofertas y promociones que puede ofrecer el taller: puesta a punto del vehículo antes de los períodos vacacionales principales (Semana Santa, verano, Navidad), cambio de neumáticos, cambio de filtro y aceite, etc. Justifíquese las opciones y decisiones que se tomen.
- c) **[2 puntos]** Implementar un método (o métodos) que permita(n) que cada mecánico vea las fichas que le corresponde gestionar y pueda editar los datos dejando constancia del trabajo realizado y el estado de reparación (por ejemplo, pendiente, en proceso, parado [hace falta piezas, pendiente de confirmación del cliente], fase de prueba, terminado).
- d) **[2,0 puntos]** Se quiere utilizar el sistema de gestión del taller para gestionar una empresa de alquiler de vehículos. Indique los cambios que serían necesarios en el diseño y la implementación para permitir esa nueva funcionalidad.

**UNIVERSIDAD NACIONAL DE EDUCACIÓN A DISTANCIA – ESCUELA TÉCNICA SUPERIOR DE
INGENIERÍA INFORMÁTICA**
**71901072 – PROGRAMACIÓN ORIENTADA A OBJETOS (GRADO EN INGENIERÍA INFORMÁTICA /
TECNOLOGÍAS DE LA INFORMACIÓN)**
SEPTIEMBRE 2017 – MODELO C – NO ESTÁ PERMITIDO EL USO DE MATERIAL ADICIONAL

PARTE TEÓRICA - TEST [2,5 PUNTOS]:

Solo una de las respuestas es válida. Las respuestas correctas se puntuarán con +1.0, mientras que las respondidas de manera incorrecta se puntuarán con -0.25. Las no contestadas no tendrán influencia ni positiva ni negativa en la nota.

Las preguntas de reserva sólo tendrán utilidad en el caso de que alguna de las 14 preguntas iniciales del test sea anulada por cualquier circunstancia. Caso de ocurrir este hecho, si se produjera la anulación de alguna de las 14 preguntas iniciales, la primera pregunta de reserva sustituiría a la pregunta anulada. Caso de que una segunda pregunta de las 14 iniciales fuese anulada, entonces la segunda pregunta de reserva sustituiría a esta segunda pregunta anulada. En aquellos hipotéticos casos en los que se produjese la anulación de una tercera o sucesivas preguntas de las 14 iniciales, entonces sólo en ese caso, las preguntas tercera y sucesivas anuladas se considerarían como correctas (al no existir más preguntas de reserva que las sustituyan).

Pregunta 1: ¿Qué entendemos por *signatura*?

- a. La cabecera de un método.
- b. El tipo de datos que devuelve un método.
- c. El tipo de datos que devuelve un constructor.
- d. Ninguna de las anteriores.

Pregunta 2: Queremos compilar el siguiente código que se puede encontrar en el texto base de la asignatura y que ha sido modificado convenientemente. ¿Cuál es el resultado que obtenemos al compilar?

```
1  public class Prueba
2  {
3      public static void main (String args[])
4      {
5          String cadena1 = new String("ejemPLO");
6          String cadena2 = new String("EJEMPLO");
7          cadena1.toUpperCase();
8          if (cadena1.toString().equals(cadena2.toString()))
9          {
10              System.out.println("Son iguales");
11          }
12          else
13          {
14              System.out.println("Son diferentes");
15          }
16      }
17  }
```

- a. Se produce una excepción y la ejecución falla.
- b. Se imprime por pantalla el mensaje: Son iguales.
- c. Se imprime por pantalla el mensaje: Son diferentes.
- d. Ninguna de las anteriores.

Pregunta 3: Según el texto de la bibliografía básica de la asignatura, ¿cuál de las siguientes afirmaciones es correcta respecto al texto de una clase?

- a. El envoltorio exterior contiene la cabecera de la clase.
- b. El propósito principal del envoltorio exterior es proporcionar un constructor a la clase.
- c. El envoltorio exterior permite sobrecargar cualquier método de la clase.
- d. Ninguna de las anteriores.

Pregunta 4: Supongamos que reescribimos el ejemplo BouncingBall del libro de la forma en que se muestra a continuación:

```
1  public class BouncingBall {  
2      static int n = 10;  
3      static Integer num = new Integer(n);  
4  
5      public static void main (String args [])  
6      {  
7          if (n!=0)  
8          {  
9              incrementar (num);  
10             System.out.println("Valor: " + num.intValue());  
11         }  
12     }  
13  
14     public static void incrementar (Integer numero)  
15     {  
16         numero++;  
17     }  
18 }
```

¿Cuál es el resultado de compilar y/o ejecutar el código?

- a. El programa no compila. Se produce un error en la línea 16
- b. El programa no compila. Se produce un error en la línea 10
- c. El programa compila e imprime el resultado “Valor: 11”
- d. Ninguna de las anteriores.

Pregunta 5: Según el texto de la bibliografía básica de la asignatura, ¿qué se entiende por abstracción?

- a. Proceso de dividir un todo en partes bien definidas que puedan construirse y examinarse por separado y que interactúen de formas bien definidas.
- b. Capacidad de ignorar los detalles de las distintas partes, para centrar la atención en un nivel superior de un problema.
- c. Capacidad de dividir un todo en partes bien definidas ignorando los detalles de las distintas partes.
- d. Ninguna de las anteriores.

Pregunta 6: Según el texto de la bibliografía básica de la asignatura, con respecto al constructor de la subclase podemos afirmar que ...

- a. Debe siempre invocar al constructor de su superclase como primera instrucción. Si el código fuente no incluye esa llamada, Java intentará insertar una llamada automáticamente.
- b. No debe invocar nunca al constructor de su superclase como primera instrucción. Si la incluye esa llamada, Java ignorará esta llamada automáticamente.
- c. Debe siempre invocar al constructor de su superclase como última instrucción. Si no incluye esa llamada, Java intentará insertar una llamada automáticamente.
- d. Debe siempre invocar al constructor de su superclase como última instrucción. Si no incluye esa llamada, Java generará un error de compilación.

Pregunta 7: ¿Cómo se llama el entorno de pruebas que soporta la prueba estructurada de unidades y las pruebas de regresión en Java?

- a. JDK.
- b. JBoss.
- c. Javadoc.
- d. JUnit.

Pregunta 8: Sea el siguiente código modificado de la clase MusicOrganizer mostrada en el libro base:

```
1 import java.util.*;
2
3 public class MusicOrganizer {
4
5     public static void main (String args []) {
6
7         ArrayList <Double> a = new ArrayList <Double> (5);
8         Integer numero = new Integer(0);
9
10        for (int i=0; i<=5; i++)
11        {
12            numero = Integer(i);
13            a.add(numero);
14        }
15        System.out.println(a.toString());
16    }
17 }
```

¿Qué ocurre cuando se intenta compilar y ejecutar el código?

- a. No compila. Hay que sustituir la línea 12 por la siguiente: numero = i; Haciendo esto, el programa compila y proporciona el resultado [0.0, 1.0, 2.0, 3.0, 4.0, 5.0]
- b. No compila. Hay que sustituir la línea 13 por la siguiente: a.add(new Double(numero.intValue())); Haciendo esto, el programa compila y proporciona el resultado [0.0, 1.0, 2.0, 3.0, 4.0, 5.0]
- c. No compila. Hay que sustituir la línea 12 por la siguiente: numero = i; y la línea 13 por la siguiente: a.add(new Double(numero.intValue())); Haciendo esto, el programa compila y proporciona el resultado [0.0, 1.0, 2.0, 3.0, 4.0, 5.0]
- d. Compila y proporciona el resultado [0.0, 1.0, 2.0, 3.0, 4.0, 5.0].

Pregunta 9: Según el texto de la bibliografía básica de la asignatura, el acceso protegido ...

- a. No puede aplicarse a métodos y constructores de una clase, sólo a los campos de una clase.
- b. No puede aplicarse a los campos de una clase, sólo a métodos y constructores de una clase.
- c. Puede aplicarse a cualquier miembro de una clase, aunque suele reservarse a campos y métodos, no constructores.
- d. Puede aplicarse a cualquier miembro de una clase, aunque suele reservarse a constructores y métodos, no campos.

Pregunta 10: Respecto a File y Path, podemos afirmar ...

- a. La interfaz File permite a un programa consultar los detalles relativos a un archivo externo, de una forma independiente del sistema de archivos concreto sobre el que se esté ejecutando el programa.
- b. La interfaz Path dispone por sí misma de sendos métodos exists y canRead.
- c. La interfaz Files proporciona un gran número de métodos estáticos para consultar los atributos de un objeto Path.
- d. Ninguna de las anteriores.

Pregunta 11: Queremos compilar el siguiente código que se puede encontrar en el texto base de la asignatura y que ha sido convenientemente modificado. ¿Qué ocurre cuando lo compilamos?

```
1 import java.awt.*;
2 public class ImageViewer extends Frame{
3     public static void main(String argv[]){
4         ImageViewer MiImageViewer=new ImageViewer();
5     }
6
7     ImageViewer(){
8         Button BotonHola=new Button("Hola");
9         Button BotonAdios=new Button("Adios");
10        add(BotonHola);
11        add(BotonAdios);
12        setSize(300,300);
13        setVisible(true);
14    }
15 }
```

- a. Compila, y muestra dos botones juntos ocupando todo el frame. “Hola” en la izquierda y “Adios” en la derecha.
- b. Compila, y muestra un botón ocupando todo el frame diciendo “Hola”.
- c. Compila, y muestra un botón ocupando todo el frame diciendo “Adios”.
- d. No compila.

Pregunta 12: Según el texto de la bibliografía básica de la asignatura, ¿qué permite el patrón Método Factoría?

- a. Garantiza que solo se cree una instancia de una clase y proporciona un acceso unificado a la misma.
- b. Proporciona una interfaz para crear objetos, pero deja que las subclases decidan qué clase específica se crea.
- c. Trata con el problema de añadir funcionalidad a un objeto existente.
- d. Define una relación uno-a-muchos, de modo que cuando un objeto cambie su estado, muchos otros pueden ser notificados.

Pregunta 13: Se quiero compilar y ejecutar el siguiente código obtenido del manual de referencia y que ha sido oportunamente modificado. ¿Cuál es el resultado que obtenemos?

```
1 public abstract class Simulator {
2     public static void main(String argv[]){
3         System.out.println(mensajePantalla());
4     }
5
6     public static String mensajePantalla(){
7         return ("Hola");
8     }
9 }
```

- a. El problema compila y se ejecuta correctamente, mostrando por pantalla el mensaje “Hola”.
- b. Se produce un error de compilación en la línea 1.
- c. Se produce un error de compilación en la línea 2.
- d. Se produce un error de compilación en la línea 3.

Pregunta 14: ¿Qué suceso se genera cuando se hace clic en un botón o se mueve el ratón?

- a. Se genera un suceso ActionEvent en ambos casos.
- b. Se genera un suceso ActionPerform en ambos casos.
- c. Se genera un suceso ActionListener en ambos casos.
- d. Ninguna de las anteriores.

