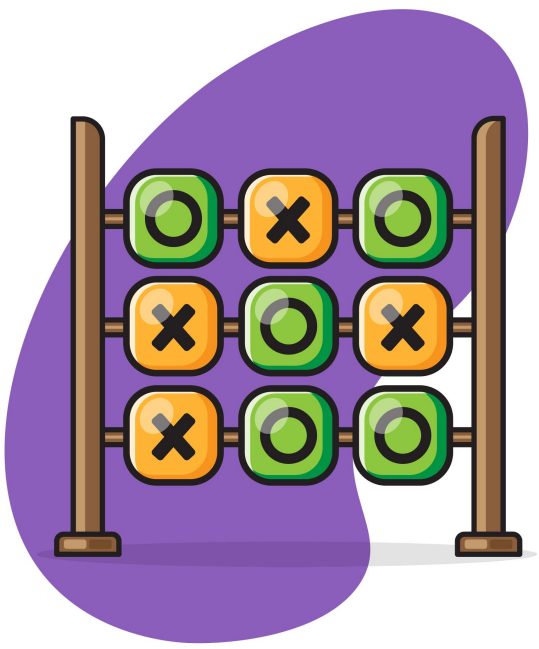
UT5
José R. Mas Davó authored 3 weeks ago

85519756

UT5Problema3.md 7.45 KiB

UT5 - Problema 3: Joc del Tres en Ratlla

Agrupament: Individual



Pregunta guia

Com podem implementar el joc del Tres en Ratlla utilitzant la Programació Orientada a Objectes (POO) en Java?

Objectius d'aprenentatge

En completar aquesta pràctica, sereu capaços de:

- Implementar classes utilitzant la Programació Orientada a Objectes (POO) en Java.
- Gestionar les interaccions entre objectes en un joc.
- Desenvolupar mètodes per a la gestió de l'estat del joc.
- Aplicar constructors, mètodes i atributs per a modelar les regles del joc.
- Desenvolupar la lògica de joc utilitzant estructures de control i algoritmes.
- Treballar amb enumerats i matrius bidimensionals.
- Implementar entrada/eixida bàsica d'usuari i validació de dades.
- Aplicar conceptes de modularització i reutilització de codi.

Context i descripció del problema

Una empresa de desenvolupament de software està treballant en una sèrie de jocs clàssics. Un dels projectes actuals és el desenvolupament del joc del Tres en Ratlla, un joc de tauler popular on dos jugadors competeixen per alinear tres de les seues fitxes en una quadrícula de 3x3. El projecte ja té algunes classes implementades, però necessitem completar les classes principals del joc per assegurar-ne el funcionament correcte.

Normes del joc:

1. El joc comença amb un tauler de 3x3 cel·les buides.
2. El jugador que comença la partida serà seleccionat aleatòriament.
3. No es pot posar una fitxa en una casella ja ocupada.
4. El joc acaba quan un jugador aconsegueix tres fitxes en línia (horitzontal, vertical o diagonal).
5. En cas que s'arribe a un empat, el tauler es netejarà i es tornarà a demanar que pose la fitxa el jugador al qual li toque el torn.

Implementació

El projecte es troba en el paquet `p53`, que conté els següents paquets:

- `p53.util`: Conté les classes `Interval` i `GestorIO`, que estan ja implementades i s'utilitzaran per a la gestió d'intervals i l'entrada/eixida de dades.
- `p53.joc.enums`: Inclou la classe `EstatCasella`, que defineix els possibles estats de les caselles del tauler (`BUIDA`, `X`, `O`).
- `p53.joc`: Aquest paquet conté les classes principals del joc que s'han d'implementar.

Classes a implementar:

