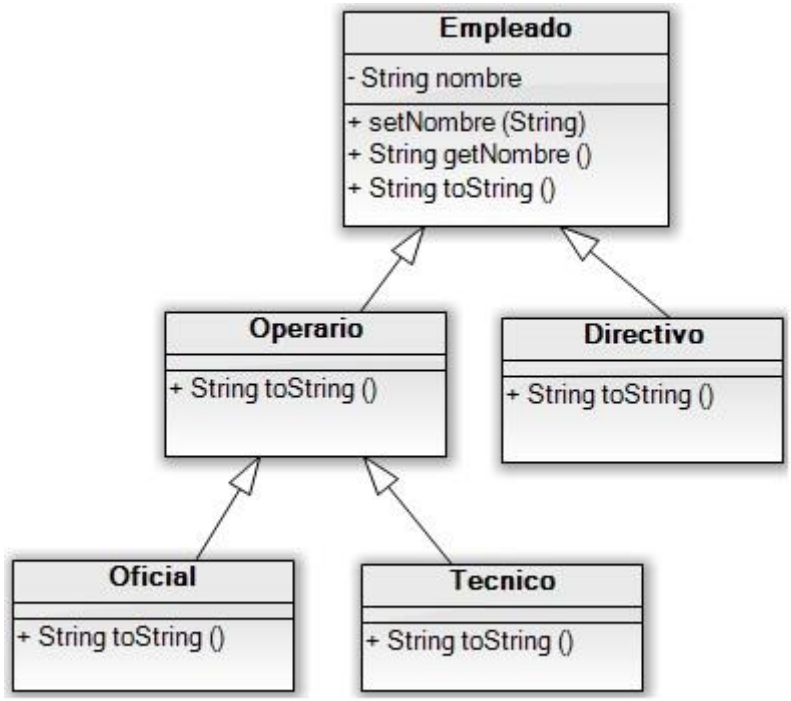
Actividad 8

Ejercicio 1. Codificar la siguiente jerarquía de clases en C# representada por este diagrama UML:



(Nota: el atributo "nombre" de la clase Empleado es público)

Todas las clases deben de sobrescribir el método toString() para que quede de la siguiente manera:

Empleado Rafa

Empleado Mario -> Directivo

Empleado Alfonso -> Operario

Empleado Luis -> Operario -> Oficial

Empleado Pablo -> Operario -> Tecnico

Ejercicio 2. Crear una clase llamada Electrodoméstico con las siguientes características:

- Sus atributos son precio_base, color, consumo (letras entre A y F) y peso.
- Por defecto, el color será blanco, el consumo F, el precio_base 100 y el peso 5kg. Usa constantes para ello.
- Los colores disponibles son blanco, negro, rojo, azul y gris. No importa si el nombre está en mayúscula o minúnculas.
- Los constructores que se implementarán serán:
    - Por defecto.

    - Un constructor con el precio y peso, el resto por defecto.
    - Un constructor con todos los atributos.
- Los métodos a implementar serán:
    - comprobarConsumoEnergia(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.
    - comprobarColo(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 €

Creamos una clase llamada Lavadora con las siguientes características:

- Atributo carga, además de los atributos heredados.
- Por defecto, la carga es de 5kg. Usa una constante para ello.
- Los constructores que se implementarán serán:
    - Constructor por defecto
    - Constructor con el precio y peso. El resto por defecto
    - Un constructor con la carga y el resto de atributos heredados.
- Métodos a implementar:
    - precioFinal(): si tiene carga mayor a 30kg, aumenta el precio 50€, sino es así no se incrementa el precio. Llama al método padre y añade el código necesario.

Creamos una clase Televisión con las siguientes características:

- Sus atributos con 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:
    - Constructor por defecto
    - Constructor con el precio y el peso, el resto por defecto

    - Un constructor con la resolución, sintonizador TDT y el resto de atributos heredados.
- Los métodos que se implementarán serán:
    - 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.

Por último, crear una clase Main:

- Crear un array de Electrodomésticos de 10 posiciones.
- Asigna a cada posición un objeto de las clases anteriormente creadas con los valores que desees.
- Recorre ese array ejecuta el método precioFinal()
- Muestra por pantalla 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 todos los electrodomésticos.

Ejercicio 3. Crearemos 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:
    - 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:
    - 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 entregado a true.
- devolver(): cambia el atributo entregado a false.
- isEntregado(): devuelve el estado del atributo entregado.

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 cuantas Series y Videojuegos hay entregados. Al contarlos, devuélvelos.