Unión Europea
Fondo Social Europeo
El FSE invierte en tu futuro

Fecha	Versión	Descripción
13/12/2021	1.0.0	Versión inicial

Ejercicios de Programación Orientada a Objetos

Librería

1. Crea una clase llamada Libro que contenga los siguientes atributos:
    - **codigo** de tipo String: Almacenará el código del libro.
    - **titulo** de tipo String: Almacenará el título del libro.
    - **autor** de tipo String: Almacenaré el autor del libro.
    - **numPaginas** de tipo int: Almacenará el número de páginas del libro.
2. Para esta clase de libros definiremos los siguientes métodos:
    - **constructor**: Se encargará de crear objetos de esta clase.
    - **Getters** y **Setters** para cada uno de los atributos.
    - Un método llamado **mostrar** que devolverá un String con el código de libro, titulo y autor.
3. Crea un programa principal en el cual se creen 4 objetos y luego invoquemos al método mostrar para poder listar estos libros.

Coche

1. Crea una clase llamada Coche para registrar información sobre un coche. La clase tiene tres getters:
    - **getMarca()** devuelve la marca del automóvil como un String.
    - **getModelo()** devuelve el modelo del automóvil como un String.

    - **getAnyo()** que devuelve el año en que se fabricó el coche como valor int.
2. La clase tiene 4 setters:
    - **setMarca(String marca)**
    - **setModelo(String modelo)**
    - **setAnyo(int anyo)**
    - **setAnyo(String anyo)**
3. Escribe dos constructores para la clase.
    - **Coche (String marca, String Modelo, int Anyo)**
    - **Coche (String marca, String Modelo, String Anyo)**
4. Crea además un método para listar los coches:
    - **mostrar()** devuelve un String con la siguiente información: Marca + modelo + anyo.
    - **claxon()** devolverá por pantalla el valor "piiiiii, piiiiii"
5. Crea un Array que contendrá 4 coches. Luego lista estos coches.

Hora

1. Crea una clase denominada Hora que nos permita representar la hora en formato 24h. La clase tiene los siguientes getters:
    - **getHoras()** nos devolverá la hora.
    - **getMinutos()** nos devolverá los minutos.
2. La clase además dispondrá de los siguientes métodos:
    - **adelantar(int minutos)** que adelantará la hora en los minutos indicados.
    - **retrasar(int minutos)** que retrasará la hora en los minutos indicados.
    - **convertir ()** que devolverá en un String la hora en formato AM/PM.
3. Un constructor que nos permita crear la Hora.
4. Un programa principal que almacene 'X' horas y luego incrementemos estas, las decrementemos y les cambiemos el formato.

Conecta4

Para este ejercicio vamos a crear el juego de Conecta4, el cual se juega por 2 personas en un tablero de 6 filas y 7 columnas. Cada uno de los jugadores marca una posición del tablero, uno con una 'X' y el otro con una 'O'. Gana aquel que consigue colocar 4 posiciones consecutivas de su ficha bien en horizontal, vertical o diagonal. Este tablero lo representaremos como un array de char con un tamaño de 6X7.

1. Crea la clase Conecta4 con los siguientes métodos:
    - **constructor** para la clase.
    - **boolean estaDisponible(int x, int y)** devuelve true si la posición indicada por la fila x y la columna y se encuentra ocupada, false en caso contrario.
    - **boolean esO(int x, int y)**, devuelve true si en la posición indicada el valor que contiene es una O o false en caso contrario.

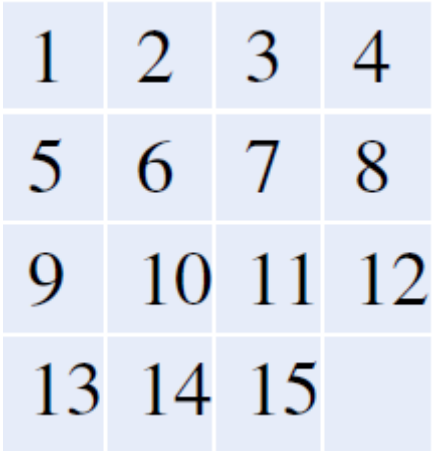
- **boolean esX(int x, int y)**, devuelve true si en la posición indicada el valor que contiene es una X o false en caso contrario.
- **void setO(int x, int y)**, escribe en la posición x,y el valor 'O'.
- **void setX(int x, int y)**, escribe en la posición x,y el valor 'X'.
- **void mostrarTablero()**, muestra por pantalla el tablero.

