Ejercicios UT5 –Desarrollo de clases

1. Una empresa informática necesita llevar un registro de todos sus empleados que se encuentran en la oficina central, para ello, se debe crear una clase Empleado que incluya lo siguiente:

   Atributos: nombre completo, permanencia (número de años en la empresa) y salario.

   Métodos:

   A. Constructor con y sin parámetros de entrada.

   B. Método que permita mostrar la clasificación de acuerdo con:

      a. Si la permanencia es menor o igual a 3, “Principiante”.
      b. Si la permanencia es mayor que 3 y menor que 18, “Intermedio”.
      c. Si la permanencia es mayor o igual a 18, “Sénior”.

   C. Método que muestre los datos del empleado por pantalla, incluida la permanencia (se debe utilizar salto de línea para separar los atributos).

   D. Un método que permita aumentar el salario en un porcentaje que sería pasado como parámetro al método.

   Implemente también una clase Main para probar la clase Empleado.

2. Desarrolle un programa que permite gestionar los cobros de las llamadas de móvil de sus usuarios. Para ello, la compañía guarda la siguiente información de cada teléfono móvil, por lo que se creará una clase denominada Móvil con los atributos: Número de teléfono, tarifa y consumo en minutos. Existen tres tarifas posibles: Elefante (coste a 0,30€ minuto), Tigre (0,18€ minuto) y Gato (0,07€ minuto).

   Las funcionalidades que tendrá la clase Móvil son las siguientes: Llamar, de manera que cuando se realiza una llamada se ajustará el consumo total del móvil. Cada llamada tendrá una duración en segundos. Reiniciar la factura, que consiste en poner a cero el consumo total. Resumen de la factura, en la que se indicará el número, la tarifa y el consumo total.

   Desde el programa principal se crearán varios móviles y se realizarán llamadas con los mismos, se mostrará la factura y se reiniciarán tarifas. Para ello se creará un menú principal que recoja las opciones disponibles con los móviles creados.

3. Se trata de crear un programa que simule una empresa de envío de paquetes. Dicha empresa tiene varias sucursales y puede enviar paquetes con diferente peso y diferente prioridad. El programa contiene tres clases:

   Clase Sucursal, cuyos atributos son: Número de sucursal, Dirección y Ciudad. Tendrá un método constructor encargado de establecer la dirección y la ciudad, siendo el número de sucursal el número que le corresponda de una secuencia común a todas las sucursales. La clase tendrá los métodos getters para la dirección y la ciudad. Además habrá un método setter que calculará el precio. Este método recibe por parámetro un objeto de tipo Paquete y calcula el precio del envío del paquete en función del peso

del mismo y la prioridad del envío (0, 1 o 2 siendo estos 0=normal, 1=Alta y 2=Urgente). Prioridad 0 el precio no se altera. Prioridad 1 el precio se incremente 10 €. Prioridad 2 el precio se incrementa 20 €.

Clase Paquete, Encargada de construir objetos de tipo Paquete con los atributos: Referencia del envío, Peso, DNI del remitente, Prioridad del envío. Esta puede ser 0, 1 y 2 (Normal, Alta, Urgente). La clase tendrá un Constructor encargado de establecer las propiedades a los valores elegidos. Además tendrá 2 setters para modificar (si se quiere) peso y prioridad y 2 getters para obtener peso y prioridad.

Clase Main Esta es la clase principal. En ella habrá un menú de opciones que permita dar de alta la sucursal y enviar paquetes.

4. Queremos gestionar la venta de entradas (no numeradas) de una feria de vehículos de segunda mano que tiene 3 zonas, la sala principal con 1000 entradas disponibles, la zona de compra-venta con 200 entradas disponibles y la zona vip con 25 entradas disponibles. Hay que controlar que existen entradas antes de venderlas.

