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 2023 – 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: ¿Cuál de las siguientes afirmaciones es correcta?

a. Los campos se conocen también con el nombre genérico de "variables".
b. Los constructores son responsables de garantizar que un objeto se configure adecuadamente al crearlo por vez primera.
c. Una variable de campo dentro de una clase comienza siempre con la palabra reservada "private".
d. Un constructor no recibe parámetros ya que siempre inicializa los valores de los campos con unos valores por defecto.

Pregunta 2: ¿Cuál de las siguientes afirmaciones sobre las variables locales es correcta?

a. El tiempo de vida de una variable local coincide con la duración del objeto al que pertenece.
b. Una variable local puede ser accedida desde cualquier punto de la clase a la que pertenece.
c. La variable local se define dentro del cuerpo de un constructor o un método.
d. De acuerdo a su tiempo de vida, las variables locales actúan como ubicaciones de almacenamiento permanente.

Pregunta 3: En programación orientada a objetos, al proceso de dividir un todo en partes bien definidas que puedan construirse y examinarse por separado y que interaccionen de formas bien definidas se le denomina…

a. Abstracción
b. Polimorfismo
c. Simplificación
d. Modularización

Pregunta 4: ¿Cuál de las siguientes formas de iterar a través de una colección ArrayList es **incorrecta**?

a. Utilizar un bucle for-each sobre los elementos de la colección.
b. Utilizar un bucle while sobre un objeto Iterator.
c. Utilizar un bucle for-each sobre un objeto Iterator.
d. Utilizar un bucle while y un método get con una variable de índice entera.

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

a. El ocultamiento de información es una práctica poco recomendada en la programación orientada a objetos ya que se considera poco ética.
b. El ocultamiento de información consiste en eliminar la documentación de una clase cuando se va a hacer uso de ella en un proyecto diferente.
c. El ocultamiento de información garantiza una mejor modularización de las aplicaciones.
d. Ninguna de las anteriores respuestas es correcta.

Pregunta 6: ¿Qué sucede cuando añadimos a un mapa (por ejemplo, a un HashMap) una entrada cuya clave ya existe en ese mapa?

a. Se mantiene la entrada original y se devuelve un error.
b. Se sobrescribe la entrada anterior que correspondía a esa clave.
c. Ambas entradas coexisten de tal forma que una búsqueda por dicha clave devolverá las dos entradas.
d. Ninguna de las restantes respuestas es correcta.

Pregunta 7: ¿Qué se mostrará por pantalla tras ejecutar el siguiente fragmento de código?

```
public class Prueba {
    public static void main(String[] args) {
        int[] array1 = {3, 2, 1};
        int[] array2 = {4, 5, 6};
        for (int i = 0; i < array1.length; i++) {
            System.out.print(array2[array1[i]-1] + " ");
        }
    }
}
```

a. Se generará una excepción de tipo "ArrayIndexOutOfBoundsException".
b. 4 5 6
c. 1 2 3
d. 6 5 4

Pregunta 8: ¿De qué fenómeno puede ser un indicador el hecho de encontrar demasiadas flechas (dependencias) entre las clases de un diagrama de clases?

a. Alto grado de cohesión.
b. Acoplamiento débil.
c. Acoplamiento fuerte.
d. Alta refactorización.

Pregunta 9: Indica cuál de las siguientes afirmaciones es **falsa**:

a. En un sistema débilmente acoplado podremos cambiar una clase sin efectuar ningún cambio en las clases restantes, y la aplicación seguirá funcionando correctamente.
b. La cohesión es relevante cuando hablamos de métodos individuales, aunque no tanto cuando hablamos de unidades formadas por una sola clase ni a nivel de módulo o paquete.
c. Una adecuada encapsulación de clases reduce el acoplamiento y por tanto conduce a un mejor diseño.
d. En un diseño dirigido por responsabilidad se asignan responsabilidades bien definidas a cada clase, definiendo qué clase debería implementar cada parte de una función de la aplicación.

Pregunta 10: ¿Cuál de las siguientes afirmaciones respecto a la depuración de código es correcta?

a. Un depurador es una herramienta software que proporciona soporte para comprobar el estilo y la validez de un segmento de código.
b. Una ventaja del depurador es que mantiene un registro permanente de cambios de estado, por lo que es sencillo ir hacia atrás y comprobar el estado de unas cuantas instrucciones atrás.
c. El uso de un depurador no se diferencia del análisis manual del código en cuanto a velocidad y susceptibilidad a errores.
d. Ninguna de las restantes respuestas es correcta.