2. Crea un programa principal en el que se cree un tablero y se permita jugar a 2 jugadores, jugador1 y jugador2. Después de varias jugadas, muestra el resultado del mismo.

Puzle

Vamos a desarrollar un programa para resolver puzles de 15 piezas ubicadas en un tablero de 4x4.

Los tamaños de las piezas son iguales al tamaño de cualquier cuadrado de la cuadrícula. Las piezas están numeradas del 1 al 15. Al comienzo del juego, las piezas se colocan en uno de los 16 cuadrados de la cuadrícula, por lo que solo hay un cuadrado abierto. Durante el transcurso del puzle , el jugador puede mover una pieza desde cualquiera de los cuadrados vecinos (de izquierda, derecha, arriba o abajo) al cuadrado abierto. El objetivo del rompecabezas es reordenar las piezas para que del 1 al 4 aparezcan en la fila superior de izquierda a derecha, del 5 al 8 aparezcan en la siguiente fila de izquierda a derecha, del 9 al 12 aparezcan en la siguiente fila, y del 13 al 15 en la fila inferior, con el cuadrado más a la derecha abierto, como se muestra en el siguiente diagrama:



Para jugar el puzle, el propio jugador revuelve el orden de las piezas, moviéndolas al azar de acuerdo con la regla.

Escriba una clase llamada **Puzle** cuyo objeto represente una configuración del puzle.
La clase debe tener tres atributos. El primero es una matriz de 4 × 4 de valores int, donde los números 0,. . . , 15 aparecen exactamente una vez. En esta matriz, 0 representa el cuadrado abierto. El segundo y tercer parámetro son los índices de fila y columna del cuadrado abierto. El constructor toma una matriz en este formato y copia el contenido en el atributo de la matriz, y luego busca un cuadrado abierto en esta matriz que se proporciona para asignar a los atributos de la fila y la columna.

Debes crear los siguientes métodos, además del constructor indicado anteriormente.

- **boolean estaResuelto()**: este método devuelve si la configuración representada por el objeto está resuelto o no.

- **boolean bajar()**: Este método mueve la pieza que se encuentra encima del cuadrado abierto hacia el espacio abierto. El movimiento no es posible si el cuadrado abierto está en la primera fila. El método devuelve como un valor booleano que indica si el movimiento fue exitoso o no.
- **boolean subir()**: Este método mueve la pieza que se encuentra bajo del cuadrado abierto hacia el espacio abierto. El movimiento no es posible si el cuadrado abierto está en la última fila. El método devuelve como un valor booleano que indica si el movimiento fue exitoso o no.
- **boolean moverIzquierda()**: Este método mueve la pieza en el cuadrado inmediatamente a la izquierda del cuadrado abierto. El movimiento no es posible si el cuadrado abierto está en la primera columna. El método devuelve si el movimiento se ha realizado correctamente o no.
- **boolean moverDerecha()**: Este método mueve la pieza en el cuadrado inmediatamente a la derecha del cuadrado abierto. El movimiento no es posible si el cuadrado abierto está en la última columna. El método devuelve si el movimiento se ha realizado correctamente o no.
- **int getPosicion(int x, int y)**: Devuelve el valor contenido para la posición x,y de la matriz.
- **int getFilaEspacio()**: Devuelve la fila en la que se encuentra la casilla con espacio.
- **int getColumnaEspacio()**: Devuelve la columna en la que se encuentra la casilla con espacio.
- **void mostrar()**: Mostrará por pantalla el tablero del puzle.

Crea un programa principal en la que se cree un objeto puzle e invoca la funciones para realizar los movimientos sobre este.