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.
* Por defecto, todos los atributos menos el DNI 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)), devuelve un -1 si está por debajo de su peso ideal, un 0 si está en su peso ideal y un 1 si tiene sobrepeso .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 numero aleatorio de 8 cifras, genera a partir de este su número su letra correspondiente. Este método será 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 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.

**2)** Haz una clase llamada **Password** que siga las siguientes condiciones:

* Que tenga los atributos**longitud** y **contraseña** . Por defecto, la longitud será de 8.
* Los constructores serán los siguiente:
* Un constructor por defecto.
* Un constructor con la longitud que nosotros le pasemos. Generará 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

**3)** Crearemos una superclase 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 será blanco, el consumo energético será F, el precioBase es de 100 € y el peso de 5 kg.
* Los colores disponibles son blanco, negro, rojo, azul y gris. No importa si el nombre está 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 será 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 será visible.
* **precioFinal()**: según el consumo energético, aumentara su precio, y según su tamaño, también. Esta es la lista de precios:

|  |  |  |
| --- | --- | --- |
| **Letra** | **Precio** | |
| A | 100 € | |
| B | 80 € | |
| C | 60 € | |
| D | 50 € | |
| E | 30 € | |
| F | 10 € | |
| **Tamaño** | | **Precio** |
| Entre 0 y 19 kg | | 10 € |
| Entre 20 y 49 kg | | 50 € |
| Entre 50 y 79 kg | | 80 € |
| Mayor que 80 kg | | 100 € |

Crearemos una subclase llamada **Lavadora** con las siguientes características:

* Su atributo es **carga**, además de los atributos heredados.
* Por defecto, la carga es de 5 kg. Usa una constante para ello.
* Los constructores que se implementaran serán:
* Un constructor por defecto.
* Un constructor con el precio y peso. El resto por defecto.
* Un constructor con la carga y el resto de atributos heredados. Recuerda que debes llamar al constructor de la clase padre.
* Los métodos que se implementara serán:
* Método get de carga.
* **precioFinal()**:, si tiene una carga mayor de 30 kg, aumentara el precio 50 €, sino es así no se incrementara el precio. Llama al método padre y añade el código necesario. Recuerda que las condiciones que hemos visto en la clase Electrodomestico también deben afectar al precio.

Crearemos una subclase llamada **Television** con las siguientes características:

* Sus atributos son **resolución** (en pulgadas) y **sintonizador TDT** (booleano), además de los atributos heredados.
* Por defecto, la resolución será de 20 pulgadas y el sintonizador será false.
* Los constructores que se implementaran serán:
* Un constructor por defecto.
* Un constructor con el precio y peso. El resto por defecto.
* Un constructor con la resolución, sintonizador TDT y el resto de atributos heredados. Recuerda que debes llamar al constructor de la clase padre.
* Los métodos que se implementara serán:
* Método get de resolución y sintonizador TDT.
* **precioFinal()**: si tiene una resolución mayor de 40 pulgadas, se incrementara el precio un 30% y si tiene un sintonizador TDT incorporado, aumentara 50 €. Recuerda que las condiciones que hemos visto en la clase Electrodomestico también deben afectar al precio.

**4)** Crearemos una clase llamada **Serie** con las siguientes características:

* Sus atributos son **titulo, numero de temporadas**, **entregado, genero y creador.**
* Por defecto, el numero de temporadas es de 3 temporadas y entregado **false**. El resto de atributos serán valores por defecto según el tipo del atributo.
* Los constructores que se implementaran serán:
* Un constructor por defecto.
* Un constructor con el titulo y creador. El resto por defecto.
* Un constructor con todos los atributos, excepto de entregado.
* Los métodos que se implementara serán:
* Métodos get de todos los atributos, excepto de entregado.
* Métodos set de todos los atributos, excepto de entregado.
* Sobrescribe los métodos toString.

Crearemos una clase **Videojuego** con las siguientes características:

* Sus atributos son **titulo, horas estimadas, entregado, genero y compañia**.
* Por defecto, las horas estimadas serán de 10 horas y entregado false. El resto de atributos serán valores por defecto según el tipo del atributo.
* Los constructores que se implementaran serán:
* Un constructor por defecto.
* Un constructor con el titulo y horas estimadas. El resto por defecto.
* Un constructor con todos los atributos, excepto de entregado.
* Los métodos que se implementara serán:
* Métodos get de todos los atributos, excepto de entregado.
* Métodos set de todos los atributos, excepto de entregado.
* Sobrescribe los métodos toString.

Como vemos, en principio, las clases anteriores no son padre-hija, pero si tienen en común, por eso vamos a hacer una interfaz llamada **Entregable** con los siguientes métodos:

* **entregar()**: cambia el atributo prestado a true.
* **devolver()**: cambia el atributo prestado a false.
* **isEntregado()**: devuelve el estado del atributo prestado.
* Método **compareTo (Object a)**, compara las horas estimadas en los videojuegos y en las series el numero de temporadas. Como parámetro que tenga un objeto, no es necesario que implementes la interfaz Comparable. Recuerda el uso de los casting de objetos.

Implementa los anteriores métodos en las clases Videojuego y Serie. Ahora crea una aplicación ejecutable y realiza lo siguiente:

* Crea dos arrays, uno de**Series** y otro de **Videojuegos**, de 5 posiciones cada uno.
* Crea un objeto en cada posición del array, con los valores que desees, puedes usar distintos constructores.
* Entrega algunos **Videojuegos** y **Series** con el método **entregar()**.
* Cuenta cuantos **Series** y **Videojuegos** hay entregados. Al contarlos, devuélvelos.
* Por último, indica el **Videojuego** tiene más horas estimadas y la serie con mas temporadas. Muestralos en pantalla con toda su información (usa el método toString()).

5 - Se necesita una aplicación para la gestión académica de un centro. Por un lado necesitamos guardar los cursos que tendrán título, descripción, horas lectivas y profesor que lo imparte. Crea la clase necesaria junto con sus atributos, constructores, getters y setters. También tendremos la clase alumno que deberá guardar información del nombre, apellidos, nif, y cursos en los que se ha matriculado.

Finalmente tendremos una clase profesor que guardará nombre, apellidos, nif, titulación del profesor.

Diseña el programa haciendo que tanto alumno como profesor sean clases hijas una clase persona.

Diseña el programa principal que permita crear alumnos y asignarles cursos, también crear profesores y asignarlos a los diferentes cursos. Una vez hecho eso saca un listado de alumnos que asisten a algún curso del profesor “Pepito Grillo”.

6. Haz un programa que utilizando una clase Persona pueda representar empleados y clientes. Por cada empleado guardaremos su nombre, apellidos, dni, dirección, departamento en el que trabaja y sueldo. Por cada cliente almacenaremos nombre, apellidos, dni, teléfono de contacto y empresa a la que pertenece. A su vez para las empresas tendremos guardados nombre de empresa, nif, dirección postal y cantidad facturada al año.

Se pide un programa principal que permita dar de alta empleados, clientes y empresas, y asociar los clientes a las empresas. Deberemos contar con un método que nos permita obtener el empleado de mayor sueldo, así como los que están por encima de la media.

También se pide obtener la lista de los clientes pertenecientes a las tres empresas que más facturan.