EJERCICIOS T. 8. Clases y objetos en Java

1. Desarrolle la clase *Producto* con tres atributos: *nombre*, *precio* y *referencia* del producto, con el nombre del producto, el precio, y un entero que sirva como número de referencia del mismo. Para lograr la encapsulación, dichos atributos han de ser privados. Implemente, pues, los métodos que permitan consultar el valor de dichos atributos desde una clase externa. La clase tendrá dos constructores, que serán el constructor por defecto, que asignará valores "vacíos" a los atributos, y otro que recibirá un valor para cada uno de dichos atributos, y se lo asignará convenientemente.

Desarrolle ahora la clase *Indice*, con dos atributos: uno será un vector de Producto, y otro será una cadena indicando el criterio por el que serán ordenados dichos productos en el vector. Los criterios posibles serán el precio del producto, y su número de referencia. La clase Indice tendrá los siguientes métodos y constructores:

- Constructor que reciba un vector de productos, y una cadena que indique el criterio por el que deberán estar ordenados. Dicho constructor se encargará de asignar al atributo correspondiente el criterio de ordenación, así como el vector de productos recibido, pero ordenados según el criterio indicado.
- Métodos de consulta del valor de los atributos albergados, puesto que éstos serán privados, para lograr la encapsulación.
- Método *ordenar*, que recibirá una cadena indicando un criterio de ordenación válido, y se encargará de actualizar los atributos del objeto para ser coherente con la nueva ordenación de los productos.
- Método *insertar*, que recibirá un nuevo producto, y lo insertará en el atributo correspondiente, haciendo que éste siga ordenado según el criterio que haya especificado.
- Método *buscar*, que recibirá un nombre de producto, y devolverá un vector de productos con todos aquellos que tengan dicho nombre.

Implemente una clase principal, llamada *Principal* o *Almacen*, que pruebe las clases desarrolladas anteriormente. Para ello realizará la siguiente secuencia de operaciones:

- Creará un nuevo vector con cinco productos. Dichos productos habrán sido instanciados previamente con los valores que usted quiera.
- Instanciará un objeto de la clase *Indice*, con el vector de productos creado antes, y con el criterio de ordenación correspondiente al número de referencia.
- Comprobará que efectivamente, los productos albergados ahora en *Indice* están ordenados por el criterio del número de referencia, mostrándolos por pantalla y viendo que efectivamente siguen dicho orden.
- Ordenará los productos del objeto de la clase *Indice* por el otro criterio de ordenación (precio del producto), haciendo uso del método pertinente.
- Comprobará de nuevo que la ordenación es correcta, según el nuevo criterio, mostrando los productos por pantalla.
- Insertará un nuevo elemento en el objeto creado de la clase *Indice* por medio del método correspondiente.
- Comprobará que el nuevo elemento ha sido insertado correctamente en la posición que le corresponde según el criterio de ordenación que había seleccionado.
- Se buscarán y mostrarán los elementos que tienen un nombre determinado, haciendo uso del método adecuado.

2. Implemente una clase *Animal* con el método comer, que imprime "Estoy comiendo", y los atributos *edad* y *peso*. Cree también las clases *Vaca* y *Tiburon* como especializaciones de *Animal*, de modo que cuando invoquemos a su método *comer* impriman "Estoy comiendo hierba", y "Estoy comiendo calamar", respectivamente.
La clase *Vaca* tendrá también el atributo *color*, iniciado por defecto con "blanco y negro". Cree una clase *Principal* que instancie un objeto de cada tipo, y visualice todos sus atributos, los cuales habrán de ser privados, para conseguir una correcta encapsulación.

3. Desarrolle una clase llamada *CuentaCorriente* que:

-Tenga los atributos nombre, apellidos, edad (del titular, en los tres casos; opcionalmente puede crear la clase Titular para gestionar estos tres atributos), número de cuenta (String) y saldo.
-Tenga un constructor con los atributos anteriores.
-Tenga un constructor con los atributos anteriores, excepto el saldo, que se iniciará a 15.3.
-Tenga un método de recuperación de cada uno de los atributos, para conseguir que estos puedan ser privados (encapsulación).
-Tenga un método de asignación de un valor para el saldo.
-Tenga un método ingresar que incremente el saldo en una cantidad.
-Tenga un método reintegro que decremente el saldo en una cantidad.
-Tenga un método para imprimir la cuenta, de modo que salga por pantalla el número de cuenta y el saldo.
-Tenga un método para comparar cuentas, sabiendo que dos cuentas serán iguales si sus números de cuenta son iguales.

