

INFORME UT5 Problema3. DAVID GARCÍA VALERO

Mi proceso de desarrollo, junto a los distintos retos que he enfrentado, ha sido:

1. Implementar la clase Coordinada:

La tarea más compleja fue crear el método `recollir()` el cual no entrañaba mucho misterio, simplemente validé fila y columna y después validé que estuviese dentro de los límites de la coordenada.

2. Implementar la clase Turno:

La parte más complicada de la clase turno es crear un número aleatorio del 0 al 1. Ya lo hemos hecho anteriormente, creé un `double Math.random`, si es mayor o igual a 0.5 el valor se cambia a 1 y en caso contrario a 0, luego este valor se pasa a `int` y se devuelve la variable al valor `this.valor`.

3. Implementar la clase Tabla:

A la hora de implementar esta clase, lo más difícil creo que fue comprender qué era un `enum` y saber cómo asignarle XS, OS o BUIDA a un `EstatCasella`.

A parte de esto, el método más difícil fue `hihaTresEnRatlla`, porque tuve que pensar muy bien cómo establecer la matriz multidimensional para que funcionase correctamente.

Teniendo 3 instancias de `i` y una instancia de `j` tengo al alcance toda la matriz utilizando solo una instancia para asignar el valor al `boolean`.

La siguiente parte fue ir estableciendo la posición de las `js` e `ies` para que mostrasen una posición u otra, según si quería saber si era tres en ralla en línea horizontal, vertical o diagonal.

Para complementar al juego en sí, creé una clase adicional `mostrarTabla()` la cual imprime el estado actual de la tabla por la consola.

4. Implementar la clase Jugador:

Para implementar esta clase, simplemente he tenido que utilizar métodos ya creados en la clase `tabla` para poder realizar los nuevos métodos de la clase `jugador`.

5. Implementar la clase principal TresEnRalla:

Utilizo varios métodos (no todos) de las distintas clases para establecer correctamente el juego, nuestro objetivo principal en el desarrollo.

A modo de resumen `jugar()`:

El juego empieza dando la bienvenida a los jugadores, tras esto, asigna a `jugadores[0]`, XS y a `jugadores[1]` OS y crea una nueva `Coordinada` llamada `coordinada`.

Entonces, empieza el bucle `while` principal del juego, el cual se repetirá siempre que no haya tres en ralla. El bucle empieza mostrando un mensaje por pantalla (Estado partida:) y muestra la tabla por la consola (método no propuesto que he creado para facilitar el desarrollo y hacer el juego más dinámico).

A continuación, (dos `IFs`) si el turno asignado aleatoriamente del 1 al 0 es 1 le tocará al jugador con las XS, en caso contrario al jugador con las OS.

El siguiente proceso es exactamente igual en los dos `IFs`:

Imprime por pantalla "Turno de (jugador)", establece nuestra Coordenada coordenada como el método que recoge los datos fila y columna para crear una coordenada (recollir), entonces un bucle while confirma que la coordenada no esté ocupada (método de tabla estaOcupada()), si no está ocupada, pone la ficha del jugador mediante el método posarFitxa de tabla. Por último cambia el valor de turno (método canviar de turno)

Una vez terminan estos dos IFs, siguiendo en el bucle while principal, establezco un if que determina si hay tres en ralla (método hihaTresEnRatlla) y dentro de este creo otros dos if para que si el que tiene tres en ralla es un jugador u otro, que cante victoria uno u otro (método cantarVictoria de jugador) y finalmente muestra la tabla de nuevo.

En el último if del bucle principal del juego compruebo si está llena la tabla (método estaPle de tabla) en caso de que sea true, la tabla se vacía, de esta manera, comienza el turno del jugador al que le toque en el siguiente bucle.

SUGERENCIAS FINALES:

El programa podría contar con un modo para jugar contra la máquina, el cual tendría que reforzarse con un método que controle el comportamiento de la misma (dónde pone su casilla), para que sea un reto interesante, pero que a la vez no sea imposible ganar.