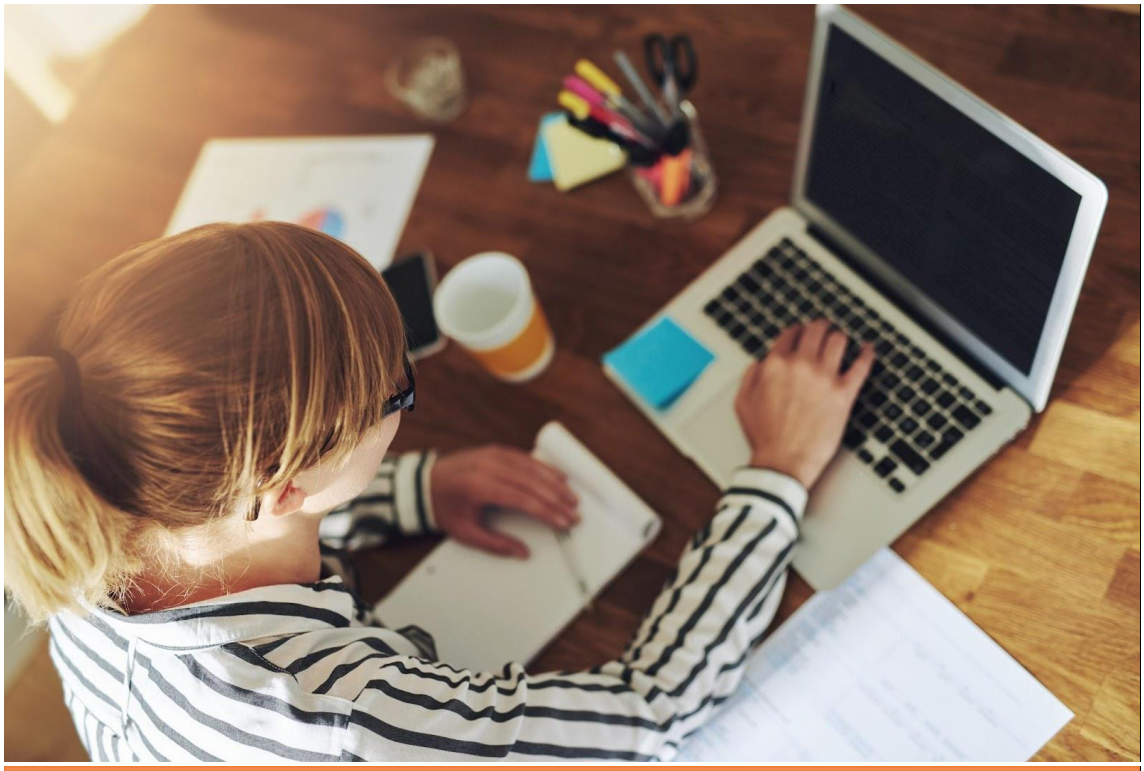

sofka
UNIVERSITY



Document Version: 1.0

Versión	Comentarios	Fecha	Descripción	Responsable
1.0	Creación	10/07/2018	Creación Documento	Manuel Salas

Tabla de Contenido

1. **Objetivo**

Practicar los conceptos básicos aprendidos y estudiados en el módulo de Desarrollo de Software en Java.

