Práctica 1: Conecta 4

Fecha de entrega: 16 de Noviembre de 2014, 16:00

Objetivo: Iniciación a la orientación a objetos y a Java; uso de arrays y enumerados; manipulación de cadenas con la clase `String`; entrada y salida por consola.

1. Introducción

El Conecta 4 es un conocido juego de tablero normalmente colocado en vertical donde dos jugadores van poniendo fichas alternativamente hasta que uno de ellos consigue colocar cuatro de sus fichas formando en fila, ya sea en vertical, horizontal o diagonal.

En esta primera práctica implementaremos una aplicación que permita a dos jugadores echar una partida a Conecta 4.

2. Descripción de la práctica

Al lanzar la aplicación, se mostrará el tablero vacío y se indicará de quién es el turno (en nuestra implementación empezarán siempre las blancas). A continuación, se le preguntará al usuario qué es lo que quiere hacer, a lo que podrá contestar las siguientes cuatro alternativas:

- Poner: se preguntará al usuario en qué columna desea poner la ficha, y ésta aparecerá colocada en el tablero.
- Deshacer: se anulará el último movimiento. Aparecerá el nuevo estado del tablero y de quién es el turno. La aplicación debe garantizar que se podrán deshacer al menos 10 movimientos (si es que los ha habido en la partida). Si no hay nada que deshacer se mostrará un mensaje de error.
- Reiniciar: reiniciará la partida, vaciando el tablero. Esta operación *no* se puede deshacer.
- Salir: saldrá de la aplicación.

El programa preguntará al usuario qué acción quiere ejecutar y, tras completarla, volverá a mostrar el estado del tablero y el jugador.

Cuando uno de los jugadores consigue colocar cuatro fichas en línea (consigue cuatro en raya), indicará que la partida ha terminado, dirá quién es el ganador y terminará. Si el tablero se completa sin tener un ganador, la partida terminará en tablas. Se indicará "`Partida terminada en tablas.`" y se acabará la ejecución.

3. Implementación

Para lanzar la aplicación se ejecutará la clase `tp.pr1.Main`, por lo que se aconseja que todas las clases desarrolladas en la práctica estén en el paquete `tp.pr1` (o subpaquetes suyos). Para implementar la práctica necesitarás, al menos, las siguientes clases:

- Ficha: enumerado que representa un color de ficha. Contiene las constantes de enumeración VACIA, BLANCA y NEGRA.
- Tablero: clase que almacena el estado de un tablero. La implementación será genérica, pues permitirá tableros de tamaño distintos al usado en el Conecta 4. Como atributos privados necesitarás al menos:

  ```
  private Ficha [ ][ ] tablero;
  private int ancho;
  private int alto;
  ```

  para almacenar el tablero, y el número de columnas y filas respectivamente. La clase contendrá una constructora con dos argumentos que fijan las dimensiones del tablero, métodos de consulta para el alto y ancho del tablero, así como para devolver la ficha colocada en una posición, o colocar una ficha en una determinada posición. El tablero se puede reiniciar a través de su método público public void reset(), que vacía todas las casillas del tablero. Recuerda implementar el método público public String toString().
- Partida: para representar el estado de una partida (tablero, turno, etc.). Entre sus atributos privados se encuentran:

  ```
  private Tablero tablero;
  private Ficha turno;
  private boolean terminada;
  private Ficha ganador;
  ```

  Esta clase contiene, entre otros, el método público public boolean ejecutaMovimiento(Ficha color, int col), que es el encargado de colocar la ficha color en la columna col correspondiente a turno en tablero, comprobar si la partida ha terminado actualizando convenientemente el atributo terminada y, en caso de no haber terminado, cambiar el turno del jugador.

  Para poder ejecutar la orden Deshacer, esta clase contiene además una pila de enteros con capacidad para al menos 10 elementos. Para representar la pila puedes utilizar los atributos:

  ```
  private int[ ] undoStack;
  private int numUndo;
  ```

donde undoStack almacena las columnas y numUndo representa el número de posiciones ocupadas en el array.

- Controlador: controla la ejecución de la aplicación, preguntando al usuario qué quiere hacer y actualizando la partida de acuerdo a lo que éste indique. El controlador necesita dos atributos privados:

```
private Partida partida;
private Scanner in;
```

  e implementa el método público public void run() que realiza la simulación del juego. Concretamente, mientras la partida no esté finalizada, solicita una orden al usuario (salir, poner, deshacer, reiniciar) y la ejecuta invocando a algún método de partida, excepto para el comando salir que finaliza la aplicación. En el caso de que, tras ejecutar la orden poner, haya un ganador o bien la partida termine en tablas, la ejecución de la aplicación termina informando al usuario del resultado final.