Pregunta 11: Disponemos del siguiente código:

```
public class Prueba {
    public static void main(String[] args) {
        A a = new A();
        B b = new B();
        a.metodo();
        b.metodo();
        System.out.println(a.x + ";" + b.x);
    }
}

class A {
    int x = 5;
    public void metodo() {
        System.out.print(x++ + ";");
    }
}

class B extends A {
    int x = 10;
    public void metodo() {
        System.out.print(x++ + ";");
    }
}
```

¿Qué se mostrará por pantalla tras compilar y ejecutar dicho código?

a. 5;10;6;11
b. 5;10;5;10
c. 6;11;6;11
d. 6;11;5;10

Pregunta 12: Disponemos del siguiente código:

```
class A {
    public void metodo() {
        System.out.print("A");
    }
}

class B extends A {
    public void metodo() {
        System.out.print("B");
    }
}

class C extends B {
    public void metodo() {
        super.metodo();
        System.out.print("C");
    }
}

public class ABC {
    public static void main(String[] args) {
        A a = new A();
        B b = new B();
        C c = new C();
        a.metodo();
        b.metodo();
        c.metodo();
    }
}
```

¿Qué se mostrará por pantalla tras compilar y ejecutar dicho código?

a. ABBC
b. AABC
c. ABCC
d. Ninguna de las restantes respuestas es correcta.

Pregunta 13: ¿Cuál es el objetivo principal de una clase abstracta?

a. La creación de instancias de dicha clase.
b. La generación de interfaces a partir de la clase.
c. Servir como una superclase para otras clases.
d. Implementar todos los métodos de la clase de la que hereda.

Pregunta 14: ¿Cuál de las siguientes afirmaciones es correcta?

a. Una excepción comprobada requiere el uso de cláusulas “throws” e instrucciones “try”.
b. Una excepción no comprobada requiere el uso de cláusulas “throws”.
c. Una excepción comprobada hereda de la clase “RuntimeException”.
d. Ninguna de las restantes respuestas es correcta.

Pregunta R1: ¿Cuál es el resultado de compilar y ejecutar el siguiente código?

```
public class Examen {
      static int num = 10;
      public static void main (String args []) {
            int numero = 5;
            new Examen ();
      }

      public Examen () {
           int aux = this.numero;
           if (aux > 1) {
                System.out.println(aux);
           }
      }
}
```

a. Se muestra por pantalla el número 5.
b. Se muestra por pantalla el número 10.
c. Se produce un error de compilación.
d. Se produce un error de ejecución.

Pregunta R2: ¿Qué se mostrará por pantalla si compilamos y ejecutamos el siguiente código?

```
class Animal {
      void caminar() {
            System.out.println("Voy caminando");
      }
}

class Perro extends Animal {
      void caminar() {
            System.out.println("Voy caminando a cuatro patas");
      }

      void ladrar() {
            System.out.println("¡Guau!");
      }
}

public class HerenciaAnimal {
      public static void main(String[] args) {
            Animal animal = new Perro();
            animal.caminar();
            ((Perro) animal).ladrar();
      }
}
```

a. Voy caminando a cuatro patas
   ¡Guau!
b. Voy caminando
   ¡Guau!
c. Se produce un error de compilación.
d. Se produce un error de ejecución.

PARTE PRÁCTICA [6,5 PUNTOS]:

La práctica del presente curso consiste en implementar un sistema integrado de gestión de una cooperativa agrícola que contemple la realidad del entorno en el que se mueve. Los actores que participan del funcionamiento de la aplicación son los siguientes:

- Productores: Desde un punto de vista estricto, los productores van a ser cualquier actor que proporciona a la empresa un determinado producto, y que cobrará por el mismo el precio que se estipule. Dentro de los productores, podemos encontrar dos tipos: pequeños productores, que son aquellos que tienen una extensión inferior o igual al límite en hectáreas que se determine en cada año fiscal (5 hectáreas) y los grandes productores, que son aquellos que superan esa cantidad. Los productores proporcionan a la cooperativa una diversidad de productos. Esta cantidad de productos está relacionada con el hecho de ser pequeño productor (pueden como máximo cinco productos diferentes) o gran productor (pueden proporcionar un número no limitado de productos). Existen también lo que se consideran "productores federados", que son aquellos productores que por la extensión total no superan el límite de los pequeños productores, pero que uno de sus productos lo "cede" a una federación de pequeños productores que se dedica a un producto determinado, de modo que esa agrupación de n pequeños productores son también (en su conjunto) un productor federado. No puede haber más de un productor federados por cada producto. El número total de pequeños productores que forma un productor federado no está limitado, si bien se exige que la suma total de extensiones no supere el límite de extensión existente para ser considerado pequeño productor (es decir, no hay un productor federado sobre un producto que tenga una extensión superior a 5 ha).
  Por otro lado, cada producto (por ejemplo, algodón) tiene lo que se denomina un "rendimiento por hectárea", en el que se indica cuántas toneladas se obtienen por hectárea de producto (por ejemplo, 2,5 toneladas de algodón por cada hectárea). Además, cada producto tiene un valor de referencia por kilogramo libre de impuestos (sin IVA). Por ejemplo, el precio del algodón podría fijarse en torno a los 80 céntimos de euro por kilogramo. **Nota: una tonelada son 1000 kg**.