RESERVA 1: ¿Cuál de las siguientes es una característica de la clase java.lang.Error?

- a. extends Exception.
- b. implements Throwable.
- c. implements Exception.
- d. Ninguna de las anteriores.

RESERVA 2: La siguiente operación: System.out.println (4 + 3); ¿qué imprime por pantalla?

- a. 0
- b. 1
- c. 6
- d. 7

PARTE PRÁCTICA [6,5 PUNTOS]:

La práctica del presente curso consiste en diseñar e implementar un sistema integrado de gestión de un taller de vehículos. Todos los dueños de algún tipo de vehículo (moto, coche, furgoneta, etc.) tienen experiencia en llevar su vehículo al taller para algún tipo de revisión y/o reparación. Puede que sea una puesta a punto antes de las vacaciones o algún viaje largo, o debido a un problema concreto o ruido que hace, o alguna fuga de líquido, etc. Hace años, el responsable de un taller no haría más que apuntar en un libro una breve descripción del vehículo, el motivo por el cual ha acudido al taller su dueño y algún número de contacto. Hoy en día, debido en parte a la competencia entre talleres y al deseo de aportar servicios de calidad para poder fidelizar al cliente, los talleres usan sistemas de gestión para todo el proceso de recepción, procesado y facturación de cada vehículo que pasa por el taller.

En general, las funciones que tienen un sistema de gestión de un taller de coches son varias:

- Recepción del vehículo: al entrar un vehículo en el taller hay que generar una ficha con los datos (si ya no forma parte del registro histórico del taller) más importantes (marca, modelo, matrícula, cliente, etc.), dejar constancia del motivo de la visita (problema mecánico, revisión, etc.) e imprimir el formulario de autorización que el dueño tiene que firmar para autorizar el trabajo.
- Asignación de los trabajos a los mecánicos: a medida que vayan terminando trabajos, el jefe del taller les va asignando nuevos vehículos.
- Procesado de los vehículos: un mecánico, al terminar la reparación de un vehículo, acude al sistema para ver los siguientes trabajos que le corresponden. Según la información del sistema, tiene que averiguar el problema, llevar a cabo el trabajo correspondiente y dejar constancia del proceso realizado en el sistema para que el comercial pueda informar al cliente de que su vehículo está listo para recoger. Una vez que el mecánico termine con un coche, en el sistema le aparece reflejado el siguiente vehículo asignado. Si por el motivo que sea (por ejemplo, falta de piezas o que un cliente no haya dado su autorización a realizar una reparación debido al coste), un mecánico deja un trabajo en un estado sin completar, anota en el sistema el motivo y pasa al siguiente trabajo.

- Gestión de usuarios: altas, bajas, modificaciones de las personas que figuran en el sistema (miembros del taller [jefe, mecánico, comercial] y clientes). La primera vez que acude un cliente al taller hay que darle de alta en el sistema.
- Gestión de clientes por parte del comercial: comunicar a los clientes el precio de una reparación, informarles de que su vehículo está listo para recoger, presentar ofertas especiales (por ejemplo, revisión antes de la inspección técnica del vehículo [ITV] o puesta a punto para las vacaciones, revisión de los neumáticos, frenos).

Se pide realizar las siguientes tareas:

- a) **[1,0 puntos]** Diseñar utilizando un paradigma orientado a objetos, los elementos necesarios para la aplicación explicada de la práctica durante el curso. Es necesario identificar la estructura y las relaciones de herencia (mediante el uso de un diagrama de clases) y de uso de las clases necesarias para almacenar y gestionar esta información. Debe hacerse uso de los mecanismos de herencia siempre que sea posible. Se valorará un buen diseño que favorezca la reutilización de código y facilite su mantenimiento.
- b) **[2,5 puntos]** Implementar un método (o métodos) que permitan generar los listados de vehículos que se han reparado en un día determinado, indicando, para cada uno de los vehículos, los mecánicos que han intervenido, las piezas que se han utilizado y una descripción de cada uno de los trabajos realizados. El listado aparecerá ordenado ascendente por la matrícula del vehículo (en el listado aparecen antes aquellos que tengan una matrícula más antigua). Justifique las opciones y decisiones que se tomen.
- c) **[2,0 puntos]** Implementar un método (o métodos) que implementen la gestión del servicio integral de revisión técnica (ITV) de los vehículos: el cliente puede dejar su coche, el taller lleva a cabo una revisión, lleva el coche para la ITV y realiza las reparaciones adicionales necesarias. Justifique las opciones y decisiones que se tomen.
- d) **[1,0 puntos]** Proporcione un método (o métodos) que muestre por pantalla un formulario básico en **modo gráfico** que recoja los parámetros necesarios para dar de alta un nuevo vehículo en el sistema. Se pide expresamente la parte gráfica (asumiendo que la lógica de funcionamiento está desarrollada). El objetivo es ver el conocimiento y destreza en el uso de las librerías Swing y/o AWT. No desarrolle código asociado a la funcionalidad del alta. Justifique las opciones y decisiones que se tomen.

**UNIVERSIDAD NACIONAL DE EDUCACIÓN A DISTANCIA – ESCUELA TÉCNICA SUPERIOR DE
INGENIERÍA INFORMÁTICA**
**71901072 – PROGRAMACIÓN ORIENTADA A OBJETOS (GRADO EN INGENIERÍA INFORMÁTICA /
TECNOLOGÍAS DE LA INFORMACIÓN)**
SEPTIEMBRE 2017 – MODELO D – NO ESTÁ PERMITIDO EL USO DE MATERIAL ADICIONAL

PARTE TEÓRICA - TEST [2,5 PUNTOS]:

Solo una de las respuestas es válida. Las respuestas correctas se puntuarán con +1.0, mientras que las respondidas de manera incorrecta se puntuarán con -0.25. Las no contestadas no tendrán influencia ni positiva ni negativa en la nota.

Las preguntas de reserva sólo tendrán utilidad en el caso de que alguna de las 14 preguntas iniciales del test sea anulada por cualquier circunstancia. Caso de ocurrir este hecho, si se produjera la anulación de alguna de las 14 preguntas iniciales, la primera pregunta de reserva sustituiría a la pregunta anulada. Caso de que una segunda pregunta de las 14 iniciales fuese anulada, entonces la segunda pregunta de reserva sustituiría a esta segunda pregunta anulada. En aquellos hipotéticos casos en los que se produjese la anulación de una tercera o sucesivas preguntas de las 14 iniciales, entonces sólo en ese caso, las preguntas tercera y sucesivas anuladas se considerarían como correctas (al no existir más preguntas de reserva que las sustituyan).

Pregunta 1: Indique cuál de las siguientes afirmaciones es correcta:

- a. La firma está formada por los parámetros de un método y proporciona la información necesaria para invocarlo.
- b. La firma es el nombre de un método y puede tener parámetros para proporcionar información adicional para realizar una tarea.
- c. La firma es el encabezado de un método y proporciona la información necesaria para invocarlo.
- d. La firma es el encabezado de un método y puede tener parámetros para proporcionar información adicional para realizar una tarea.

Pregunta 2: Dado el siguiente fragmento de código:

```
int indice = 1;
boolean[] examen = new boolean[8];
boolean poo = examen [indice];
```

Indica cual de las siguientes afirmaciones es correcta en relación al valor de la variable poo.

- a. poo tiene el valor false
- b. poo tiene el valor 0
- c. poo tiene el valor null
- d. Se produce una excepción y poo no posee ningún valor

Pregunta 3: Dadas las siguientes expresiones:

```
1. (8 == 8) | (10 > 8) == false | true == true
2. (8 > 8) && (8 > 8) == (8 > 8) == false
```

Indica cual de las siguientes opciones es la correcta:

- a. La expresión 1 es evaluada como falsa y la expresión 2 como falsa.
- b. La expresión 1 es evaluada como falsa y la expresión 2 como verdadera.
- c. La expresión 1 es evaluada como verdadera y la expresión 2 como falsa.
- d. La expresión 1 es evaluada como verdadera y la expresión 2 como verdadera.

Pregunta 4: Indique cuál de las siguientes afirmaciones es correcta:

- a. Un objeto colección puede almacenar un número no definido de otros objetos.
- b. Un bucle se puede utilizar para ejecutar un bloque de instrucciones repetidamente, teniendo que escribirlas múltiples veces.
- c. Un iterador es un objeto que proporciona funcionalidad para iterar a través de todos los elementos de una colección.
- d. Una matriz es un tipo especial de colección que puede almacenar un número variable de elementos.

Pregunta 5: Indica cual de las siguientes afirmaciones es correcta:

- a. Un objeto de tipo String puede ser modificado una vez que está creado, por tanto no es un ejemplo de objeto inmutable
- b. La clase String tiene un método de nombre trim que permite modificar caracteres en cualquier posición de una cadena
- c. Como regla general, las cadenas de texto de tipo String se suelen comparar mediante el operador ==
- d. Un objeto es inmutable si su contenido o su estado no puede ser cambiado una vez que se ha creado

Pregunta 6: Dado el siguiente fragmento de código, indique cuál de las siguientes afirmaciones es el resultado de su ejecución:

```
class Test
{
    public static void main (String args [])
    {
        int n, c = 1, serie = 5;
        System.out.print ("Cantidad de terminos: ");
        n = 7;
        while (c < n)
        {
            System.out.print ("," + serie);
            serie += 5;
            c++;
        }
    }
}
```

- a. Cantidad de terminos: 5,10,15,20,25,30,
- b. Cantidad de terminos: ,5,10,15,20,25,30
- c. Cantidad de terminos: ,5,10,15,20,25,30,35
- d. Cantidad de terminos: ,5,10,15,20,25,30,35,40

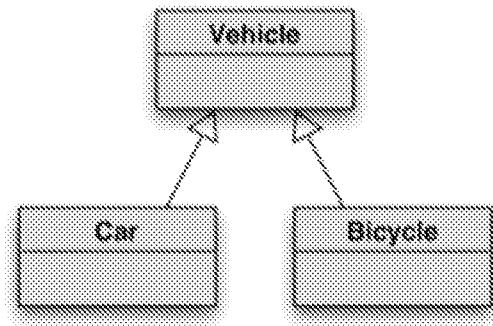
Pregunta 7: Indica cual de las siguientes afirmaciones es correcta:

- a. El término acoplamiento describe cuánto se ajusta una unidad de código a una tarea lógica o a una entidad
- b. El acoplamiento describe la conectividad de los propios objetos de una clase
- c. Un encapsulamiento apropiado en las clases reduce el acoplamiento
- d. Un sistema débilmente acoplado se caracteriza por la imposibilidad de modificar una de sus clases sin tener que realizar cambios en ninguna otra

Pregunta 8: Indica cual de las siguientes afirmaciones es correcta:

- a. La depuración es el intento únicamente de localizar el origen de un error.
- b. Una prueba positiva es la prueba de un caso que se espera que no funcione correctamente.
- c. Las pruebas son la actividad consistente en averiguar si un fragmento de código presenta el comportamiento deseado.
- d. Una aserción es una expresión que establece una condición que esperamos que sea cierta o falsa.