- Main: Es la clase que contiene el método main de la aplicación. En este caso el método main simplemente crea una partida vacía, crea un controlador con dicha partida, e invoca al método run del controlador.

El resto de información concreta para implementar la práctica será explicada por el profesor durante las distintas clases de teoría y laboratorio. En esas clases se indicará qué aspectos de la implementación se consideran obligatorios para poder aceptar la práctica como correcta y qué aspectos se dejan a la voluntad de los alumnos.

4. Ejemplo de ejecución

A continuación mostramos algunos ejemplos de ejecución. Como se puede observar, las órdenes que da el usuario *no* son sensibles a mayúsculas ni minúsculas (puedes escribirlas indistintamente). En las ejecuciones que siguen, el texto en negrita representa lo que el usuario de la aplicación introduce por teclado.

En la siguiente ejecución las blancas ponen una ficha y luego se intentan deshacer dos movimimientos (el último falla). Tras eso, se solicita la terminación.

```
|       |
|       |
|       |
|       |
|       |
|       |
+-------+
 1234567

Juegan blancas
Qué quieres hacer? poner
Introduce la columna: 3
|       |
|       |
|       |
|       |
|       |
```

```
|  O    |
+-------+
 1234567

Juegan negras
Qué quieres hacer? deshacer
|       |
|       |
|       |
|       |
|       |
|       |
+-------+
 1234567

Juegan blancas
Qué quieres hacer? deshacer
Imposible deshacer.
|       |
|       |
|       |
|       |
|       |
|       |
+-------+
 1234567

Juegan blancas
Qué quieres hacer? salir
```

En la siguiente ejecución, se hace un movimiento válido, otro inválido y después se reinicia la partida. En ese momento se intenta deshacer el reinicio, lo cual es imposible, y se termina.

```
|       |
|       |
|       |
|       |
|       |
|       |
+-------+
 1234567

Juegan blancas
Qué quieres hacer? poner
Introduce la columna: 4
|       |
|       |
|       |
|       |
|       |
|   O   |
+-------+
 1234567

Juegan negras
```

```
Qué quieres hacer? poner
Introduce la columna: 8
Movimiento incorrecto
|       |
|       |
|       |
|       |
|       |
|   O   |
+-------+
 1234567

Juegan negras
Qué quieres hacer? reiniciar
Partida reiniciada.
|       |
|       |
|       |
|       |
|       |
|       |
+-------+
 1234567

Juegan blancas
Qué quieres hacer? deshacer
Imposible deshacer.
|       |
|       |
|       |
|       |
|       |
|       |
+-------+
 1234567

Juegan blancas
Qué quieres hacer? salir
```

El último ejemplo muestra una partida en la que el primer jugador gana consiguiendo colocar cuatro fichas en vertical.

```
|       |
|       |
|       |
|       |
|       |
|       |
+-------+
 1234567

Juegan blancas
Qué quieres hacer? poner
Introduce la columna: 4
|       |
|       |
```

```
|       |
|       |
|       |
|   O   |
+-------+
 1234567

Juegan negras
Qué quieres hacer? poner
Introduce la columna: 3
|       |
|       |
|       |
|       |
|       |
|  XO   |
+-------+
 1234567

Juegan blancas
Qué quieres hacer? poner
Introduce la columna: 4
|       |
|       |
|       |
|       |
|   O   |
|  XO   |
+-------+
 1234567

Juegan negras
Qué quieres hacer? poner
Introduce la columna: 3
|       |
|       |
|       |
|       |
|  XO   |
|  XO   |
+-------+
 1234567

Juegan blancas
Qué quieres hacer? poner
Introduce la columna: 4
|       |
|       |
|       |
|   O   |
|  XO   |
|  XO   |
+-------+
 1234567

Juegan negras
```

```
Qué quieres hacer? poner
Introduce la columna: 3
|       |
|       |
|       |
|  XO   |
|  XO   |
|  XO   |
+-------+
 1234567

Juegan blancas
Qué quieres hacer? poner
Introduce la columna: 4
|       |
|       |
|   O   |
|  XO   |
|  XO   |
|  XO   |
+-------+
 1234567

Ganan las blancas
```

5. Entrega de la práctica

La práctica debe entregarse utilizando el mecanismo de entregas del campus virtual, no más tarde de la fecha y hora indicada en la cabecera de la práctica. El fichero debe tener al menos el siguiente contenido[1]:

- Directorio `src` con el código de todas las clases de la práctica.
- Fichero `alumnos.txt` donde se indicará el nombre de los componentes del grupo.

[1] Puedes incluir también opcionalmente los ficheros de información del proyecto de Eclipse