Desarrolle una clase llamada *CuentaAhorro* que:

- Sea una especialización de CuentaCorriente.
- Tenga un atributo adicional con el interés aplicado a la cuenta.
- Tenga un constructor con parámetros que incluyan al titular de la cuenta, y el resto de campos de la clase.
- Tenga un constructor con parámetros que incluyan todos los de la clase, exceptuando el saldo, que se iniciará a 15.3.
- Tenga un constructor con parámetros que incluyan todos los de la clase, exceptuando el saldo, que se iniciará a 15.3, y el interés, iniciado a 2.5.
- Tenga un método de recuperación de cada uno de los atributos.
- Tenga un método llamado calcularInteres, que incremente el saldo según el tipo de interés.

Desarrolle una clase Principal que en su método main cree varias cuentas de distinto tipo, y trabaje con ellas.

4. Haga uso de los conceptos que ya conoce de polimorfismo para resolver la siguiente práctica con el mejor diseño posible.

Desarrolle una aplicación de control de llamadas realizadas en una centralita telefónica. En dicha centralita se van registrando las llamadas. Registrar una llamada consiste en contar el número de llamadas realizadas, así como el coste total de todas las llamadas realizadas. La centralita mostrará por pantalla todas las llamadas según las vaya registrando.

Existen dos tipos de llamadas:

- Las llamadas locales, que cuestan 15 céntimos el minuto.
- Las llamadas provinciales, que cuestan 20 céntimos por minuto si se realizan en la franja horaria 1, 25 si se realizan en la franja 2, y 30 céntimos si son realizadas en la franja 3.

Todas las llamadas tienen como datos el número de origen de la llamada, el número de destino, y la duración en segundos.

Desarrolle la clase Principal, que en su método main cree una centralita, registre varias llamadas de distinto tipo, y le pida a la centralita un informe total de llamadas y la facturación total realizada.

5. Haga uso de los conceptos que ya conoce de polimorfismo para resolver la siguiente práctica con el mejor diseño posible.

Desarrolle una aplicación de encriptación de códigos numéricos. El encriptador recibirá un código numérico, y una implementación de un algoritmo de encriptación / desencriptación, y la orden de encriptar o desencriptar el código.

Existen distintos algoritmos de encriptación / desencriptación:

- Multiplicación por un número determinado
- Diferencia existente con un número determinado
- Operación XOR con un número determinado

Desarrolle la clase Principal, que en su método main cree un encriptador, y le pida encriptar y desencriptar varios códigos, usando distintos algoritmos de encriptación.

6. Realice los cambios que estime oportunos en el ejercicio 4 (centralita telefónica) para aplicar sobre él el concepto de clases abstractas o de interfaces, según lo que estime oportuno.

7. Realice los cambios que estime oportunos en el ejercicio 5 (algoritmos de encriptación) para aplicar sobre él el concepto de clases abstractas o de interfaces, según lo que estime oportuno.

8. Reimplemente el diseño correspondiente a las clases Figura, Cuadrado, Rectángulo, Triángulo y Circunferencia, de modo que todas las figuras finales tengan los atributos color y posición (valor X e Y), y además aquellos atributos que sean característicos de sus dimensiones. Cada figura final habrá de tener, además, los métodos correspondientes para calcular el área y el perímetro de la misma.
Realice un diseño en que haga uso de herencia y clases y / o métodos abstractos y / o interfaces, según estime oportuno, justificando su elección.

9. Todo centro de estudios ha de tener como mínimo un número de pisos, un número de aulas, y un número de despachos. Dicho valor mínimo es un valor constante para todos los centros, marcado por la legislación. Desarrolle una interfaz para almacenar dichos valores, y que además tenga la plantilla necesaria para determinar el número de aprobados a partir de un array de notas (cada nota es un float), el número de suspensos a partir de un array de notas, y la nota media a partir de un array de notas.

Desarrolle también una clase *Instituto* que implemente la interfaz anterior, y pruebe su correcto funcionamiento.

10. Escriba diferentes programas de prueba que le permitan practicar con los diferentes aspectos de clases y objetos en Java.