- Logística: Este grupo de actores tienen como función principal la de proporcionar a los clientes finales y distribuidores los productos disponibles en la cooperativa. La logística se divide principalmente en dos grandes grupos: productos perecederos y productos no perecederos. Así, los productores de productos perecederos requieren de una logística más "costosa" (debe mantenerse unas condiciones de transporte determinadas) y el coste en sí depende tanto del producto como de la distancia. Esto se debe a que si la distancia de transporte del producto es inferior a 100 km, entonces la propia empresa de gran logística realiza todo el proceso. Si la distancia es superior a los 100 km, entonces hay que emplear una doble logística: una "gran logística" que implica el envío del producto a la capital de la provincia donde se va a surtir el producto, y después una "pequeña logística" realiza el reparto desde la capital de la provincia hasta el destino final (se asegura que esta pequeña logística siempre realiza trayectos inferiores a los 100 km). Respecto a los productos no perecederos, la filosofía es la misma que en el caso anterior con la particularidad de

que la "gran logística" requiere la contratación de diferentes tramos múltiplos de 50 km. Es decir, enviar un producto a 140 kilómetros de distancia de la cooperativa implicará la contratación de dos tramos de 50 km de "gran logística" y un tramo final de distribución de 40 km de "pequeña logística".
El coste de la gran logística está relacionado con el coste del producto, número de trayectos y kilómetros. Así, cada trayecto tiene un precio fijo de 0,5 * Precio de Referencia del Producto Libre de Impuestos * Kilogramos contratados. Cada kilómetro se pagará a precio fijo. Como máximo cada trayecto contratado puede gestionar un máximo de 1 Tonelada de producto.
El coste de la pequeña logística va relacionado simplemente con el kilometraje recorrido.
Es importante tener en cuenta (por las implicaciones en lo que al diseño se refiere) que desde el punto de vista de la logística, pueden existir (y de hecho, deben existir) diferentes ofertas para poder surtir los productos tanto a los distribuidores como a los consumidores finales (lo que se traducirá en costes diferentes en la logística).

- Distribuidores y Consumidores Finales: Los últimos actores son los distribuidores y consumidores finales. Los primeros se dedican a comprar grandes cantidades para posteriormente ponerlos a la venta a unos clientes finales. Los segundos adquieren productos a la cooperativa directamente.
  Desde el punto de vista de los distribuidores, excluyendo la logística, la cooperativa vende sus productos añadiendo un 5% al precio de referencia libre de impuestos del producto. Un distribuidor está obligado a comprar al menos 1 tonelada del producto que sea, y deberá pagar la logística de la forma en la que se ha indicado en el apartado anterior.
  Desde el punto de vista de los consumidores finales, excluyendo la logística, la cooperativa vende sus productos añadiendo un 15% al precio de referencia libre de impuestos del producto. Un consumidor final no puede adquirir más de 100 kg de producto, y deberá pagar la logística de la misma forma en la que lo hacen las distribuidoras, añadiendo el correspondiente IVA (que por simplicidad, supondremos siempre que es el 10%). El IVA se aplica tanto al producto en sí (que vende la cooperativa) como al precio de la logística.

Además, se plantean las siguientes consideraciones específicas:

- Las compras a la cooperativa se realizan en una determinada fecha y se sirven dentro de un plazo máximo de diez días. Si al realizar la petición, se solicita que se entregue en un plazo superior a diez días, entonces habrá que revisar el valor del producto en el momento de proporcionarlo (siempre diez días antes de la fecha de entrega solicitada).
- Los precios de los productos se revisan semanalmente, y pueden experimentar subidas o bajadas.
- Cada pedido lleva asociado un número de pedido.
- Se lleva un registro de pedidos por cada distribuidor y/o consumidor final.
- El importe que recibe cada productor (sea del tipo que sea) será proporcional a lo que se genera en cada pedido. Así, si se realiza un pedido de algodón de 300 kg, y hay tres productores de 1 ha, 3 ha y 0,5 ha, cada uno de los productores recibe la

parte proporcional a lo que aportan (es decir, sería como si le hubiese comprado 66,67 kg al primero, 200 kg al segundo y 33,33 kg al tercero).

- No se puede vender más cantidad de producto del existente en la cooperativa.
- Importante tener en cuenta que los productores reciben el importe establecido como precio de referencia libre de impuestos (que es diferente al precio que el distribuidor o consumidor final compra, puesto que hay un porcentaje adicional que se añade y que revierte en la cooperativa).

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 de todos los productos con los que trabaja la cooperativa agrícola, indicando para cada uno de ellos los productores que los trabajan, de qué tipo son (pequeño, grande y productor federado) y la cantidad total de hectáreas que manejan de dicho producto.

c) **[2,0 puntos]** Implementar un método (o métodos) que permita calcular el coste total de todos los pedidos servidos por la cooperativa en un intervalo de tiempo determinado. Para ello, se deberá introducir la fecha de inicio y de fin de dicho intervalo, y se considerarán únicamente los pedidos que se hayan servido dentro del intervalo especificado.

d) [**1,5 puntos]** Implementar un método (o métodos) que permita gestionar las siguientes estadísticas relacionadas con el trabajo de la cooperativa agrícola:
   - Los 5 distribuidores cuyos pedidos suman un mayor coste total.
   - Los 5 productos más solicitados a la cooperativa.
   - Los 5 consumidores finales que más pedidos han hecho a la cooperativa (aunque su coste total no sea el más elevado).