Pregunta 9 : Dada la siguiente jerarquía de herencia:



Indica cual de las siguientes asignaciones es correcta:

- a. Vehicle v1 = new Car();
- b. Car c1 = new Vehicle();
- c. Car c2 = new Bicycle();
- d. Todas las asignaciones anteriores son correctas.

Pregunta 10: Dado el siguiente fragmento de código que pretende mostrar un ejemplo de sobrescritura:

```
class Examen {  
    private float pregunta = 1.0f ;  
    protected float getNota () {return pregunta;}  
}  
  
class Test extends Examen {  
    private float nota = 2.0f;  
    //Insertar código aquí  
}
```

Indique cual de las siguientes opciones completaría el código anterior para dar lugar a un ejemplo correcto de sobrescritura:

- a. public float getNota (float valor) { return valor;}
- b. float getNota () { return nota;}
- c. float double getNota () { return nota;}
- d. public float getNota () { return nota;}

Pregunta 11: Indica cual de las siguientes afirmaciones es correcta:

- a. La definición de un método abstracto está compuesta de una cabecera de método, sin que exista un cuerpo del mismo.
- b. Una clase abstracta es una clase que esta pensada para crear instancias.
- c. Para que una subclase de una clase abstracta se transforme en abstracta, debe proporcionar implementaciones para todos los métodos abstractos heredados.
- d. Una interfaz Java es una especificación de un tipo (en la forma de un nombre de tipo y un conjunto de métodos) que define alguna implementación para los métodos.

Pregunta 12: La ejecución del siguiente fragmento de código

```
import javax.swing.*;  
  
class PrimerFrame extends JFrame  
{  
    public PrimerFrame()  
    {  
        setTitle("Mi primer programa gráfico");  
        setSize(400,100);  
    }  
}  
  
public class FrameTest  
{  
    public static void main(String[] args)  
    {  
        JFrame frame = new PrimerFrame();  
        frame.setVisible(true);  
    }  
}
```

Da lugar al siguiente programa



Pero este último programa tiene el problema de que cuando se cierra la ventana, a pesar de que dejamos de verla, el programa no finaliza su ejecución. De esta forma, para que el programa funcione correctamente, hemos de interceptar el evento que se produce cuando cerramos la ventana y hacer que el programa termine su ejecución en ese momento.

Indique que clase hemos de definir en este caso y asociárselo al JFrame del ejemplo:

- a. ActionListener
- b. ComponentListener
- c. ItemListener
- d. WindowListener

Pregunta 13: Indique cual de las siguientes afirmaciones es correcta en relación a la programación por parejas:

- a. Es uno de los elementos de una técnica que se conoce como programación extrema.
- b. Consiste en programar una clase por duplicado con el objetivo de depurar los errores más fácilmente.
- c. Es una manera de producir código, opuesta a la programación extrema en la que un solo programador desarrolla las clases asignadas.
- d. Era una técnica de programación tradicional que las empresas eliminaron para reducir costes.

Pregunta 14: En el siguiente fragmento de código hemos definido la ejecución de cinco bloques. Estos bloques se ejecutarán dependiendo de las excepciones que se produzcan en cada caso.

```
// Bloque1
try{
    // Bloque2
} catch (ArithmaticException e) {
    // Bloque3
} finally{
    // Bloque4
}
// Bloque5
```

Indique cual de las siguientes afirmaciones es correcta:

- El Bloque4 no se ejecutará si se produce una excepción de tipo aritmético en el Bloque2
- El Bloque4 no se ejecutará si se produce un acceso a un objeto nulo (null) en el Bloque2
- El Bloque4 se ejecutará antes que el Bloque3 si se produce una excepción de tipo aritmético en el Bloque2
- El Bloque4 se ejecutará antes de que la excepción producida por un acceso a un objeto nulo (null) en el Bloque2 se propague hacia arriba

RESERVA 1: Indique cuál de las siguientes afirmaciones es correcta:

- Los campos se conocen como variables de tipo de clase.
- Los constructores permiten que cada objeto sea preparado adecuadamente cuando es creado.
- El alcance de una variable define la sección de código desde donde la variable puede ser declarada pero no accedida.
- El tiempo de vida de una variable describe el número de veces que es utilizada en un método.

RESERVA 2: Indique cuál de las siguientes afirmaciones es correcta:

- Una superclase es una clase que es implementada por otra.
- Las clases que están vinculadas mediante una relación de herencia forman una jerarquía de herencia.
- Una subclase es una clase que implementa a otra clase.
- La herencia nos permite heredar pero no reutilizar en un nuevo contexto clases que fueron escritas previamente.

PARTE PRÁCTICA [6,5 PUNTOS]:

La práctica del presente curso consiste en diseñar e implementar un sistema integrado de gestión de un taller de vehículos. Todos los dueños de algún tipo de vehículo (moto, coche, furgoneta, etc.) tienen experiencia en llevar su vehículo al taller para algún tipo de revisión y/o reparación. Puede que sea una puesta a punto antes de las vacaciones o algún viaje largo, o debido a un problema concreto o ruido que hace, o alguna fuga de líquido, etc. Hace años, el responsable de un taller no haría más que apuntar en un libro una breve descripción del vehículo, el motivo por el cual ha acudido al taller su dueño y algún número de contacto. Hoy en día, debido en parte a la competencia entre talleres y al deseo de aportar servicios de calidad para poder fidelizar al cliente, los talleres usan sistemas de gestión para todo el proceso de recepción, procesado y facturación de cada vehículo que pasa por el taller.

En general, las funciones que tienen un sistema de gestión de un taller de coches son varias:

- Recepción del vehículo: al entrar un vehículo en el taller hay que generar una ficha con los datos (si ya no forma parte del registro histórico del taller) más importantes (marca, modelo, matrícula, cliente, etc.), dejar constancia del motivo de la visita (problema mecánico, revisión,

etc.) e imprimir el formulario de autorización que el dueño tiene que firmar para autorizar el trabajo.

- Asignación de los trabajos a los mecánicos: a medida que vayan terminando trabajos, el jefe del taller les va asignando nuevos vehículos.
- Procesado de los vehículos: un mecánico, al terminar la reparación de un vehículo, acude al sistema para ver los siguientes trabajos que le corresponden. Según la información del sistema, tiene que averiguar el problema, llevar a cabo el trabajo correspondiente y dejar constancia del proceso realizado en el sistema para que el comercial pueda informar al cliente de que su vehículo está listo para recoger. Una vez que el mecánico termine con un coche, en el sistema le aparece reflejado el siguiente vehículo asignado. Si por el motivo que sea (por ejemplo, falta de piezas o que un cliente no haya dado su autorización a realizar una reparación debido al coste), un mecánico deja un trabajo en un estado sin completar, anota en el sistema el motivo y pasa al siguiente trabajo.
- Gestión de usuarios: altas, bajas, modificaciones de las personas que figuran en el sistema (miembros del taller [jefe, mecánico, comercial] y clientes). La primera vez que acude un cliente al taller hay que darle de alta en el sistema.
- Gestión de clientes por parte del comercial: comunicar a los clientes el precio de una reparación, informarles de que su vehículo está listo para recoger, presentar ofertas especiales (por ejemplo, revisión antes de la inspección técnica del vehículo [ITV] o puesta a punto para las vacaciones, revisión de los neumáticos, frenos).

Se pide realizar las siguientes tareas:

- a) **[1,0 puntos]** Diseñar utilizando un paradigma orientado a objetos, los elementos necesarios para la aplicación explicada de la práctica durante el curso. Es necesario identificar la estructura y las relaciones de herencia (mediante el uso de un diagrama de clases) y de uso de las clases necesarias para almacenar y gestionar esta información. Debe hacerse uso de los mecanismos de herencia siempre que sea posible. Se valorará un buen diseño que favorezca la reutilización de código y facilite su mantenimiento.
- b) **[2,0 puntos]** Implementar un método (o métodos) que permita generar un listado por cada mecánico de los vehículos reparados en un día determinado por dicho mecánico, indicando, para cada uno de los vehículos, otros mecánicos que hayan intervenido, las piezas que se han utilizado y una descripción de cada uno de los trabajos realizados. El listado aparecerá ordenado ascendente por la matrícula del vehículo (en el listado aparecen antes aquellos que tengan una matrícula más antigua). Justifique las opciones y decisiones que se tomen.
- c) **[2,0 puntos]** Implementar un método (o métodos) que implemente el libro de mantenimiento digital. Este libro de mantenimiento consta de toda la información posible acerca de cada una de las reparaciones hechas en el vehículo. El usuario puede acceder online a este servicio y descargar dicha información. Justifique las opciones y decisiones que se tomen.
- d) **[1,5 puntos]** Proporcione un método (o métodos) que permita mostrar por pantalla un formulario básico en **modo gráfico** que permita recoger los parámetros necesarios para dar de alta un nuevo cliente en el sistema. Se pide expresamente la parte gráfica (asumiendo que la lógica de funcionamiento está desarrollada). El objetivo es ver el conocimiento y destreza en el uso de las librerías Swing y/o AWT. No desarrolle código asociado a la funcionalidad del alta. Justifique las opciones y decisiones que se tomen.

**UNIVERSIDAD NACIONAL DE EDUCACIÓN A DISTANCIA – ESCUELA TÉCNICA SUPERIOR DE
INGENIERÍA INFORMÁTICA**
**71901072 – PROGRAMACIÓN ORIENTADA A OBJETOS (GRADO EN INGENIERÍA INFORMÁTICA /
TECNOLOGÍAS DE LA INFORMACIÓN)**
JUNIO 2018 – MODELO B – NO ESTÁ PERMITIDO EL USO DE MATERIAL ADICIONAL

PARTE TEÓRICA - TEST [2,5 PUNTOS]:

Solo una de las respuestas es válida. Las respuestas correctas se puntuarán con +1.0, mientras que las respondidas de manera incorrecta se puntuarán con -0.25. Las no contestadas no tendrán influencia ni positiva ni negativa en la nota.

Las preguntas de reserva sólo tendrán utilidad en el caso de que alguna de las 14 preguntas iniciales del test sea anulada por cualquier circunstancia. Caso de ocurrir este hecho, si se produjera la anulación de alguna de las 14 preguntas iniciales, la primera pregunta de reserva sustituiría a la pregunta anulada. Caso de que una segunda pregunta de las 14 iniciales fuese anulada, entonces la segunda pregunta de reserva sustituiría a esta segunda pregunta anulada. En aquellos hipotéticos casos en los que se produjese la anulación de una tercera o sucesivas preguntas de las 14 iniciales, entonces sólo en ese caso, las preguntas tercera y sucesivas anuladas se considerarían como correctas (al no existir más preguntas de reserva que las sustituyan).