2. **Ejercicios Propuestos**

1. Declara 2 variables numéricas (con el valor que desees), he indica cual es mayor de los dos. Si son iguales indicarlo también. Ve cambiando los valores para comprobar que funciona.
2. Al ejercicio anterior agregar entrada por consola de dos valores e indicar si son mayores, menores o iguales.
3. Haz una aplicación que calcule el área de un círculo(pi*R2). El radio se pedirá por teclado (recuerda pasar de String a double con **Double.parseDouble**). Usa la constante PI y el método pow de Math.
4. Lee un número por teclado que pida el precio de un producto (puede tener decimales) y calcule el precio final con IVA. El IVA sera una constante que sera del 21%.
5. Muestra los números impares y pares del 1 al 100 (ambos incluidos). Usa un bucle while.
6. Realiza el ejercicio anterior usando un ciclo for.
7. Lee un número por teclado y comprueba que este número es mayor o igual que cero, si no lo es lo volverá a pedir (do while), después muestra ese número por consola.
8. Crea una aplicación por consola que nos pida un día de la semana y que nos diga si es un día laboral o no. Usa un **switch** para ello.
9. Del texto, “La sonrisa sera la mejor arma contra la tristeza” Reemplaza todas las **a** del String anterior por una **e**, adicionalmente concatenar a esta frase una adicional que ustedes quieran incluir por consola y las muestre luego unidas.
10. Realizar una aplicación de consola, que al ingresar una frase por teclado elimine los espacios que esta contenga.
11. Realizar la construcción de un algoritmo que permita de acuerdo a una frase pasada por consola, indicar cual es la longitud de esta frase, adicionalmente cuantas vocales tiene de “a,e,i,o,u”.
12. Pedir dos palabras por teclado, indicar si son iguales, sino son iguales mostrar sus diferencias.
13. Realizar un algoritmo que permita consulta la fecha y hora actual en el formato (AAAA/MM/DD) (HH:MM:SS)
14. Crear un programa que pida un numero por teclado y que imprima por pantalla los números desde el numero introducido hasta 1000 con saldos de 2 en 2.

15. Hacer un programa que muestre el siguiente menú de opciones

****** GESTION CINEMATOGRÁFICA ********
1- NUEVO ACTOR
2- BUSCAR ACTOR
3- ELIMINAR ACTOR
4- MODIFICAR ACTOR
5- VER TODOS LOS ACTORES
6- VER PELICULAS DE LOS ACTORES
7- VER CATEGORIA DE LAS PELICULAS DE LOS ACTORES
8- SALIR

EL SISTEMA SOLO VA A SALIR CUANDO SE DIGITE EL NUMERO 8, MIENTRAS SE DIGITE UNA DE LAS CINCO OPCIONES DEBE SEGUIR MOSTRADO EL MENU Y SI EL USUARIO DIGITA UN NUMERO QUE NO ESTA EN EL MENU SE DEBE ARROJAR UN MENSAJE " OPCION INCORRECTO". Y MOSTRAR EL MENU NUEVAMENTE.
PISTA: CONVINAR SWICHT Y ALGUNO DE LOS BUCLES.

16. 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, usa una constante para ello.
Se implantarán 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 será 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 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.

17. Crearemos una superclase llamada Electrodoméstico con las siguientes características:

    - Sus atributos son **precio base, color, consumo energético** (letras entre A y F) y **peso**. Indican que se podrán heredar.
    - Por defecto, el color será blanco, el consumo energético será F, el precioBase es de 100 € y el peso de 5 kg. Usa constantes para ello.
    - Los colores disponibles son blancos, negro, rojo, azul y gris. No importa si el nombre está en mayúsculas o en minúsculas.
    - Los constructores que se implementarán 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 invocará 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 invocará 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 implementarán 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 implementarán 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.

Ahora crea una clase ejecutable que realice lo siguiente:

- Crea un array de Electrodomésticos de 10 posiciones.
- Asigna a cada posición un objeto de las clases anteriores con los valores que desees.
- Ahora, recorre este array y ejecuta el método precioFinal().

- Deberás mostrar el precio de cada clase, es decir, el precio de todas las televisiones por un lado, el de las lavadoras por otro y la suma de los Electrodomésticos (puedes crear objetos Electrodoméstico, pero recuerda que Televisión y Lavadora también son electrodomésticos). Recuerda el uso operador instance of.

Por ejemplo, si tenemos un Electrodoméstico con un precio final de 300, una lavadora de 200 y una televisión de 500, el resultado final será de 1000 (300+200+500) para electrodomésticos, 200 para lavadora y 500 para televisión.

18. Crear una clase llamada **Serie** con las siguientes características:

- Sus atributos son **título, numero de temporadas**, **entregado, género y creador.**
- Por defecto, el número 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 implementarán serán:

    - Un constructor por defecto.
    - Un constructor con el título 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 **título, horas estimadas, entregado, género y compañía**.
- 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 implementarán serán:

    - Un constructor por defecto.
    - Un constructor con el título 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 número de temporadas. Como parámetro que tenga un objeto, no es necesario que implementes la interfaz Comparable. Recuerda el uso del 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 más temporadas. Muéstralos en pantalla con toda su información (usa el método toString()).