Ejercicios Prácticos de Programación Orientada a Objetos

1) Crea una clase llamada Cuenta que tendrá los siguientes atributos: titular y cantidad (puede tener decimales).
El titular será obligatorio y la cantidad es opcional. Crea dos constructores que cumpla lo anterior.
Crea sus métodos get, set y toString.
Tendrá dos métodos especiales:

ingresar(double cantidad): se ingresa una cantidad a la cuenta, si la cantidad introducida es negativa, no se hará nada.

retirar(double cantidad): se retira una cantidad a la cuenta, si restando la cantidad actual a la que nos pasan es negativa, la cantidad de la cuenta pasa a ser 0.

2) Haz 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. Piensa que modificador de acceso es el más adecuado, también su tipo. Si quieres añadir algún atributo puedes hacerlo.
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 esta 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 esta 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.
- **toString()**: devuelve toda la información del objeto.
- **generaDNI()**: genera un número aleatorio de 8 cifras, genera a partir de este su número su letra correspondiente. Este método sera invocado cuando se construya el objeto. Puedes dividir el método para que te sea más fácil. No será visible al exterior.
- Métodos set de cada parámetro, excepto de DNI.

Ahora, crea una clase ejecutable que haga lo siguiente:
Pide por teclado el nombre, la edad, sexo, peso y altura.

Crea 3 objetos de la clase anterior, el primer objeto obtendrá las anteriores variables pedidas por teclado, el segundo objeto obtendrá todos los anteriores menos el peso y la altura y el último por defecto, para este último utiliza los métodos set para darle a los atributos un valor.
Para cada objeto, deberá comprobar si esta 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.
Puedes usar métodos en la clase ejecutable, para que os sea mas fácil.

3) Haz una clase llamada **Password** que siga las siguientes condiciones:
Que tenga los atributos **longitud** y **contraseña** . Por defecto, la longitud sera de 8.
Los constructores serán los siguiente:

- Un constructor por defecto.
- Un constructor con la longitud que nosotros le pasemos. Generara una contraseña aleatoria con esa longitud.

Los métodos que implementa serán:

- **esFuerte()**: devuelve un booleano si es fuerte o no, para que sea fuerte debe tener mas de 2 mayúsculas, mas de 1 minúscula y mas de 5 números.
- **generarPassword()**: genera la contraseña del objeto con la longitud que tenga.

Método get para contraseña y longitud.
Método set para longitud.

Ahora, crea una clase clase ejecutable:
Crea un array de Passwords con el tamaño que tu le indiques por teclado.
Crea un bucle que cree un objeto para cada posición del array.
Indica también por teclado la longitud de los Passwords (antes de bucle).
Crea otro array de booleanos donde se almacene si el password del array de Password es o no fuerte (usa el bucle anterior).
Al final, muestra la contraseña y si es o no fuerte (usa el bucle anterior). Usa este simple formato:
contraseña1 valor_booleano1
contraseña2 valor_bololeano2

4) Crearemos una clase llamada **Electrodomestico** con las siguientes características:
Sus atributos son **precio base, color, consumo energético** (letras entre A y F) y **peso**.
Por defecto, el color sera blanco, el consumo energético sera F, el precioBase es de 100 € y el peso de 5 kg. Usa constantes para ello.
Los colores disponibles son blanco, negro, rojo, azul y gris. No importa si el nombre esta en mayúsculas o en minúsculas.

Los constructores que se implementaran serán

Un constructor por defecto.

Un constructor con el precio y peso. El resto por defecto.

Un constructor con todos los atributos.

Los métodos que implementara serán:

- Métodos get de todos los atributos.

- **comprobarConsumoEnergetico(char letra)**: comprueba que la letra es correcta, sino es correcta usara la letra por defecto. Se invocara al crear el objeto y no sera visible.

- **comprobarColor(String color)**: comprueba que el color es correcto, sino lo es usa el color por defecto. Se invocara al crear el objeto y no sera visible.

- **precioFinal()**: según el consumo energético, aumentara su precio, y según su tamaño, también. Esta es la lista de precios:

5**)** Crear una clase Libro que contenga los siguientes atributos:

– ISBN

– Titulo

– Autor

– Número de páginas

Crear sus respectivos métodos get y set correspondientes para cada atributo.

Crear el método toString() para mostrar la información relativa al libro con el siguiente formato: “El libro con ISBN creado por el autor tiene páginas”

En el fichero main, crear 2 objetos Libro (los valores que se quieran) y mostrarlos por pantalla.

Por último, indicar cuál de los 2 tiene más páginas.

6**)** Vamos a realizar una clase llamada Raices, donde representaremos los valores de una ecuación de 2º grado.

Tendremos los 3 coeficientes como atributos, llamémosles a, b y c.

Hay que insertar estos 3 valores para construir el objeto.

Las operaciones que se podrán hacer son las siguientes:

- obtenerRaices(): imprime las 2 posibles soluciones

- obtenerRaiz(): imprime única raíz, que será cuando solo tenga una solución posible.

- getDiscriminante(): devuelve el valor del discriminante (double), el discriminante tiene la siguiente formula, (b^2)-4*a*c

- tieneRaices(): devuelve un booleano indicando si tiene dos soluciones, para que esto ocurra, el discriminante debe ser mayor o igual que 0.

- tieneRaiz(): devuelve un booleano indicando si tiene una única solución, para que esto ocurra, el discriminante debe ser igual que 0.

- calcular(): mostrara por consola las posibles soluciones que tiene nuestra ecuación, en caso de no existir solución, mostrarlo también.