   La clase Zona con sus atributos se muestra a continuación:

```
public class Zona {

  private int entradasPorVender;

  public Zona(int n){
    entradasPorVender = n;
  }

  public int getEntradasPorVender() {
    return entradasPorVender;
  }
```

   Método que vende un número de entradas. Comprueba si quedan entradas libres antes de realizar la venta. El método recibe como parámetro el número de entradas a vender

   El menú del programa debe ser el que se muestra a continuación. Cuando elegimos la opción 2, nos debe preguntar para qué zona queremos las entradas y cuántas queremos. Lógicamente, el programa debe controlar que no se puedan vender más entradas de la cuenta.

   1. Mostrar número de entradas libres
   2. Vender entradas
   3. Salir

5. La empresa LibreCoders te ha contratado para desarrollar un software de gestión de una cuenta bancaria para la cooperativa de banca BdeBanco. Se trata de una aplicación Java formada por una clase principal BdeBanco y otra llamada CuentaBancaria.

El programa pedirá los datos necesarios para crear una cuenta bancaria. Si son válidos, creará la cuenta y mostrará el menú principal para permitir actuar sobre la cuenta. Tras cada acción se volverá a mostrar el menú.

1. Datos de la cuenta. Mostrará el IBAN, el titular y el saldo.
2. IBAN. Mostrará el IBAN.
3. Titular. Mostrará el titular.
4. Saldo. Mostrará el saldo disponible.
5. Ingreso. Pedirá la cantidad a ingresar y realizará el ingreso si es posible.
6. Retirada. Pedirá la cantidad a retirar y realizará la retirada si es posible.
7. Movimientos. Mostrará una lista con el historial de movimientos.
8. Salir. Termina el programa.

Clase CuentaBancaria: Una cuenta bancaria tiene como datos asociados el iban (international bank acount number, formado por dos letras y 22 números, por ejemplo ES6621000418401234567891), el titular (un nombre completo), el saldo (dinero en euros) y los movimientos (histórico de los movimientos realizados en la cuenta, un máximo de 5 para simplificar).

Cuando se crea una cuenta es obligatorio que tenga un iban y un titular (que no podrán cambiar nunca). El saldo será de 0 euros y la cuenta no tendrá movimientos asociados. El saldo solo puede variar cuando se produce un ingreso (entra dinero en la cuenta) o una retirada (sale dinero de la cuenta). En ambos casos se deberá registrar la operación en los movimientos. Los ingresos y retiradas solo pueden ser de valores superiores a cero.

El saldo de una cuenta nunca podrá ser inferior a -50 euros. Si se produce un movimiento que deje la cuenta con un saldo negativo (no inferior a -50) habrá que mostrar el mensaje “AVISO: Saldo negativo”. Si se produce un movimiento superior a 3.000 euros se mostrará el mensaje “AVISO: Notificar a hacienda”.

Clase BdeBanco: Clase principal con función main. Encargada de interactuar con el usuario, mostrar el menú principal, dar feedback y/o mensajes de error, etc. Utilizará la clase CuentaBancaria. Puede implementar los métodos que considere.

6. **El Tren.** Se necesita crear un programa para poder gestionar la venta de tickets de tren. Por teclado, el cliente introducirá los siguientes datos (izq) y la salida será (dcha):

```
Indica ciudad origen:
Madrid
Indica ciudad destino:
Granada
Indica la fecha (dd/mm/aaaa):
01/03/2024
¿Quieres ventana? (si/no):
si
¿Quieres ir en sentido de la marcha? (si/no):
si
Indica la fila del asiento:
7
```

```
Origen: Madrid          Hora de salida: 10:00
Destino: Granada        Hora de llegada: 11:15
-------------------------------------------------
Fecha: 01/03/2024       Precio con IVA: 26.015
-------------------------------------------------
Asiento Fila: 7         Posición: A
```

Clase Asientos: Donde habrá sobrecarga de constructores. Un constructor tendrá solo la ventana y sentido de la marcha (habrá que tener en cuenta que tenemos que calcular la posición del asiento dependiendo las opciones escogidas para ventana y sentido marcha: Asientos A= sentido de la marcha Ventana. Asientos B= sentido de la marcha pasillo. Asientos C= sentido de la Contramarcha Ventana. Asientos D= sentido de la Contramarcha pasillo. Constructor con todos los atributos menos posición que se calcula dependiendo de otros parámetros.

