ACTIVIDADES TEMA 0.- PROGRAMACIÓN CON JAVA.

***Actividad 1**. Pedir por teclado el Nombre, Sueldo, Categoría Laboral (1, 2, 3 o 4) y Población de un empleado. Si el empleado es de Valencia aumentar el sueldo un 20% y mostrar los datos finales del modo:*
El empleado {nombre} con categoría laboral {cLaboral} de {población} cobra {sueldoFinal}

***Actividad 2**. Pedir por teclado el Nombre, Sueldo, Categoría Laboral (1, 2, 3 o 4) y Población de un empleado. Si el empleado es de Valencia aumentar el sueldo un 20% , en caso contrario aumentar el sueldo un 10%. Mostrar los datos finales del modo:*
El empleado {nombre} con categoría laboral {cLaboral} de {población} cobra {sueldoFinal}

***Actividad 3**. Pedir por teclado el Nombre, Sueldo, Categoría Laboral (1, 2, 3 o 4) y Población de un empleado. Si la Categoría Laboral del empleado es 1 aumentar el sueldo un 10%, si la Categoría Laboral del empleado es 2 aumentar el sueldo un 20%, si la Categoría Laboral del empleado es 3 aumentar el sueldo un 30%, si la Categoría Laboral del empleado es 4 aumentar el sueldo un 40% y en cualquier otro caso el sueldo no se modifica. Mostrar los datos finales del modo:*
El empleado {nombre} con categoría laboral {cLaboral} de {población} cobra {sueldoFinal}

***Actividad 4**. Introducir por teclado el importe de 10 facturas y calcular el importe total de las facturas. Mostrar los datos finales del modo:*
El total del las facturas es {importeFinal}

***Actividad 5**. Introducir por teclado el importe de facturas hasta introducir el valor 0 y mostrar el menor importe de las facturas. Realizar el ejercicio con un bucle do-while. Mostrar los datos finales del modo:*
El menor importe de las facturas introducido es {importeMenor}

***Actividad 6**. Introducir por teclado el importe de facturas hasta introducir el valor 0 y mostrar el mayor importe de las facturas. Realizar el ejercicio con un bucle while. Mostrar los datos finales del modo:*
El mayor importe de las facturas introducido es {importeMayor}

***Actividad 7**. Pedir por teclado dos fechas e indicar el día de la semana de cada fecha y los días transcurridos entre ambas fechas. Mostrar los datos finales del modo:*
Día de la Semana de la Fecha Inicial: {diaSemanaInicial}
Día de la Semana de la Fecha Final: {diaSemanalFinal}
Días transcurridos entre {fechaIncial} y {fechaFinal}: {diasTranscurridos}

***Actividad 8**. Crear una aplicación con una clase llamada mueble. Las propiedades o atributos de la clase mueble serán el código del mueble, material y el precio. Los métodos de la clase Inmueble serán el método constructor por defecto, el constructor que inicializa los valores de los atributos, los métodos set y get y el método toString.*

Además un método sobrecargado llamado calculaIVA. Si no recibe parámetro, calcula el 21% sobre el precio. Y si recibe como parámetro un porcentaje de IVA y devuelve el importe del IVA aplicando dicho porcentaje.

***Actividad 9**. Crear dos subclases de la clase mueble de la actividad anterior llamadas Mesa y Silla.*

La subclase Mesa hereda de la clase mueble atributos y métodos. Además dispone de atributos con el ancho, largo de la mesa. Además se redefine el método constructor que permite inicializa todos sus atributos y el método toString() que muestra los atributos todos los atributos de la subclase.

La subclase Silla hereda de la clase mueble atributos y métodos. Además dispone de atributos con el numPatas, materialTapizado y respaldo (boolean). Además se redefine el método constructor que permite inicializa todos sus atributos y el método toString() que muestra los atributos todos los atributos de la subclase.

Crear varias instancias de ambas subclases, mostrar sus datos, calcular su precio final para todas las instancias.

***Actividad 10**. Crear una aplicación que permita gestionar la cola de matriculas de estudiantes en una Universidad:*

- *Crear una clase llamada Alumno que tenga los atributos dni, nombre, apellidos y curso.*
- *Crear constructor que inicialice todos los atributos de la clase, métodos setter, getter y toString().*
- *Crear un ArrayList para almacenar los alumnos.*

Realizar las siguientes acciones:

- *Introducir alumnos y almacenarlos en el ArrayList.*
- *Listar todos los alumnos que hay en la cola.*
- *Eliminar un alumno atendido de la cola (el primero).*
- *Volver a listar todos los alumnos que hay en la cola.*