**Definición de Clase y Objetos**

1. Queremos modelar una casa con muchas bombillas, de forma que cada bombilla se puede encender o apagar individualmente. Para ello:
   1. Haremos una clase Bombilla con una variable privada que indique si está encendida o apagada, así como un método que nos diga si la bombilla está encendida.
   2. Además queremos poner un interruptor general de la luz, tal que si saltan los fusibles, todas las bombillas quedan apagadas. Cuando el fusible se repara, las bombillas vuelven a estar encendidas o apagadas, según estuvieran antes del percance. Para modelar en Java esta característica usaremos una variable de clase (static) que nos diga si hay luz en el edificio o no. Cada objeto Bombilla se enciende y se apaga individualmente; pero sólo responde que está encendida si su interruptor particular está activado y además hay luz general.

La clase tendrá dos atributos:

* Un interruptor general que indica si los fusibles están activados o desactivados. Este atributo es compartido por todos los objetos de la clase.
* Un interruptor particular que indica si la bombilla esté encendida o apagada.

Los métodos de la clase serán:

* Un constructor sin parámetros
* Un método que active los fusibles
* Un método que desactive los fusibles
* Un método que encienda la bombilla
* Un método que devuelva verdadero si la bombilla está encendida o falso en caso contrario.

Diseña la clase bombilla y realiza un programa de prueba. El programa creará tres bombillas: una para el dormitorio, otra para el salón y otra para la cocina.

* El programa mostrará u menú con las siguientes opciones:
* Encender habitación
* Encender Salón
* Encender cocina
* Apagar habitación
* Apagar Salón
* Apagar cocina
* Activar interruptor general
* Desactivar interruptor general
* Consultar estado de las bombillas
* Salir

Pedirá una de las opciones y ejecutará la opción indicada. Realiza las siguientes pruebas:

* Enciende la cocina y el salón.
* Comprueba el estado de las bombillas
* Activa el interruptor general
* Comprueba el estado de las bombillas

1. Desarrollar una clase Empleado cuyos atributos sean:
   * Una empresa, el nombre de la empresa a la que pertenece el empleado. El nombre de la empresa es inmutable, una vez que se inicializa no se puede cambiar.
   * Un nombre, apellidos y nombre del empleado.
   * Un departamento, con el numero del departamento al que pertenece el empleado. Los departamento pueden ser: 1 para Contabilidad, 2 para Producción o 3 para Ventas.
   * Un sueldo, sueldo del empleado que debe ser mayor de 500 y menor o igual que 2000.
   * Un número de empleado (numEmpleado), también es inmutable y no puede ser negativo.
   * Horas extra: número entero mayor o igual que 0 y menor o igual a 20 que representa el número de horas extras que el empleado ha realizado y aún no ha cobrado.

Estos datos pueden ser accedidos por las clases que hereden de empleado y por todas las clases que se encuentren en el mismo paquete.

Para esta clase se piden los siguientes constructores y métodos:

* Un constructor con el nombre de la empresa, el número de empleado y el nombre.
* Un constructor con todos los parámetros.
* Los métodos **get** que permitan consultar el valor de sus atributos.
* Los métodos **set** que permitan modificar sus atributos.
* Método **toString(),** devuelve todos los datos del empleado.
* Método **calcularNomina(),** devuelve el monto de la nómina del empleado. La nómina se calcula con la fórmula **sueldo+horasExtra\*30**.
* Método **subirSueldo()**, recibe un número entero que representa el porcentaje de subida y sube el sueldo del empleado en ese porcentaje.
* Método **pagarNomina()**, paga la nómina a los empleados. El método dejará a 0 el número de horas extra.

Crea un método de prueba que declare un array de 5 empleados y guarde cinco objetos de tipo empleado. Los dos primeros objetos se crearán con el primer constructor y los tres siguientes con el segundo constructor.

Después se pedirán por teclado los datos que faltan de los dos primeros empleados, para dejar completos los atributos.

Después se recorrerá el array y se mostrarán los datos de cada uno.

Después se consultará el nombre, la empresa y el sueldo de cada empleado.

Recorre el array y todos aquellos empleados que cobren menos de 700 euros se les subirá el sueldo un 10%.

Recorre el array y guarda el número de horas de cada empleado.

Recorre el array y primero se mostrará el monto de la nómina de cada empleado y a continuación se pagará la nómina.

1. Desarrollar una clase Fecha cuyos atributos sean: dia, mes y año.
   * Un constructor con todos sus parámetros
   * Un método privado que comprueba que la fecha sea correcta. Si la fecha no es correcta volverá a pedir los datos y devolverá un objeto Fecha correcto.
   * Métodos set y get de cada atriburo
   * Método toString que devuelva la fecha en el formato dd/mm/aaaa

Añade a la clase Empleado un atributo llamado FechaNacimiento de tipo Fecha y los método get y set necesarios.

1. Crea una clase Cuenta (bancaria) con atributos para el número de cuenta (un entero largo), el DNI del cliente (otro entero largo), el saldo actual y el interés anual que se aplica a la cuenta (porcentaje). Define en la clase los siguientes métodos:

Constructor por defecto y constructor con DNI, saldo e interés

* Métodos **set y get** de los atributos. El número de cuenta tendrá un método **get** pero no tendrá método **set**.
* **actualizarSaldo():** actualizará el saldo de la cuenta aplicándole el interés diario (interés anual dividido entre 365 aplicado al saldo actual).
* **ingresar(double):** permitirá ingresar una cantidad en la cuenta.
* **retirar(double):** permitirá sacar una cantidad de la cuenta (si hay saldo).
* Método que nos permita mostrar todos los datos de la cuenta.

El número de cuenta se asignará de forma correlativa a partir de 100001, asignando el siguiente número al último asignado.