Pregunta 1: Indica cual de las siguientes declaraciones es válida para el método main:

- a. public static void main(String args[]);
- b. static public void main(String);
- c. public static void main(String);
- d. public static int main(String args[]);

Pregunta 2: Indique el orden seguido en los ejemplos del texto de la bibliografía básica de la asignatura en cuanto a la parte interna de una clase:

```
public class NombreClase
{
    PARTE INTERNA DE UNA CLASE
}
```

- a. Constructores, Métodos y Campos
- b. Métodos, Constructores y Campos
- c. Campos, Constructores y Métodos
- d. Campos, Métodos y Constructores

Pregunta 3: Indica cual de las siguientes afirmaciones es correcta:

- a. El lenguaje Java tiene tres tipos de ciclo: while, while-do y for.
- b. En caso de un bucle no relacionado con colecciones el bucle for-each no tiene utilidad.
- c. El tipo de la variable de ciclo no tiene porqué ser el mismo que el tipo del elemento declarado para la colección que estamos recorriendo con un ciclo.
- d. Un índice es un objeto que proporciona funcionalidad para recorrer todos los elementos de una colección.

Pregunta 4: Dado el siguiente fragmento de código:

```
int A = 9;
float B = 3.3F;
char C = 'w';

System.out.println(A + B > 12);
System.out.println(A >= 8 && C != 'w');
System.out.println((C == 'c') || ((A + B) == 12));
```

Indica cual será la salida por pantalla (cada valor en una línea diferente):

- a. true true false
- b. true false true
- c. true false false
- d. false false false

Pregunta 5: Supongamos que queremos implementar una Agenda, ¿cuál sería la salida del siguiente código?

```
public class Agenda {

    public static void main(String argv[]){
        Agenda agenda = new Agenda();
    }

    protected Agenda(){
        for(int i=0; i<10; i++){
            System.out.println(i);
        }
    }
}
```

- a. Error de Compilación ya que los constructores no pueden ser declarados como “protected”.
- b. Error en tiempo de ejecución ya que los constructores no pueden ser declarados como “protected”.
- c. Compilación correcta y salida de los dígitos de 0 a 10.
- d. Compilación correcta y salida de los dígitos de 0 a 9.

Pregunta 6: Indica cual de las siguientes afirmaciones es correcta:

- a. La interfaz de una clase describe lo que una clase hace y cómo se puede utilizar mostrando su implementación.
- b. El código fuente completo que define una clase es la interfaz de dicha clase.
- c. Se dice que un objeto es inmutable si su contenido o estado no puede cambiarse después de crearlo excepto si la interfaz es mutable.
- d. La documentación de la librería de clases Java muestra detalles acerca de todas las clases de la librería.

Pregunta 7: Indique cuales de las siguientes expresiones resultan verdaderas:

1. $!(4 < 5)$
2. $(2 > 2) \text{ || } ((4 == 4) \&\& (1 < 0))$
3. $(2 > 2) \text{ || } (4 == 4) \&\& (1 < 0)$
4. $(2 > 2) \text{ || } !(4 == 4) \&\& (1 < 0))$
5. $(34 != 33) \&\& !\text{false}$

- a. Las expresiones 4 y 5.
- b. Las expresiones 3 y 4.
- c. Las expresiones 2 y 4.
- d. Las expresiones 3 y 5.

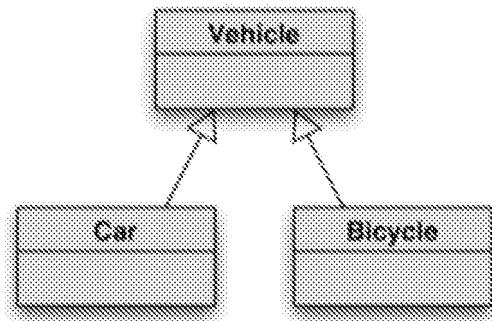
Pregunta 8: Indica cual de las siguientes afirmaciones es correcta:

- a. El término cohesión describe la interconexión de las clases.
- b. El término acoplamiento describe lo bien que una unidad de código se corresponde con una tarea lógica o con una entidad.
- c. La duplicación de código produce errores de ejecución.
- d. Una adecuada encapsulación de las clases reduce el acoplamiento y conduce, por tanto, a un mejor diseño.

Pregunta 9 : Indica cual de las siguientes afirmaciones es correcta:

- a. La prueba es la actividad de descubrir si una pieza de código produce el comportamiento pretendido.
- b. La depuración es el intento de localizar y corregir el origen de un error.
- c. La prueba de unidad se refiere a las pruebas de las partes individuales de una aplicación, como los métodos y las clases.
- d. Todas las respuestas anteriores son correctas.

Pregunta 10: Dada la siguiente jerarquía de herencia:



Indica cual de las siguientes asignaciones es correcta:

- a. `Vehicle v1 = new Vehicle();`
- b. `Vehicle v2 = new Bicycle();`
- c. `Vehicle v3 = new Car();`
- d. Todas las asignaciones anteriores son correctas.

Pregunta 11: Indica cual de las siguientes afirmaciones es correcta:

- a. El tipo dinámico de una variable v es el tipo tal como está declarado en el código fuente.
- b. El tipo estático de una variable v es el tipo del objeto que está almacenado actualmente en v.
- c. Declarar un campo o un método protegido impide acceder directamente a él desde las subclases.
- d. Todas las afirmaciones anteriores son falsas.

Pregunta 12: Indique cual de las siguientes opciones declarará un método en una clase que fuerza a una subclase a implementarlo:

- a. static void methoda (double d1) {}
- b. abstract public void methoda();
- c. public native double methoda();
- d. protected void methoda (double d1){}

Pregunta 13: ¿Qué significa el siguiente fragmento de código Java?: String saludar() { return "Hola"; }

- a. Hay un método “String saludar” que no recibe ningún parámetro de entrada y devuelve el valor “Hola”.
- b. Hay una variable “String” cuyo valor es “saludar() { return “Hola”; }”
- c. Hay un método “saludar” que no recibe ningún parámetro de entrada y devuelve una cadena (String) cuyo valor es “Hola”.
- d. El fragmento no representa un fragmento de código legal en Java.

Pregunta 14: En el siguiente fragmento de código hemos definido la ejecución de cinco bloques. Estos bloques se ejecutarán dependiendo de las excepciones que se produzcan en cada caso.

```
// Bloque1
try{
    // Bloque2
} catch (ArithmeticException e) {
    // Bloque3
} finally{
    // Bloque4
}
// Bloque5
```

Indique cual de las siguientes afirmaciones es correcta:

- a. El Bloque4 se ejecutará antes de que la excepción producida por un acceso a un objeto nulo (null) en el Bloque2 se propague hacia arriba
- b. El Bloque4 no se ejecutará si se produce una excepción de tipo aritmético en el Bloque2
- c. El Bloque4 no se ejecutará si se produce un acceso a un objeto nulo (null) en el Bloque2
- d. El Bloque4 se ejecutará antes que el Bloque3 si se produce una excepción de tipo aritmético en el Bloque2

RESERVA 1: Indique cuál de las siguientes afirmaciones es correcta en relación a BlueJ:

- a. Un punto de interrupción es un indicador asociado a un conjunto de líneas de código.
- b. Los puntos de interrupción se definen mediante la ventana del editor.
- c. Los puntos de interrupción solo se pueden definir dentro de las clases abstractas.
- d. Todas las afirmaciones anteriores son falsas.

RESERVA 2: Dado el siguiente fragmento de código,

```
int electrodomestico = 1;  
boolean[] ventas = new boolean[3];  
boolean financiado = ventas [electrodomestico];
```

Indica cual de las siguientes afirmaciones es correcta en relación al valor de la variable aprobado.

- a. financiado tiene el valor false
- b. financiado tiene el valor 0
- c. financiado tiene el valor null
- d. Se produce una excepción y financiado no posee ningún valor

PARTE PRÁCTICA [6,5 PUNTOS]:

La Práctica del presente curso va a consistir en diseñar e implementar un sistema integrado de gestión de una tienda de electrodomésticos. Hoy en día las tiendas de electrodomésticos además de disponer de una gran cantidad de productos en sus tiendas, disponen de diferentes perfiles de empleados (técnicos, cajeros, financiación y postventa) para atender de la mejor manera posible a sus clientes. Además, los clientes disponen de un perfil que además de sus datos personales incluyen un historial de compras, generación de facturas, descarga de manuales, comprobación de estado de garantía y promociones. De esta forma, la práctica consiste en desarrollar un sistema de gestión que englobe todas estas características teniendo en cuenta un diseño orientado a objetos.

En general, las funciones que tienen un sistema de gestión de una tienda de electrodomésticos son varias:

- Venta de un electrodoméstico (cajero): cuando un cliente pasa por la línea de cajas es necesario generar una ficha de cliente en el caso de que no disponga de ella. El identificador principal es el DNI y los datos más importantes son el nombre, apellidos, dni, domicilio y número de teléfono. Esta ficha tendrá disponible un histórico de los productos comprados y su fecha de adquisición. En el caso de solicitar financiación, deberá constar en la ficha y el cliente debería pasar por la oficina de financiación para obtener el visto bueno.
- Financiación (financiación): El empleado de la oficina de financiación recibirá clientes que previamente hayan pasado por la línea de cajas para comprar productos y analizará la ficha de financiación. Solicitará la última nómina al cliente, dejando constancia de la cantidad en la ficha del cliente y en caso de que el cargo mensual no supere el 15% de la nómina en un máximo de financiación de 60 meses, la financiación se aprobará.
- Reparación de Electrodomésticos (técnico): Los clientes podrán llevar sus productos comprados en la tienda a reparar. Las condiciones de reparación serán las siguientes: reparación gratuita en los dos primeros años. A partir de esa fecha, se cargará un importe al cliente dependiendo de la reparación efectuada.
- Devolución de electrodoméstico (postventa): Un cliente, presentando su DNI, podrá devolver uno o varios electrodomésticos en el caso de que el periodo de compra no supera los 3 meses.
- Gestión comercial (comercial): Este empleado generará una serie de comunicaciones con el cliente ofreciendo diferentes posibilidades de compra.
- Gestión de usuarios: altas, bajas, modificaciones de las personas que figuran en el sistema (empleados -- técnicos, cajeros, financiación, postventa y comerciales -- y clientes). La primera vez que acude un cliente a la tienda hay que darle de alta en el sistema.
- Gestión de clientes: Cada uno de los empleados tendrán un tipo de relación con el cliente teniendo que dejar constancia en la ficha del cliente la operación realizada y sus detalles (cliente, empleado que atiende, tipo de operación, productos involucrados, fecha, etc.)

