**Herencia**

1. Crea una clase llamada Punto3D que herede de la clase Punto2D. Un Punto3D queda definido por tres atributos que almacenan las coordenadas del punto: x, y, z. La clase Punto2D tendrá los mismos constructores y métodos que la clase Punto2D.

Constructores

Constructor por defecto

Constructor que recibe el valor de las coordenadas

Constructor que crea un punto como una copia de otro punto

Métodos

Asignar nuevos valores a un punto

Métodos para obtener cada una de las coordenadas del punto

Para obtener el punto medio entre el punto actual y el que se le pasa como parámetro

Para obtener la distancia entre el punto actual y el que se le pasa como parámetro

1. Crea una clase llamada **Electrodoméstico** 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.

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 a implementar son:

* 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**, que hereda de la clase Electrodomestico 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**, que hereda de Electrodomestico, con las siguientes características:

* Los atributos son resolución (en pulgadas) y sintonizador TDT (booleano), ademas de los atributos heredados.
* Por defecto, la resolución será de 20 pulgadas y el sintonizador sera 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 a implementar son:

* 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 Electrodomesticos 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 Electrodomesticos (puedes crear objetos Electrodomestico, pero recuerda que Television y Lavadora también son electrodomésticos). Recuerda el uso operador **instanceof.**

Por ejemplo, si tenemos un Electrodomestico 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.

1. Se desea modelar el comportamiento de una empresa, para ello será necesario:

Desarrollar una clase **Empresa** cuyos atributos sean un nombre, un tamaño y un array de empleados personal (array de la clase Empleado, se pide posteriormente). El tamaño de la empresa no puede ser modificado una vez se crea la empresa. Para esta clase se piden los siguientes métodos:

* El constructor **Empresa** que recibe como argumentos una cadena de texto nombre y un valor entero tamaño. El tamaño del array de empleados personal viene dado por el valor de la variable tamaño pasada al constructor.
* Los métodos de acceso **getNombre** y **getTamaño**.
* El método de acceso **obtenerEmpleado**, que recibe como argumento un número entero menor que el tamaño de la empresa, y devuelve los datos correspondientes al empleado que se encontraba en esa posición. Si el número entero que recibe es mayor o igual que el tamaño de la empresa se mostrará un mensaje por pantalla y se retornará **null**.
* El método **despideEmpleado**, que recibe como argumento un número entero menor que el tamaño de la empresa, y asigna la referencia null al correspondiente campo del array de empleados. Si el número entero que recibe es mayor o igual que el tamaño de la empresa se mostrará un mensaje por pantalla advirtiendo del error.

Desarrollar una clase **Empleado** cuyos atributos sean una empresa, un nombre, un sueldo y un número de empleado (numEmpleado). Estos datos han de ser protegidos (protected). Además, el número de empleado y la empresa son inmutables. Para esta clase se piden los siguientes constructores y métodos:

* El constructor **Empleado** que recibe como argumentos empr, (una referencia a un objeto Empresa), una cadena de texto nombre y un valor entero sueldo, un valor entero numEmpleado.
* Los métodos de acceso **getNombre**, **getSueldo** y **getNumeroEmpleado**.
* Los métodos **setNombre** y **setSueldo**.
* El método sobrescrito **toString** de la clase Object (véase formato) que devuelve un String con el nombre, número y sueldo de un objeto Empleado.
* El método **aumentarSueldo**, que recibe como argumento un número entero N y módica el sueldo del objeto Empleado sobre el que se evalúa, aumentándolo un N%.

Añadir a la clase Empresa el método **nuevoEmpleado**, que recibe como argumentos una cadena de texto nombre y un valor entero sueldo. Buscará en la primera posición libre (que no sea null) del array personal, que indicará el número del nuevo empleado a crear y la posición del array donde se almacena. Si no hay posiciones libres mostrara un mensaje por pantalla indicando la situación.

* Desarrollar una clase Ejecutivo derivada de la clase Empleado anterior, con un campo entero adicional **productividad**. Para esta clase se piden los siguientes constructores y métodos:
* El constructor **Ejecutivo** que recibe como argumentos empr (una referencia a un objeto Empresa), una cadena de texto nombre, un valor entero sueldo, un valor entero productividad y un número de empleado numero.
* Los métodos de acceso **getProductividad** y **setProductividad**.
* Redefinir el método **toString** para que devuelva una cadena (String) con el nombre, número, sueldo y productividad del Ejecutivo.
* Redefinir el método **aumentarSueldo**, que recibe como argumento un número entero N y modifica el sueldo del Ejecutivo sobre el que se evalúa, aumentándolo en (productividad\*N)%.

Añadir a la clase Empresa un método **ascender** que reciba un número de empleado y la productividad que va a tener en su nuevo puesto y que cree un nuevo objeto Ejecutivo con los datos del objeto Empleado sobre el que se evalúa el método y cambia la referencia en el array de personal de la empresa a la que pertenece dicho objeto.

Añadir a la clase Empresa un método **aumentarSueldoATodos**, que aumentará el sueldo a todos los empleados. Recibirá el porcentaje a aplicar a todos los empleados (tened en cuenta que gracias a la herencia y el polimorfismo, no será necesario diferenciar entre Empleados normales y Ejecutivos).

Crear una clase **PruebaEmpresa** en la que se cree una empresa (se solicitará al usuario el número máximo de empleados que puede tener la empresa, así como el nombre de la misma) y mediante un menú textual se ofrecerán las siguientes opciones:

Añadir a un nuevo empleado: se pedirá al usuario su nombre y su sueldo.

Consultar los datos de un empleado (por supuesto puede ser un ejecutivo): se pedirá al usuario el número del empleado que se quiere consultar.

Despedir a un empleado (por supuesto puede ser un ejecutivo): se pedirá al usuario su número de empleado. Previamente a su despido se mostraran sus datos por pantalla y se pedirá confirmación.

Ascender a un empleado: se pedirá al usuario el número de empleado. Si es un ejecutivo se indicara que no se le puede ascender.

Aumentar el sueldo a todos los empleados de la empresa. Se pedirá al usuario el porcentaje de subida.