1. Desarrolla una clase Cancion con los siguientes atributos:

* **titulo**: una variable String que guarda el título de la canción.
* **autor**: una variable String que guarda el autor de la canción.

y los siguientes métodos:

* **Cancion(String, String):** constructor que recibe como parámetros el título y el autor de la canción (por este orden).
* **Cancion():** constructor predeterminado que inicializa el título y el autor a cadenas vacías.
* **dameTitulo():** devuelve el título de la canción.
* **dameAutor():** devuelve el autor de la canción.
* **ponTitulo(String):** establece el título de la canción.
* **ponAutor(String):** establece el autor de la canción.

A continuación crea una clase Prueba que declare un array de tipo **Cancion** y la inicialice con objetos creados con el constructor por defecto. A continuación el programa mostrará un menú con las siguientes opciones:

* **Almacenar canción:** pedirá por teclado los datos de una canción y la almacenará en la primera posición vacía del array.
* **Mostrar discografía:** mostrará todas las canciones almacenadas en el array en la forma Titulo Autor, cada una en una línea. Al lado de cada una mostrará la posición que ocupa en el array.
* **Eliminar canción:** pedirá la posición del array de la que se quiere eliminar la canción y esa posición la pondrá a null.
* **Número de canciones:** devuelve el numero de canciones que hay almacenados en el array.

1. Vamos a crear una clase llamada Persona que siga las siguientes condiciones:

**Sus atributos son:** nombre, edad, DNI, sexo (H hombre, M mujer), peso y altura. No queremos que se accedan directamente a ellos desde ninguna otra clase. Piensa que modificador de acceso es el más adecuado, también su tipo.

Por defecto, serán valores por defecto según su tipo (0 números, cadena vacía para String, etc.). Sexo será hombre por defecto.

Se implantaran varios constructores:

* Un constructor por defecto.
* Un constructor con el nombre, edad y sexo, el resto por defecto.
* Un constructor con todos los atributos como parámetro.

Los métodos que se implementaran son:

* **calcularIMC():** calculara si la persona está en su peso ideal (peso en kg/(altura^2 en m)), si esta fórmula devuelve un valor menor que 20, la función devuelve un -1, si devuelve un número entre 20 y 25 (incluidos), significa que está por debajo de su peso ideal la función devuelve un 0 y si devuelve un valor mayor que 25 significa que tiene sobrepeso, la función devuelve un 1. Te recomiendo que uses constantes para devolver estos valores.
* **esMayorDeEdad():** indica si es mayor de edad, devuelve un booleano.
* **comprobarSexo(char sexo):** comprueba que el sexo introducido es correcto. Si no es correcto, sera H. No sera visible al exterior.
* **comprobarDNI():** comprueba que el DNI introducido es correcto. El número de DNI está compuesto de 8 digitos. No será visible al exterior. Si no es correcto cuando se pasa al constructor, será por defecto.
* **toString():** devuelve toda la información del objeto.
* Métodos **set** y **get** de cada atributo

Ahora, crea una clase **Prueba** que haga lo siguiente:

* Crea 3 objetos de la clase persona:
  + Uno por defecto
  + Otro con el nombre, la edad y el sexo
  + Otro con todos los parámetros
* Pedir por teclado los datos que faltan del primer y del segundo objeto.
* Para cada objeto, deberá comprobar si está en su peso ideal, tiene sobrepeso o por debajo de su peso ideal con un mensaje.
* Indicar para cada objeto si es mayor de edad.
* Por último, mostrar la información de cada objeto.

1. Crea una clase llamada Libro que guarde la información de cada uno de los libros de una biblioteca. La clase debe guardar el título del libro, autor, número de ejemplares del libro y número de ejemplares prestados. La clase contendrá los siguientes métodos:

* Constructor por defecto.
* Constructor con parámetros.
* Métodos Setters/getters
* Método préstamo que incremente el atributo correspondiente cada vez que se realice un préstamo del libro. No se podrán prestar libros de los que no queden ejemplares disponibles para prestar. Devuelve true si se ha podido realizar la operación y false en caso contrario.
* Método devolución que decrementa el atributo correspondiente cuando se produzca la devolución de un libro. No se podrán devolver libros que no se hayan prestado. Devuelve true si se ha podido realizar la operación y false en caso contrario.
* Método toString para mostrar los datos de los libros. Este método se heredada de Object y lo debemos modificar (override) para adaptarlo a la clase Libro.

Escribe un programa para probar el funcionamiento de la clase Libro.

1. Define una clase llamada Punto que permita manejar un punto de dos coordenadas. Los atributos de la clase son:

* X define la coordenada x del punto
* Y define la coordenada y del punto.

Se deben definir los siguientes constructores:

* Un constructor por defecto
* Un constructor con los valores de las coordenadas
* Un constructor que recibe un objeto punto

Se deben definir los siguientes métodos:

* Métodos **set** y **get**.
* Método **setXY()** que permita modificar los valores de las coordenadas x e y a la vez
* Método **cuadranteXY()** que permita obtener el cuadrante en el que se encuentra el punto. Sabiendo que:
  + x=0 e y=0 está en el eje de coordenadas
  + x>0 e y>0 en el primer cuadrante
  + x<0 e y>0 en el segundo
  + x<o e y<0 en el tercero
  + x>0 e y<0 en el cuarto
* Método **puntoMedio()** que admite como parámetro un objeto de tipo punto y devuelve el punto medio entre el objeto y el punto que recibe como parámetro
* Método **distanciaPuntos()** que admite como parámetro un objeto Punto y devuelve la distancia entre los dos puntos.
* Sobre-escribe el método **toString()** para que muestre la información del punto en la forma (x,y)