1. **Classe Coordenada:**
   - **Atributs:**
     - `int fila`: fila de la casella.
     - `int columna`: columna de la casella.
     - `final Interval LIMITS`: interval d'1 a 3 que determina el rang vàlid de caselles.
   - **Constructors:**
     - Constructor buit que no inicialitza cap atribut.
     - Constructor complet que inicialitza els atributs `fila` i `columna`.
   - **Mètodes:**
     - `getFila()`, `getColumna()`: Retornen els valors de `fila` i `columna`.
     - `validar()`: Retorna `true` si `fila` i `columna` es troben dins de l'interval vàlid.
     - `recollir()`: Demana a l'usuari que introduïsca una fila i una columna, valida els valors i retorna una coordenada vàlida.
2. **Classe Torn:**
   - **Atributs:**
     - `int valor`: valor que determina el torn del jugador.
   - **Constructors:**
     - Constructor buit que inicialitza `valor` a un número aleatori entre 0 i 1.
   - **Mètodes:**
     - `toca()`: Retorna l'atribut `valor`.
     - `noToca()`: Retorna l'oposat de `valor`.
     - `canviar()`: Inverteix el valor de `valor`.
3. **Classe Tauler:**
   - **Atributs:**
     - `EstatCasella[][] caselles`: matriu de 3x3 que representa les caselles del tauler.
     - `final int DIMENSIO = 3`: dimensió del tauler.
   - **Constructors:**
     - Constructor buit que inicialitza la matriu `caselles` i crida al mètode `buidar`.
   - **Mètodes:**
     - `hiHaTresEnRatlla(EstatCasella jugador)`: Verifica si el jugador té tres en ratlla.
     - `hiHaTresEnRatlla()`: Comprova si hi ha tres en ratlla independentment del jugador.
     - `estaOcupada(Coordenada coordenada, EstatCasella jugador)`: Verifica si la casella està ocupada pel jugador.
     - `estaOcupada(Coordenada coordenada)`: Verifica si la casella està ocupada.
     - `posarFitxa(Coordenada coordenada, EstatCasella jugador)`: Assigna l'estat del jugador a la casella.
     - `buidar()`: Buida totes les caselles del tauler.
     - `estaPle()`: Retorna `true` si totes les caselles estan ocupades.
4. **Classe Jugador:**
   - **Atributs:**
     - `EstatCasella color`: color del jugador (X o O).
   - **Constructors:**

        - Constructor complet que inicialitza l'atribut `color`.

    - **Mètodes:**

        - `cantarVictoria()`: Mostra un missatge indicant que el jugador ha guanyat.
        - `errorPosada(Tauler tauler, Coordenada coordenada)`: En cas que el tauler estiga ocupat en la coordenada donada, retornarà un String "Coordenada ocupada en el tauler" i en cas contrari retornarà null.
        - `recollirCoordenadaPosadaValida(Tauler tauler)`: Recull una coordenada vàlida per posar la fitxa. Si en recollir-la és errònia (està ja ocupada) mostrarà un error i tornarà a recollir la Coordenada fins que no hi haja error. Finalment retornarà la Coordenada recollida.
        - `posarFitxa(Tauler tauler)`: Imprimirà per pantalla el missatge "Posa el jugador amb " i el color del jugador. Posteriorment recollirà i guardarà una Coordenada vàlida en el Tauler; i finalment posarà la fitxa en el Tauler

5. **Classe TresEnRatlla (classe principal)**

    - **Atributs:**

        - `Tauler tauler`: el tauler del joc.
        - `Jugador[] jugadors`: array de jugadors (X i O).
        - `Torn torn`: per determinar el jugador que té el torn.

    - **Constructors:**

        - Constructor buit que inicialitza els atributs.

    - **Mètodes:**

        - `jugar()`: Conté la lògica principal del joc seguint les normes.

            - *Pista:* haurà d'haver-hi una estructura iterativa que es repetisca mentre no es produïsca un tres en ratlla. Una vegada s'ix de l'estructura repetitiva s'haurà de cridar al mètode "cantarVictoria" del jugador que haja guanyat.

              Dins de l'estructura repetitiva el primer que farem és comprovar si el tauler està ple, i en eixe cas s'haurà de buidar el tauler. Posteriorment es mostrarà per pantalla el tauler. El jugador del torn que toca posarà una fitxa en el tauler, i després es canviarà de torn.

        - `public static void main(String[] args)`: Crea una instància de `TresEnRatlla` i inicia el joc.

Instruccions

1. **Investigació preliminar:**

    - Repassa les normes del joc del Tres en Ratlla.
    - Revisa els conceptes de POO, incloent-hi classes, atributs, mètodes i constructors.

2. **Desenvolupament del programa:**

    - Implementa les classes `Coordenada`, `Torn`, `Tauler`, `Jugador` i `TresEnRatlla` seguint les especificacions.
    - Assegura't que els mètodes implementats segueixen les normes del joc i la lògica de POO.

3. **Proves i depuració:**

    - Prova el programa amb diferents escenaris per assegurar-te que totes les funcionalitats del joc funcionen correctament.
    - Depura qualsevol error o comportament inesperat.

4. **Reflexió i documentació:**

    - Escriu un informe breu que incloga:
        - El procés de desenvolupament i les decisions preses.
        - Els reptes que has trobat i com els has resolt.
        - Suggeriments per a possibles millores o extensions del joc.

Entrega

- Envia el projecte de NetBeans exportat (`UT5Problema3.zip`) i l'informe en format PDF.
- Recorda seguir la convenció de noms.
- **Important:** Afig el teu nom i cognoms sempre com a comentari al principi dels fitxers JAVA (En NetBeans després de `@author`).