Se pide realizar las siguientes tareas:

- a) **[1,0 puntos]** Diseñar utilizando un paradigma orientado a objetos, los elementos necesarios para la aplicación explicada de la práctica durante el curso. Es necesario identificar la estructura y las relaciones de herencia (mediante el uso de un diagrama de clases) y de uso de las clases necesarias para almacenar y gestionar esta información. Debe hacerse uso de los mecanismos de herencia siempre que sea posible. Se valorará un buen diseño que favorezca la reutilización de código y facilite su mantenimiento.
- b) **[2,0 puntos]** Implementar un método (o métodos) que permita generar un listado por cada cliente de los electrodomésticos comprados en un día determinado, indicando, para cada uno de los electrodomésticos, otros empleados que hayan intervenido, como el cajero, financiación o postventa. El listado aparecerá ordenado ascendente por el precio de venta del electrodoméstico (en el listado aparecen antes aquellos que tengan un precio menor). Justifique las opciones y decisiones que se tomen.
- c) **[2,0 puntos]** Implementar un método (o métodos) que implemente la garantía digital. Esta garantía consta de toda la información posible acerca de la venta, reparaciones y financiación hechas en el electrodoméstico. El usuario puede acceder online a este servicio y descargar dicha información. Justifique las opciones y decisiones que se tomen.
- d) **[1,5 puntos]** Implementar un método (o métodos) que permita gestionar las diferentes ofertas y promociones que puede ofrecer la tienda: ofertas en los principales periodos de compra (navidades, días especiales, etc.) y rebajas en los dos períodos típicos (Enero, Julio, Black Friday, etc.).

**UNIVERSIDAD NACIONAL DE EDUCACIÓN A DISTANCIA – ESCUELA TÉCNICA SUPERIOR DE
INGENIERÍA INFORMÁTICA**
**71901072 – PROGRAMACIÓN ORIENTADA A OBJETOS (GRADO EN INGENIERÍA INFORMÁTICA /
TECNOLOGÍAS DE LA INFORMACIÓN)**
JUNIO – MODELO A – NO ESTÁ PERMITIDO EL USO DE MATERIAL ADICIONAL

PARTE TEÓRICA - TEST [2,5 PUNTOS]:

Solo una de las respuestas es válida. Las respuestas correctas se puntuarán con +1.0, mientras que las respondidas de manera incorrecta se puntuarán con -0.25. Las no contestadas no tendrán influencia ni positiva ni negativa en la nota.

Las preguntas de reserva sólo tendrán utilidad en el caso de que alguna de las 14 preguntas iniciales del test sea anulada por cualquier circunstancia. Caso de ocurrir este hecho, si se produjera la anulación de alguna de las 14 preguntas iniciales, la primera pregunta de reserva sustituiría a la pregunta anulada. Caso de que una segunda pregunta de las 14 iniciales fuese anulada, entonces la segunda pregunta de reserva sustituiría a esta segunda pregunta anulada. En aquellos hipotéticos casos en los que se produjese la anulación de una tercera o sucesivas preguntas de las 14 iniciales, entonces sólo en ese caso, las preguntas tercera y sucesivas anuladas se considerarían como correctas (al no existir más preguntas de reserva que las sustituyan).

Pregunta 1: ¿Cuál de las siguientes sentencias se ejecuta de manera correcta?

- a. String animales [] = {"Perro", "Gato", "Lobo"};
- b. String animales = {"Perro", "Gato", "Lobo"};
- c. String animales [] = new String {"Perro" "Gato" "Lobo"};
- d. String animales [] = { "Perro" "Gato" "Lobo"};

Pregunta 2: ¿En qué condiciones puede volverse a invocar un constructor de una clase para un objeto después de que ese objeto haya sido creado?

- a. Cuando queremos resetear todos los campos del objeto a sus valores iniciales.
- b. Cuando se ha creado un objeto abstracto y se le quiere dar valores iniciales a sus atributos.
- c. Cuando se implementa una interfaz para el objeto en cuestión.
- d. Nunca.

Pregunta 3: El uso de índices fuera de los límites legales de una matriz hará que se produzca ...

- a. Se produce un error en tiempo de ejecución denominado ArrayIndexOutOfBoundsException
- b. Se produce un error en tiempo de compilación denominado ArrayIndexOutOfBoundsException
- c. Se produce un error en tiempo de ejecución denominado ArrayIndexOutOfBoundsException
- d. Se produce un error en tiempo de compilación denominado ArrayIndexOutOfBoundsException

Pregunta 4: Un método cohesionado ...

- a. Será responsable de al menos una tarea bien definida, pero puede serlo de más.
- b. Será responsable, idealmente, de una y sólo una tarea o entidad bien definida coherente.
- c. Es aquel método abstracto que se ha instanciado en una clase determinada.
- d. Es aquel que se crea en una clase interna para ser invocado desde la clase circundante.

Pregunta 5: Sea el siguiente fragmento de código modificado de la clase MailItem mostrada en el libro de texto:

```
1  public class MailItem {  
2      static int num1 = 10;  
3      public static void main (String args []) {  
4          int num2 = 5;  
5          new MailItem ();  
6      }  
7      public MailItem () {  
8          int aux = this.num2;  
9          if (aux > 1) {  
10              System.out.println(aux);  
11          }  
12      }  
13  }
```

¿Cuál es el resultado que produce?

- a. Se produce un error de compilación.
- b. Se produce un error de ejecución.
- c. No produce ningún error pero no muestra nada por pantalla.
- d. No se produce ningún error y muestra por pantalla el valor 5.

Pregunta 6: Sea el siguiente código modificado de la clase MusicOrganizer mostrada en el libro base:

```
1  import java.util.*;  
2  public class MusicOrganizer {  
3      public static void main (String args []) {  
4          ArrayList <String> a = new ArrayList ();  
5          for (int i=0; i<=5; i++)  
6          {  
7              a.add("Hola");  
8          }  
9          System.out.println("Funciona");  
10     }  
11 }
```

¿Cuál es el resultado de compilar y ejecutar este código?

- a. Se produce un error de ejecución al definir un ArrayList de 5 elementos y querer insertar 6 elementos.
- b. No se produce ningún tipo de error y proporciona el resultado por pantalla “Funciona”.
- c. La línea 4 provoca un warning pero se ejecuta sin problemas proporcionando el resultado por pantalla “Funciona”.
- d. La línea 7 provoca un warning pero se ejecuta sin problemas proporcionando el resultado por pantalla “Funciona”.

Pregunta 7: Supongamos que reescribimos el ejemplo BouncingBall del libro de la forma en que se muestra a continuación:

```
1  public class BouncingBall {  
2      int n;  
3      public static void main (String args [])  
4      {  
5          if (n!=0)  
6          {  
7              n = n + 1;  
8              System.out.println("El número es " + n);  
9          }  
10     }  
11 }
```

¿Cuál es la línea que provoca que el código produzca uno o varios errores de compilación?

- a. Línea 5.
- b. Línea 3.
- c. Línea 2.
- d. Línea 1.

Pregunta 8: Supongamos que reescribimos el ejemplo BouncingBall del libro de la forma en que se muestra a continuación:

```
1  import java.util.Random;  
2  public class BouncingBall {  
3      public static void main (String args []) {  
4          Random randomGenerator;  
5          randomGenerator = new Random();  
6          int index = randomGenerator.nextInt();  
7          System.out.println(index);  
8      }  
9  }
```

¿Cuál es la línea que provoca que el código produzca uno o varios errores de compilación?

- a. No se produce error de compilación
- b. En la línea 4
- c. En la línea 5
- d. En la línea 6

Pregunta 9: ¿Cómo se llama el entorno de pruebas que soporta la prueba estructurada de unidades y las pruebas de regresión en Java?

- a. JDK
- b. JBoss
- c. Javadoc
- d. JUnit

Pregunta 10: Respecto al constructor de la subclase ...

- a. Debe siempre invocar al constructor de su superclase como primera instrucción. Si no incluye esta llamada, Java intentará insertar una llamada automáticamente.
- b. No debe invocar nunca al constructor de su superclase como primera instrucción. Si la incluye esta llamada, Java ignorará esta llamada automáticamente.
- c. Debe siempre invocar al constructor de su superclase como última instrucción. Si no incluye esta llamada, Java intentará insertar una llamada automáticamente.
- d. Debe siempre invocar al constructor de su superclase como última instrucción. Si no incluye esta llamada, Java generará un error de compilación.

Pregunta 11: Respecto a las variables polimórficas ...

- a. Toda variable de objeto en Java es potencialmente polimórfica.
- b. Son aquellas que exclusivamente pertenecen a clases abstractas.
- c. Son la instancia de una clase abstracta, permitiendo sólo almacenar objetos de ese tipo.
- d. Son aquellas que implementan una interfaz y que provienen de una clase abstracta.

Pregunta 12: Si una clase B extiende una clase abstracta A que tiene un método abstracto met, ¿qué podemos afirmar?

- a. Que necesariamente B es abstracta.
- b. Que si B implementa el método met, entonces seguro que B no es abstracta.
- c. Que no puedo crear instancias de A.
- d. Que puedo crear instancias de A.

Pregunta 13: Se define como excepción no comprobada ...

- a. Aquellas subclases de la clase estándar RunnertimeException
- b. Aquellas subclases de la clase estándar RunnableTimeException
- c. Aquellas subclases de la clase estándar RunningtimeException
- d. Aquellas subclases de la clase estándar RuntimeException

Pregunta 14: Según el texto de la bibliografía básica de la asignatura, ¿cuál de las siguientes afirmaciones NO es correcta?

- a. Los campos almacenan datos de manera no persistente dentro de un objeto.
- b. Los constructores son responsables de garantizar que un objeto se configure apropiadamente en el momento de crearlo por primera vez.
- c. Los métodos implementan el comportamiento de un objeto; proporcionan su funcionalidad.
- d. Los campos se definen fuera de los constructores y métodos.

RESERVA 1: ¿Cuál de las siguientes es una característica de la clase java.lang.Error?

- a. extends Exception.
- b. implements Throwable.
- c. implements Exception.
- d. Ninguna de las anteriores.

RESERVA 2: Según el texto de la bibliografía básica de la asignatura, ¿qué se entiende por abstracción?

- a. Proceso de dividir un todo en partes bien definidas que puedan construirse y examinarse por separado y que interactúen de formas bien definidas.
- b. Capacidad de ignorar los detalles de las distintas partes, para centrar la atención en un nivel superior de un problema.
- c. Capacidad de dividir un todo en partes bien definidas ignorando los detalles de las distintas partes.
- d. Ninguna de las anteriores.

PARTE PRÁCTICA [6,5 PUNTOS]:

La Práctica del presente curso va a consistir en diseñar e implementar un sistema integrado de gestión de una tienda de electrodomésticos. Hoy en día las tiendas de electrodomésticos además de disponer de una gran cantidad de productos en sus tiendas, disponen de diferentes perfiles de empleados (técnicos, cajeros, financiación y postventa) para atender de la mejor manera posible a sus clientes. Además, los clientes disponen de un perfil que además de sus datos personales incluyen un historial de compras, generación de facturas, descarga de manuales, comprobación de estado de garantía y promociones. De esta forma, la práctica consiste en desarrollar un sistema de gestión que englobe todas estas características teniendo en cuenta un diseño orientado a objetos.