Se tendrán que hacer los getter y setter correspondientes y necesarios.

Se tendrán que hacer (si es necesario) los métodos necesarios.

Clase Billetes: Con los atributos que consideres necesarios para obtener el resultado mostrado anteriormente y un método donde se calcula el precio con IVA.

Clase Main: Se introducirá por teclado el billete deseado y se crearán los objetos necesarios para que el sistema funcione correctamente.

7. **Fechas:** Crea una clase Fecha. En la que los atributos que debe tener son: día, mes y año, numéricos enteros. Cree dos constructores: El constructor por defecto (puede estar vacío) y otro en el que asignaremos a cada atributo los valores de día, mes y año que pasemos al instanciar el objeto. Programe los siguientes métodos:

   - Método para saber si un año es bisiesto. A partir del año que tenemos en el atributo año del objeto que hayamos creado, comprobar si este es bisiesto o no. Nos devolverá true ó false según el caso. Recuerde que un año sea bisiesto si es múltiplo de 4 y no múltiplo de 100 ó si es múltiplo de 400.

   - Método para que, dadas dos fechas nos devuelva el tiempo transcurrido entre ellas. En este caso trabajaremos con la fecha del objeto que utilicemos y otra nueva introducida por teclado. Los parámetros que enviaremos al invocar el método serán el día, mes y año de esta última.Debéis validar la fecha en el método principal y tener en cuenta que tenemos meses de 30 días. El método nos devolverá los días transcurridos entre ellas.

   - Método que nos devuelva la fecha como una cadena con el formato DD/MM/AAAA

   - Método que permita comparar dos fechas.

   - Método toString para mostrar las fechas con el formato DD/MM/AAAA.

   En el programa principal (clase con el método main) instanciaremos dos objetos uno con nuestra fecha de nacimiento (miFechaNacimiento) y otro con el día, mes, y año que se haya introducido por teclado fechaReferencia). En este último caso validar los datos introducidos antes de crear el objeto, de tal manera que trabajemos con fechas válidas.

Ejercicios básicos de Recursividad

8. Crea un método que imprima los dígitos desde 1 hasta N. Se debe pasar como parámetro el número N
9. Crea un método que obtenga la suma de los números naturales desde 1 hasta N. Se debe pasar como parámetro el número N
10. Crea un método que obtenga el factorial de un número N. Se debe pasar como parámetro el número N
11. Crea un método que calcule el número de fibonacci a partir de un número pasado como parámetro
12. Crear un método que obtenga el resultado de elevar un número a otro. Ambos números se deben pasar como parámetros
13. Crea un método que dado un número, lo imprima invertido por pantalla
14. Crea un método que imprima por pantalla un Rectángulo a partir de los valores de la base y la altura
15. Crea un método que imprima por pantalla un Triángulo rectángulo a partir del valor de la altura del triángulo
16. Crea un método que compruebe si una palabra está ordenada alfabéticamente
17. Crea un método que compruebe si una palabra es un palíndromo
18. Crea un método que compruebe si un número es binario. Un número binario está formado únicamente por ceros y unos
19. Crea un método que obtenga el número binario de un número N pasado como parámetro.
20. Crea un método que compruebe si un número está ordenado de forma decreciente y creciente
21. Crea un método que compruebe si un número es simétrico. Los números simétricos son iguales a partir del dígito central pero comparando en dirección opuesta.
22. Genere un programa recursivo que indique si una cadena posee paréntesis balanceados.