Universidad Nacional Autónoma de México

Facultad de Ciencias

Modelado y Programación

Proyecto II

Integrantes de equipo:

Hermes Alberto Delgado Díaz

319258613

Emiliano Sebastian Paredes Gonzalez

320047956

Kevin Steve Quezada Ordoñez

318331241

12 de mayo del 2024

Los Fósiles

Tres en raya

12/05/2024

Problematica:

La Secretaria de Educación Pública(SEP) quiere crear un juego para alumnos de educación básica. Su intención es que este juego ayude a mejorar cognitivamente a los alumnos de toda la república.

Para esto, la SEP lanzó una convocatoria donde puedes lanzar propuestas para el juego, además de participar en el desarrollo de este.

El equipo fósiles penso en un juego que se llama Tres en raya.

Explicación del juego

Tres en raya es un juego de estrategia para dos jugadores que se juega en un tablero de 3x3. El objetivo del juego es conseguir una fila, columna o diagonal de tres fichas del mismo jugador (ya sea X o O), o en su defecto, evitar que el oponente lo haga.

El juego comienza con un tablero vacío. Los jugadores alternan turnos, uno colocando fichas con la letra "Xz el otro con la letra .º", hasta que uno de los jugadores consigue formar una línea de tres fichas de su tipo o hasta que se llenan todas las casillas del tablero, lo que resulta en un empate.

Las reglas básicas son simples:

1. Los jugadores se turnan para colocar sus fichas en el tablero, una ficha por turno.
2. No se puede colocar una ficha en una casilla que ya esté ocupada.
3. Gana el primer jugador que consiga formar una línea de tres fichas de su tipo, ya sea en horizontal, vertical o diagonal.
4. Si todas las casillas del tablero están ocupadas y ningún jugador ha conseguido formar una línea de tres fichas, el juego termina en empate.

El tres en raya es un juego simple pero puede volverse estratégico con la práctica, ya que los jugadores deben anticipar los movimientos del oponente y planificar sus propias jugadas para ganar o bloquear al otro jugador.

División en paquetes

El proyecto se dividio en paquetes, cuatro principales y dentro de estos subpaquetes que ayudan a organizar de mejor manera las clases y tener un mejor entendimiento:

- ***Colors***

  El siguiente paquete contiene la clase Colors que da estilo a la vista.

- ***Modelo***

  El paquete modelo contiene un subpaquete llamado Jugador además de las clases Sujeto, Tablero, Tablero1v1, Tablero1vPC.

  - ***Jugador***

    El paquete jugador un subpaquete llamado Dificultad además contiene las clases Jugador, JugadorX, JugadorO y JugadorPC.

    - ***Dificultad***

      El paquete dificultad contiene las clases Dificultad, Aleatorio, Pensante.

- ***Vista***

  El paquete vista contiene las clases Observador, InicioVista, TableroVista, TableroVistaPC

- ***Controlador***

  El paquete controlador contiene un subpaquete llamado command además de las clases Controlador1v1, Controlador1vPC, ControladorInicio

  - ***Command***

    El paquete command contiene las clases Command y MovimientoCommand

Patrones de Diseño

- ***MVC***

  La implementación de modelo se utilizo en las clases Tablero, Tablero1v1 y Tablero1vPC.

  La implementación de vista se utilizo en las clases InicioVista, TableroVista y TableroVistaPC.

  La implemetación de controlador se utilizo en las clases ControladorInicio, Controlador1v1 y Controlador1vPC.

- ***Template Method***

  El patrón Template Method siver para extender pasos particulares de algún algoritmo, pero no todo en concreto o la estructura. En este caso la clase Jugador tiene un método realizarMovimiento y las subclases JugadorX, JugadorO, JugadorPC lo implementas cada una a su manera. Además se utilizo ya que el algoritmo es casi identico, solo tiene pequeñas diferencias entre si. El patrón Template Method fue investigado por cuenta propia, nos basamos en la página web llamada *'Refactoring Guru'*[5] además de el repositorio de github [6]

- ***Strategy***

  Este patrón se utilizó en las dificultades del jugadorPC, tenemos las dificultades Aleatoria y Pensante, las dos realizan el mismo movimiento, pero cada una a su propia manera.

- ***Observer***

  Este patrón se utilizó para poder comunicar las vistas con los modelos, cada vez que los modelos cambien, en este caso los tableros, las vistas mostraran el tablero modificado haciendo una actualización de este mismo.

- ***Command***

  Este patrón se utilizo ya que Command es un patrón de diseño de comportamiento que convierte una solicitud en un objeto independiente que contiene toda la información sobre la solicitud. En este caso utilizamos command para realizar los movimientos. De esta manera el controladorPC y controlador1v1 no saben como se realiza el movimiento y no lo necesitan saber, de eso se encarga la clase MovimientoCommand que con su método execute(), realizamos el movimiento del jugador además de notificaciones. La información de este patrón se saco dela página web llamada *'Refactoring Guru'*[4] además de el repositorio de github [6]

Implementación de Inteligencia Artificial en dificultad en modo 1vsPC

El jugadorPC realiza el movimiento buscando el mejor movimiento de todos los posibles a realizar, se utilizo el algoritmo minimax. Para entender el proceso de este algoritmo nos basamos en el video *'Simple Explanation of the Minimax Algorithm with Tic-Tac-Toe'* [1] y para poder implementarlo nos basamos en el video *'Tic-Tac-Toe AI Player using the Minimax Algorithm: A Step-By-Step Python Coding Tutorial'* [2]. Además buscamos información del libro *'Digital Notes of Artificial Intelligence'* [3].

Comandos de ejecución

- Para compilar el programa

```
user:~/src$ javac TresEnRaya.java
```

- Para ejecutar el programa

```
user:~/src$ java TresEnRaya
```

Referencias

[1] Sciencie Buddies. *Simple Explanation of the Minimax Algorithm with Tic-Tac-Toe*. Consultado en: 2024. 2024. URL: `https://www.youtube.com/watch?v=5y2a0Zhgq0U`.

[2] Sciencie Buddies. *Tic-Tac-Toe AI Player using the Minimax Algorithm: A Step-By-Step Python Coding Tutorial*. Consultado en: 2024. 2024. URL: `https://www.youtube.com/watch?v=JWRxdoPrk7M`.

[3] SRI INDU COLLEGE OF ENGINEERING y TECHNOLOGY. *DIGITAL NOTES ON ARTIFICIAL INTELLIGENCE*. Autonomous Institution – UGC, Govt. of India, 2023.

[4] Refactoring Guru. *Command*. Consultado en: 2024. 2024. URL: `https://refactoring.guru/es/design-patterns/command/java/example`.

[5] Refactoring Guru. *Template Method*. Consultado en: 2024. 2024. URL: `https://refactoring.guru/es/design-patterns/template-method`.

[6] Mkejeiri. *Java design patterns*. Consultado en: 2024. 2020. URL: `https://github.com/mkejeiri/Java-Design-Pattern`.