En general, las funciones que tienen un sistema de gestión de una tienda de electrodomésticos son varias:

- Venta de un electrodoméstico (cajero): cuando un cliente pasa por la línea de cajas es necesario generar una ficha de cliente en el caso de que no disponga de ella. El identificador principal es el DNI y los datos más importantes son el nombre, apellidos, dni, domicilio y número de teléfono. Esta ficha tendrá disponible un histórico de los productos comprados y su fecha de adquisición. En el caso de solicitar financiación, deberá constar en la ficha y el cliente debería pasar por la oficina de financiación para obtener el visto bueno.
- Financiación (financiación): El empleado de la oficina de financiación recibirá clientes que previamente hayan pasado por la línea de cajas para comprar productos y analizará la ficha de financiación. Solicitará la última nómina al cliente, dejando constancia de la cantidad en la ficha del cliente y en caso de que el cargo mensual no supere el 15% de la nómina en un máximo de financiación de 60 meses, la financiación se aprobará.
- Reparación de Electrodomésticos (técnico): Los clientes podrán llevar sus productos comprados en la tienda a reparar. Las condiciones de reparación serán las siguientes: reparación gratuita en los dos primeros años. A partir de esa fecha, se cargará un importe al cliente dependiendo de la reparación efectuada.
- Devolución de electrodoméstico (postventa): Un cliente, presentando su DNI, podrá devolver uno o varios electrodomésticos en el caso de que el periodo de compra no supera los 3 meses.
- Gestión comercial (comercial): Este empleado generará una serie de comunicaciones con el cliente ofreciendo diferentes posibilidades de compra.
- Gestión de usuarios: altas, bajas, modificaciones de las personas que figuran en el sistema (empleados -- técnicos, cajeros, financiación, postventa y comerciales -- y clientes). La primera vez que acude un cliente a la tienda hay que darle de alta en el sistema.
- Gestión de clientes: Cada uno de los empleados tendrán un tipo de relación con el cliente teniendo que dejar constancia en la ficha del cliente la operación realizada y sus detalles (cliente, empleado que atiende, tipo de operación, productos involucrados, fecha, etc.)

Se pide realizar las siguientes tareas:

- a) **[1,0 puntos]** Diseñar utilizando un paradigma orientado a objetos, los elementos necesarios para la aplicación explicada de la práctica durante el curso. Es necesario identificar la estructura y las relaciones de herencia (mediante el uso de un diagrama de clases) y de uso de las clases necesarias para almacenar y gestionar esta información. Debe hacerse uso de los mecanismos de herencia siempre que sea posible. Se valorará un buen diseño que favorezca la reutilización de código y facilite su mantenimiento.

- b) **[2,0 puntos]** Implementar un método (o métodos) que permita generar un listado por fabricante de los electrodomésticos que hay en stock, indicando, para cada uno de los fabricantes, el número total de electrodomésticos diferentes existente. El listado aparecerá ordenado descendente por el nombre del fabricante, y dentro de él, los electrodomésticos de ese fabricante aparecerán ordenados ascendente por precio de venta del electrodoméstico (en el listado aparecen antes aquellos que tengan un precio menor). Justifique las opciones y decisiones que se tomen.
- c) **[2,0 puntos]** Implementar un método (o métodos) que desarrolle una tarjeta de puntos del cliente. Cada cliente acumulará el 10% del precio de cada compra realizada, de modo que esa cantidad puede ser usada en una próxima compra. Cuando se va a comprar, si hay disponibles puntos, podrán usarse hasta para pagar el 50% de la siguiente compra. Por ejemplo, si tenemos 240 puntos acumulados y compramos un lavavajillas de 300 €, podremos usar hasta 150 puntos (el 50% del precio total de la compra), y pagar el resto (150 €) en dinero. De este modo, al finalizar la compra, se habrán consumido 150 puntos y se habrán pagado 150 €, quedando en la tarjeta de puntos del cliente 90 puntos restantes (a los 240 puntos que tenía se le han descontado los 150 puntos empleados en la compra). Justifique las opciones y decisiones que se tomen.
- d) **[1,5 puntos]** Implementar un método (o métodos) gestionar la facturación de un cliente. Así, el sistema generará, para un cliente determinado, un listado de todas las facturas que el cliente ha generado entre dos meses suministrados a la aplicación, y proporcionará también la información de las diferentes ofertas que se le han aplicado al mismo. La información la mostrará desglosada por meses, así como el resumen global. Justifique las opciones y decisiones que se tomen.

**UNIVERSIDAD NACIONAL DE EDUCACIÓN A DISTANCIA – ESCUELA TÉCNICA SUPERIOR DE
INGENIERÍA INFORMÁTICA**
**71901072 – PROGRAMACIÓN ORIENTADA A OBJETOS (GRADO EN INGENIERÍA INFORMÁTICA /
TECNOLOGÍAS DE LA INFORMACIÓN)**
SEPTIEMBRE 2018 – MODELO A – NO ESTÁ PERMITIDO EL USO DE MATERIAL ADICIONAL

PARTE TEÓRICA - TEST [2,5 PUNTOS]:

Solo una de las respuestas es válida. Las respuestas correctas se puntuarán con +1.0, mientras que las respondidas de manera incorrecta se puntuarán con -0.25. Las no contestadas no tendrán influencia ni positiva ni negativa en la nota.

Las preguntas de reserva sólo tendrán utilidad en el caso de que alguna de las 14 preguntas iniciales del test sea anulada por cualquier circunstancia. Caso de ocurrir este hecho, si se produjera la anulación de alguna de las 14 preguntas iniciales, la primera pregunta de reserva sustituiría a la pregunta anulada. Caso de que una segunda pregunta de las 14 iniciales fuese anulada, entonces la segunda pregunta de reserva sustituiría a esta segunda pregunta anulada. En aquellos hipotéticos casos en los que se produjese la anulación de una tercera o sucesivas preguntas de las 14 iniciales, entonces sólo en ese caso, las preguntas tercera y sucesivas anuladas se considerarían como correctas (al no existir más preguntas de reserva que las sustituyan).

Pregunta 1: Dado el siguiente fragmento de código:

```
float A = 3.3F;
char B = 'z';
int C = 3;

System.out.println(A + B < 12);
System.out.println(A >= 8 || C != 'w');
System.out.println((C == 'c') && ((A + B) == 12));
```

Indica cual será la salida por pantalla (cada valor en una línea diferente):

- a. false true false
- b. true false true
- c. true false false
- d. false false false

Pregunta 2: ¿Qué significa el siguiente fragmento de código Java?: **String saludar() { return "Hola"; }**

- a. Hay un método “saludar” que no recibe ningún parámetro de entrada y devuelve una cadena cuyo valor es “Hola”.
- b. Hay un método “String saludar” que no recibe ningún parámetro de entrada y devuelve el valor “Hola”.
- c. Hay una variable “String” cuyo valor es “saludar() { return “Hola”; }”
- d. El fragmento no representa un fragmento de código legal en Java.

Pregunta 3: ¿Cuál de las siguientes sentencias se ejecuta de manera correcta?

- a. String coches [] = new String {"BMW" "AUDI" "SEAT"};
- b. String coches = {"BMW", "AUDI", "SEAT"};
- c. String coches [] = { "BMW" "AUDI" "SEAT"};
- d. String coches [] = { "BMW", "AUDI", "SEAT"};

Pregunta 4: Según el libro de la asignatura, la duplicación de código es un síntoma de:

- a. Buena cohesión.
- b. La solución inevitable de un problema complejo.
- c. Mala cohesión.
- d. Mal encapsulamiento.

Pregunta 5: Queremos compilar el siguiente código que se puede encontrar en el texto base de la asignatura y que ha sido modificado convenientemente. ¿Cuál es el resultado que obtenemos al compilar?

```
public class Prueba {  
    public static void main (String args[]) {  
        String cadena1 = new String("ejemPLo");  
        String cadena2 = new String("ejemplo");  
        cadena1 = cadena1.toLowerCase();  
        if (cadena1.toString().equals(cadena2.toString())) {  
            System.out.println("Son iguales");  
        }  
        else {  
            System.out.println("Son diferentes");  
        }  
    }  
}
```

- a. Se produce una excepción y la ejecución falla.
- b. Se imprime por pantalla el mensaje: Son diferentes.
- c. Se imprime por pantalla el mensaje: Son iguales.
- d. Ninguna de las anteriores.

Pregunta 6: Indica cual de las siguientes afirmaciones es correcta:

- a. A partir de una clase tan solo se puede crear un solo objeto.
- b. Los métodos pueden devolver información de algún objeto mediante un valor de retorno.
- c. Los métodos siempre tienen parámetros con los que obtener la información necesaria.
- d. El estado de los objetos se representa mediante los métodos implementados.

Pregunta 7: Dado el siguiente fragmento de código:

```
1. public class Suma {  
2.     static int n;  
3.     public static void main (String args []) {  
4.         int j = 0;  
5.         for (int j = 0; j++; j < 10) {  
6.             if (n!=0) {  
7.                 n = n + j;  
8.                 System.out.println("El número es " + n);  
9.             }  
10.        }  
11.    }  
12. }
```

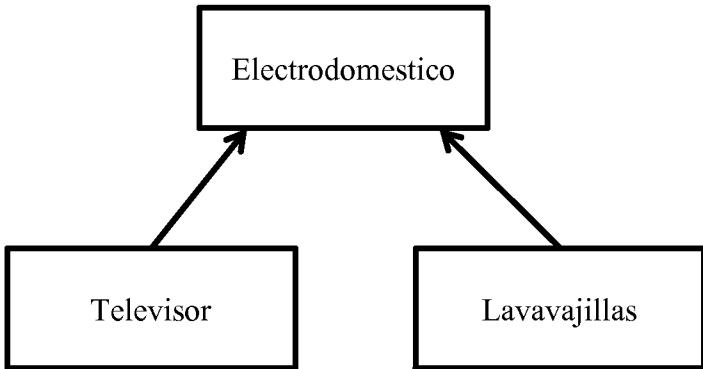
¿Cuál es la línea que provoca que el código produzca uno o varios errores de compilación?

- a. No se produce error de compilación
- b. En la línea 4
- c. En la línea 5
- d. En la línea 6

Pregunta 8: Indica cual de las siguientes afirmaciones es correcta:

- a. Un objeto de tipo String puede ser modificado una vez que está creado, por tanto no es un ejemplo de objeto inmutable
- b. Un objeto es inmutable si su contenido o su estado no puede ser cambiado una vez que se ha creado
- c. La clase String tiene un método de nombre trim que permite modificar caracteres en cualquier posición de una cadena
- d. Las cadenas de texto de tipo String solamente se pueden comparar mediante el operador “==”

Pregunta 9: Dada la siguiente jerarquía de herencia:



Indica cual de las siguientes asignaciones es correcta:

- a. Televisor t1 = new Electrodomestico();
- b. Lavavajillas l1 = new Electrodomestico();
- c. Electrodomestico e1 = new Lavavajillas();
- d. Lavavajillas l1 = new Televisor();

Pregunta 10: Un método de acceso o selector:

- a. Permite acceder al constructor de la clase que lo define.
- b. Habitualmente devuelve void.
- c. Devuelve siempre información sobre el estado de un objeto.
- d. Devuelve siempre un objeto de la clase Object.

Pregunta 11: Respecto a las clases internas ...

- a. Las instancias de la clase interna no están necesariamente asociadas a instancias de la clase circundante.
- b. Presentan un acoplamiento muy estrecho con la clase circundante.
- c. No se consideran una parte de la clase circundante.
- d. No pueden acceder a los métodos privados de la clase circundante.

Pregunta 12: Sea el siguiente fragmento de código modificado de la clase `MailItem` mostrada en el libro de texto:

```
1  public class MailItem {  
2      static int num1 = 10;  
3      public static void main (String args []) {  
4          int num2 = 5;  
5          new MailItem ();  
6      }  
7      public MailItem () {  
8          int aux = this.num1;  
9          if (aux > 1) {  
10              System.out.println(aux);  
11          }  
12      }  
13  }
```

¿Cuál es el resultado que produce?

- a. Se produce un error de compilación.
- b. Se produce un error de ejecución.
- c. No produce ningún error pero no muestra nada por pantalla.
- d. No se produce ningún error y muestra por pantalla el valor 10.

Pregunta 13: Si llamamos al método `toUpperCase()` de una instancia (`nombre`) de la clase `String` de la siguiente forma:

```
patata.toUpperCase();
```

¿Cuál sería el resultado de compilar y/o ejecutar el código?

- a. El código generaría un error de ejecución.
- b. El código convertiría la cadena `nombre` en mayúsculas.
- c. El código generaría un error de compilación.
- d. Ninguna de las anteriores.

Pregunta 14: Dado el siguiente fragmento de código:

```
1.  public class Suma {  
2.      static int n;  
3.      public static void main (String args []) {  
4.          for (int j = 0; j < 10; j++) {  
5.              if (n!=0) {  
6.                  n = n + j;  
7.                  System.out.println("El número es " + n);  
8.              }  
9.          }  
10.     }  
11. }
```

¿Cuál es la línea que provoca que el código produzca uno o varios errores de compilación?

- a. En la línea 2
- b. En la línea 5
- c. No se produce error de compilación
- d. En la línea 6

RESERVA 1: Según el texto de la bibliografía básica de la asignatura, ¿qué se entiende por abstracción?

- Capacidad de ignorar los detalles de las distintas partes, para centrar la atención en un nivel superior de un problema.
- Capacidad de dividir un todo en partes bien definidas ignorando los detalles de las distintas partes.
- Proceso de dividir un todo en partes bien definidas que puedan construirse y examinarse por separado y que interactúen de formas bien definidas.
- Ninguna de las anteriores.

RESERVA 2: Dado el siguiente fragmento de código que pretende mostrar un ejemplo de sobrescritura:

```
class Examen {  
    private float pregunta = 1.0f ;  
    protected float getNota () {return pregunta;}  
}  
  
class Test extends Examen {  
    private float nota = 2.0f;  
    //Insertar código aquí  
}
```

Indique cual de las siguientes opciones completaría el código anterior para dar lugar a un ejemplo correcto de sobrescritura:

- public float getNota (float valor) { return valor;}
- float getNota () { return nota;}
- float double getNota () { return nota;}
- public float getNota () { return nota;}

PARTE PRÁCTICA [6,5 PUNTOS]:

La Práctica del presente curso va a consistir en diseñar e implementar un sistema integrado de gestión de una tienda de electrodomésticos. Hoy en día las tiendas de electrodomésticos además de disponer de una gran cantidad de productos en sus tiendas, disponen de diferentes perfiles de empleados (técnicos, cajeros, financiación y postventa) para atender de la mejor manera posible a sus clientes. Además, los clientes disponen de un perfil que además de sus datos personales incluyen un historial de compras, generación de facturas, descarga de manuales, comprobación de estado de garantía y promociones. De esta forma, la práctica consiste en desarrollar un sistema de gestión que englobe todas estas características teniendo en cuenta un diseño orientado a objetos.

En general, las funciones que tienen un sistema de gestión de una tienda de electrodomésticos son varias:

- Venta de un electrodoméstico (cajero): cuando un cliente pasa por la línea de cajas es necesario generar una ficha de cliente en el caso de que no disponga de ella. El identificador principal es el DNI y los datos más importantes son el nombre, apellidos, dni, domicilio y número de teléfono. Esta ficha tendrá disponible un histórico de los productos comprados y su fecha de adquisición. En el caso de solicitar financiación, deberá constar en la ficha y el cliente debería pasar por la oficina de financiación para obtener el visto bueno.
- Financiación (financiación): El empleado de la oficina de financiación recibirá clientes que previamente hayan pasado por la línea de cajas para comprar productos y analizará la ficha de financiación. Solicitará la última nómina al cliente, dejando constancia de la cantidad en la ficha del cliente y en caso

de que el cargo mensual no supere el 15% de la nómina en un máximo de financiación de 60 meses, la financiación se aprobará.

- Reparación de Electrodomésticos (técnico): Los clientes podrán llevar sus productos comprados en la tienda a reparar. Las condiciones de reparación serán las siguientes: reparación gratuita en los dos primeros años. A partir de esa fecha, se cargará un importe al cliente dependiendo de la reparación efectuada.
- Devolución de electrodoméstico (postventa): Un cliente, presentando su DNI, podrá devolver uno o varios electrodomésticos en el caso de que el periodo de compra no supera los 3 meses.
- Gestión comercial (comercial): Este empleado generará una serie de comunicaciones con el cliente ofreciendo diferentes posibilidades de compra.
- Gestión de usuarios: altas, bajas, modificaciones de las personas que figuran en el sistema (empleados -- técnicos, cajeros, financiación, postventa y comerciales -- y clientes). La primera vez que acude un cliente a la tienda hay que darle de alta en el sistema.
- Gestión de clientes: Cada uno de los empleados tendrán un tipo de relación con el cliente teniendo que dejar constancia en la ficha del cliente la operación realizada y sus detalles (cliente, empleado que atiende, tipo de operación, productos involucrados, fecha, etc.)

Se pide realizar las siguientes tareas:

- a) **[1,0 puntos]** Diseñar utilizando un paradigma orientado a objetos, los elementos necesarios para la aplicación explicada de la práctica durante el curso. Es necesario identificar la estructura y las relaciones de herencia (mediante el uso de un diagrama de clases) y de uso de las clases necesarias para almacenar y gestionar esta información. Debe hacerse uso de los mecanismos de herencia siempre que sea posible. Se valorará un buen diseño que favorezca la reutilización de código y facilite su mantenimiento.
- b) **[1,5 puntos]** Implementar los métodos que gestionan las altas, bajas y modificaciones de las personas que figuran en el sistema (empleados -- técnicos, cajeros, financiación, postventa, comerciales y clientes). La primera vez que acude un cliente a la tienda hay que darle de alta en el sistema.
- c) **[2,0 puntos]** Implementar un método (o métodos) que desarrolle la reparación de los electrodomésticos. El sistema tendrá que recuperar los datos de venta del aparato. Se almacenará información sobre la avería. Si se trata de algo pequeño el cliente puede llevarlo a la tienda para la reparación. Si es más grande tendrá que ir un técnico a su casa. Si el aparato está en garantía de dos años se hará la reparación de una forma gratuita. De otra manera, el cliente tendrá que pagar.
- d) **[2,0 puntos]** Implementar un método (o métodos) que permita(n) asignar a los usuarios clientes ofertas en función de las compras realizadas anteriormente. De este modo, los clientes que hayan gastado en sus últimas compras más de 999 euros recibirán un descuento de 100 euros en su próxima compra; los que hayan gastado más de 1.999 euros recibirán un descuento de 200 euros y los que hayan gastado más de 2.999 euros recibirán un descuento de 300 euros. Una vez aplicado el descuento, las compras tenidas en cuenta para la oferta no deberán volver a ser consideradas. Justifíquense las opciones y decisiones que se tomen.

**UNIVERSIDAD NACIONAL DE EDUCACIÓN A DISTANCIA – ESCUELA TÉCNICA SUPERIOR DE
INGENIERÍA INFORMÁTICA**
**71901072 – PROGRAMACIÓN ORIENTADA A OBJETOS (GRADO EN INGENIERÍA INFORMÁTICA /
TECNOLOGÍAS DE LA INFORMACIÓN)**
SEPTIEMBRE 2018 – MODELO B – NO ESTÁ PERMITIDO EL USO DE MATERIAL ADICIONAL

PARTE TEÓRICA - TEST [2,5 PUNTOS]:

Solo una de las respuestas es válida. Las respuestas correctas se puntuarán con +1.0, mientras que las respondidas de manera incorrecta se puntuarán con -0.25. Las no contestadas no tendrán influencia ni positiva ni negativa en la nota.

Las preguntas de reserva sólo tendrán utilidad en el caso de que alguna de las 14 preguntas iniciales del test sea anulada por cualquier circunstancia. Caso de ocurrir este hecho, si se produjera la anulación de alguna de las 14 preguntas iniciales, la primera pregunta de reserva sustituiría a la pregunta anulada. Caso de que una segunda pregunta de las 14 iniciales fuese anulada, entonces la segunda pregunta de reserva sustituiría a esta segunda pregunta anulada. En aquellos hipotéticos casos en los que se produjese la anulación de una tercera o sucesivas preguntas de las 14 iniciales, entonces sólo en ese caso, las preguntas tercera y sucesivas anuladas se considerarían como correctas (al no existir más preguntas de reserva que las sustituyan).

Pregunta 1: Dado el siguiente fragmento de código:

```
1. public class Suma {  
2.     static int n;  
3.     public static void main (String args []) {  
4.         for (int j = 0; j++; j < 10) {  
5.             if (n!=0) {  
6.                 n = n + j;  
7.                 System.out.println("El número es " + n);  
8.             }  
9.         }  
10.    }  
11. }
```

¿Cuál es la línea que provoca que el código produzca uno o varios errores de compilación?

- a. No se produce error de compilación
- b. En la línea 4
- c. En la línea 5
- d. En la línea 6

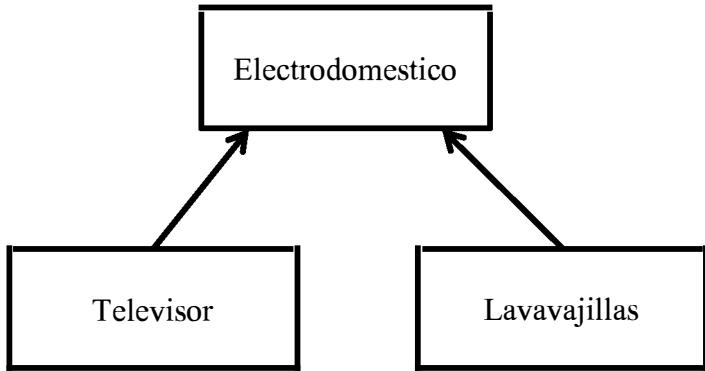
Pregunta 2: Dado el siguiente fragmento de código:

```
int A = 12;  
float C = 9.9F;  
char B = 'a';  
  
System.out.println(A + B < 12);  
System.out.println(A >= 8 || C != 'w');  
System.out.println((C == 'c') && ((A + B) == 12));
```

Indica cual será la salida por pantalla (cada valor en una línea diferente):

- a. true false true
- b. false true false
- c. true false false
- d. False false false

Pregunta 3: Dada la siguiente jerarquía de herencia:



Indica cual de las siguientes asignaciones es correcta:

- a. `Electrodomestico e1 = new Lavavajillas();`
- b. `Televisor t1 = new Electrodomestico();`
- c. `Lavavajillas l1 = new Electrodomestico();`
- d. Todas las asignaciones anteriores son correctas.

Pregunta 4: Un método de modificación o mutador:

- a. Devuelve siempre información sobre el estado de un objeto.
- b. Permite modificar el estado únicamente de los campos públicos de la clase.
- c. Habitualmente devuelve void.
- d. Permite acceder al constructor de la clase que lo define.

Pregunta 5: Se dice que un objeto es inmutable si:

- a. Existirá más que una copia de su contenido o estado después de su creación.
- b. Su contenido o estado es visible fuera de la clase en la que está definido.
- c. Su contenido o estado cambia después de su creación.
- d. Su contenido o estado no puede cambiarse después de su creación.

Pregunta 6: Según el texto de la bibliografía básica de la asignatura, ¿qué almacenan las variables declaradas a partir de una clase?

- a. Objetos.
- b. Copias de objetos.
- c. Referencias a objetos.
- d. Ninguna de las anteriores.

Pregunta 7: Queremos compilar el siguiente código que se puede encontrar en el texto base de la asignatura y que ha sido modificado convenientemente. ¿Cuál es el resultado que obtenemos al compilar?

```
public class Prueba {  
    public static void main (String args[]) {  
        String cadena1 = new String("ejemPLo");  
        String cadena2 = new String("ejemplo");  
        cadena1.toLowerCase();  
        if (cadena1.toString().equals(cadena2.toString())) {  
            System.out.println("Son iguales");  
        }  
        else {  
            System.out.println("Son diferentes");  
        }  
    }  
}
```

- a. Se produce una excepción y la ejecución falla.
- b. Se imprime por pantalla el mensaje: Son diferentes.
- c. Se imprime por pantalla el mensaje: Son iguales.
- d. Ninguna de las anteriores.

Pregunta 8: Según el texto de la bibliografía básica de la asignatura, ¿cuál de las siguientes afirmaciones **SÍ** es correcta respecto al texto de una clase?

- a. El propósito principal del envoltorio exterior es proporcionar un constructor a la clase.
- b. El envoltorio exterior permite sobrecargar cualquier método de la clase.
- c. El envoltorio exterior contiene la cabecera de la clase.
- d. Ninguna de las anteriores.

Pregunta 9: Para lograr que una clase entre en el depurador en BlueJ a hacer una instancia en BlueJ, ¿qué hay que hacer con el código fuente?:

- a. Compilarlo de nuevo con la opción Debug activado.
- b. Lanzar directamente el depurador.
- c. Meter un punto de ruptura.
- d. Se hace automáticamente al encontrar un error en el código.

Pregunta 10: ¿Qué significa el siguiente fragmento de código Java?: **int uno(int i) { return 1 + i; }**

- a. Hay un método “int uno” que no recibe ningún parámetro de entrada y devuelve el valor 1.
- b. Hay un método “uno” que recibe un parámetro de entrada i y devuelve un entero cuyo valor es $1 + 1$.
- c. Hay una variable “int” cuyo valor es “uno(int i) { return 1 + i; }”
- d. El fragmento no representa un fragmento de código legal en Java.

Pregunta 11: Un método cohesionado ...

- a. Será responsable de al menos una tarea bien definida, pero puede serlo de más.
- b. Es aquel método abstracto que se ha instanciado en una clase determinada.
- c. Es aquel que se crea en una clase interna para ser invocado desde la clase circundante.
- d. Será responsable de una y sólo una tarea bien definida.

Pregunta 12: ¿Cuál de las siguientes sentencias se ejecuta de manera correcta?

- a. String electrodomesticos [] = new String {"Bosh" "Balay" "Siemens"};
- b. String electrodomesticos [] = {"Bosh", "Balay", "Siemens"};
- c. String electrodomesticos = {"Bosh", "Balay", "Siemens"};
- d. String electrodomesticos [] = { "Bosh" "Balay" "Siemens"};

Pregunta 13: Indica cual de las siguientes afirmaciones es correcta:

- a. Los campos también son conocidos como variables de estado.
- b. El alcance de una variable define la sección de código desde donde la variable puede ser declarada.
- c. El tiempo de vida de una variable describe el número de veces que es utilizada en un método.
- d. Los constructores permiten que cada objeto sea preparado adecuadamente cuando es creado.

Pregunta 14: ¿Cuál sería la salida del siguiente código?

```
public class Agenda {  
    public final void metodoAgregarContacto(){  
        System.out.println("Agregar Elemento");  
    }  
}  
  
public class MiAgenda {  
    public static void main(String argv[]){  
        Agenda agenda = new Agenda();  
        agenda.metodoAgregarContacto();  
    }  
}
```

- a. Error en tiempo de ejecución indicando que Agenda no ha sido definida como final.
- b. Error en tiempo de compilación indicando que una clase con métodos finales deben ser declarada también como final.
- c. Error en tiempo de compilación indicando que no se puede heredar de una clase con métodos finales.
- d. Éxito en la compilación y salida “Agregar Elemento”.

RESERVA 1: Indica cual de las siguientes afirmaciones es correcta:

- a. Una aserción es una expresión que establece una condición que esperamos que resulte verdadera.
- b. Un seguimiento es la actividad de trabajar a través de un segmento de código línea por línea, mientras se observan cambios de estado y otros comportamientos de la aplicación.
- c. La prueba es la actividad de descubrir si una pieza de código produce el comportamiento pretendido.
- d. Todas las respuestas anteriores son correctas.

RESERVA 2: Sea el siguiente fragmento de código modificado de la clase MailItem mostrada en el libro de texto:

```
1  public class MailItem {  
2      static int num1 = 5;  
3      public static void main (String args []) {  
4          int num2 = 10;  
5          new MailItem ();  
6      }  
7      public MailItem () {  
8          int aux = this.num1;  
9          if (aux > 1) {  
10              System.out.println(aux);  
11          }  
12      }  
13  }
```

¿Cuál es el resultado que produce?

- a. Se produce un error de compilación.
- b. No se produce ningún error y muestra por pantalla el valor 5.
- c. No produce ningún error pero no muestra nada por pantalla.
- d. No se produce ningún error y muestra por pantalla el valor 10.

PARTE PRÁCTICA [6,5 PUNTOS]:

La Práctica del presente curso va a consistir en diseñar e implementar un sistema integrado de gestión de una tienda de electrodomésticos. Hoy en día las tiendas de electrodomésticos además de disponer de una gran cantidad de productos en sus tiendas, disponen de diferentes perfiles de empleados (técnicos, cajeros, financiación y postventa) para atender de la mejor manera posible a sus clientes. Además, los clientes disponen de un perfil que además de sus datos personales incluyen un historial de compras, generación de facturas, descarga de manuales, comprobación de estado de garantía y promociones. De esta forma, la práctica consiste en desarrollar un sistema de gestión que englobe todas estas características teniendo en cuenta un diseño orientado a objetos.

En general, las funciones que tienen un sistema de gestión de una tienda de electrodomésticos son varias:

- Venta de un electrodoméstico (cajero): cuando un cliente pasa por la línea de cajas es necesario generar una ficha de cliente en el caso de que no disponga de ella. El identificador principal es el DNI y los datos más importantes son el nombre, apellidos, dni, domicilio y número de teléfono. Esta ficha tendrá disponible un histórico de los productos comprados y su fecha de adquisición. En el caso de solicitar financiación, deberá constar en la ficha y el cliente debería pasar por la oficina de financiación para obtener el visto bueno.
- Financiación (financiación): El empleado de la oficina de financiación recibirá clientes que previamente hayan pasado por la línea de cajas para comprar productos y analizará la ficha de financiación. Solicitará la última nómina al cliente, dejando constancia de la cantidad en la ficha del cliente y en caso de que el cargo mensual no supere el 15% de la nómina en un máximo de financiación de 60 meses, la financiación se aprobará.
- Reparación de Electrodomésticos (técnico): Los clientes podrán llevar sus productos comprados en la tienda a reparar. Las condiciones de reparación serán las siguientes: reparación gratuita en los dos primeros años. A partir de esa fecha, se cargará un importe al cliente dependiendo de la reparación efectuada.

- Devolución de electrodoméstico (postventa): Un cliente, presentando su DNI, podrá devolver uno o varios electrodomésticos en el caso de que el periodo de compra no supera los 3 meses.
- Gestión comercial (comercial): Este empleado generará una serie de comunicaciones con el cliente ofreciendo diferentes posibilidades de compra.
- Gestión de usuarios: altas, bajas, modificaciones de las personas que figuran en el sistema (empleados -- técnicos, cajeros, financiación, postventa y comerciales -- y clientes). La primera vez que acude un cliente a la tienda hay que darle de alta en el sistema.
- Gestión de clientes: Cada uno de los empleados tendrán un tipo de relación con el cliente teniendo que dejar constancia en la ficha del cliente la operación realizada y sus detalles (cliente, empleado que atiende, tipo de operación, productos involucrados, fecha, etc.)

Se pide realizar las siguientes tareas:

- a) **[1,0 puntos]** Diseñar utilizando un paradigma orientado a objetos, los elementos necesarios para la aplicación explicada de la práctica durante el curso. Es necesario identificar la estructura y las relaciones de herencia (mediante el uso de un diagrama de clases) y de uso de las clases necesarias para almacenar y gestionar esta información. Debe hacerse uso de los mecanismos de herencia siempre que sea posible. Se valorará un buen diseño que favorezca la reutilización de código y facilite su mantenimiento.
- b) **[1,5 puntos]** Implementar un método (o métodos) para gestionar la devolución de electrodomésticos (postventa): Un cliente, presentando su DNI, podrá devolver uno o varios electrodomésticos en el caso de que el periodo de compra no supera a los 3 meses.
- c) **[1,5 puntos]** Implementar un método (o métodos) para gestionar el plan Renove de Electrodomésticos que ofrece descuentos a los usuarios clientes para la sustitución de frigoríficos, lavadoras o lavavajillas con el etiquetado energético de clase A+++. Dicha oferta va en función del tipo de electrodoméstico: frigoríficos (150€), lavadoras (80€), o lavavajillas (110€). El sistema debería almacenar los datos del electrodoméstico, del cliente y del aparato antiguo.
- d) **[2,5 puntos]** Se quiere extender el negocio de la tienda a la venta en línea a través de una página Web. No es necesario desarrollar el sitio Web. Lo que se pide es el método (o métodos) que gestiona(n) un catálogo para el sitio Web. Además de los datos técnicos de los electrodomésticos, fotos, etc., tendrá que contener datos sobre las opiniones de los clientes de la tienda en línea sobre ellos.