Nagy Házi Feladat
Dokumentáció

Dénes Lilla
N4JWEM

2021. December. 2

Specifikáció

Feladat: Masyu játék

A házi feladatként elkészített programom a Masyu nevű logikai játékot fogja megvalósítani.

A játékot egy négyzetrácsos pályán játsszák, melynek néhány mezőjén körök helyezkednek el. Ezek a körök fekete vagy fehér színűek lehetnek.

A cél az, hogy a felhasználó egy összefüggő hurkot rajzoljon, ami az összes kört tartalmazó mezőn áthalad. A vonal mindig az adott mező egyik oldalán lép be a mezőbe és egy másik oldalán távozik, tehát 90 fokos kanyarokra van lehetőség.

A körök színe meghatározza azt, hogy a vonalnak hogyan kell áthaladnia az adott mezőn:

- A fehér körökön egyenesen kell átmennie a vonalnak, és vagy az előtte vagy pedig az utána lévő mezőben kanyarodnia egyet
- A fekete köröknél kanyarodnia kell a vonalnak, és mind az előtte mind az utána következő mezőn egyenesen átmennie

Ha létrejön egy hurok ami az összes körrel rendelkező mezőn átmegy a szabályoknak megfelelően, akkor a feladvány sikeresen meg lett oldva.

A program működése

Főmenü

A program elindítása után egy főmenü jelenik meg, ahol a felhasználó a megfelelő gombra való kattintással ki tudja választani, hogy új játékot szeretne indítani, vagy pedig a legutóbb elmentett játékot folytatni.

Új játék

Új játék esetén a program betölt egyet a txt fájlban tárolt pályák közül. Az, hogy a pályák pontosan milyen formában kerülnek tárolásra a fájlban , a mentésről szóló részben olvasható részletesen.

A pályák 6x6-os méretűek.

Folytatás

Abban az esetben ha a felhasználó a folytatást választja, a program a legutóbb elmentett játékállást tölti be. A betöltés itt is txt fájlból történik.

Játék

Két cella közötti vonal húzására a választott két mezőre való kattintással van lehetőség.

Ha pedig olyankor kattintunk rá a két mezőre, amikor azok már össze vannak kötve, akkor az ezeket összekötő vonal törlődik.

Ha olyan irányban húzunk meg egy vonalat, ami megsérti valamelyik szabályt (pl. fehér körnél kanyarodik), akkor a program érzékeli, hogy a vonal hibás, és nem rajzolja be.

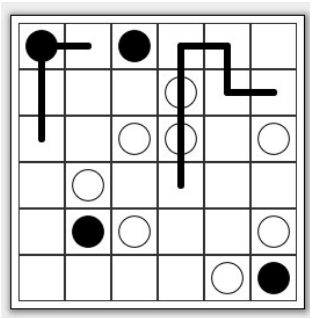
Amennyiben pedig a felhasználó sikeresen megoldotta feladványt, azt a program automatikusan érzékeli, és jelzi a felhasználó számára hogy a megoldás helyes.

Mentés

A játék közben a képernyőn látható egy menü sor, melyben található egy mentés gomb is, melyre kattintva a felhasználó el tudja menteni a játék jelenlegi állását.

A program ekkor elmenti a tábla elrendezését és a vonal állapotát két külön txt fájlba egy megadott formátumban.

Az alábbi állás esetén például:



A pálya a következőképpen kerül mentésre a txt fájlban:

X.X…

...O..

..OO.O

.O….

.XO..O

….OX

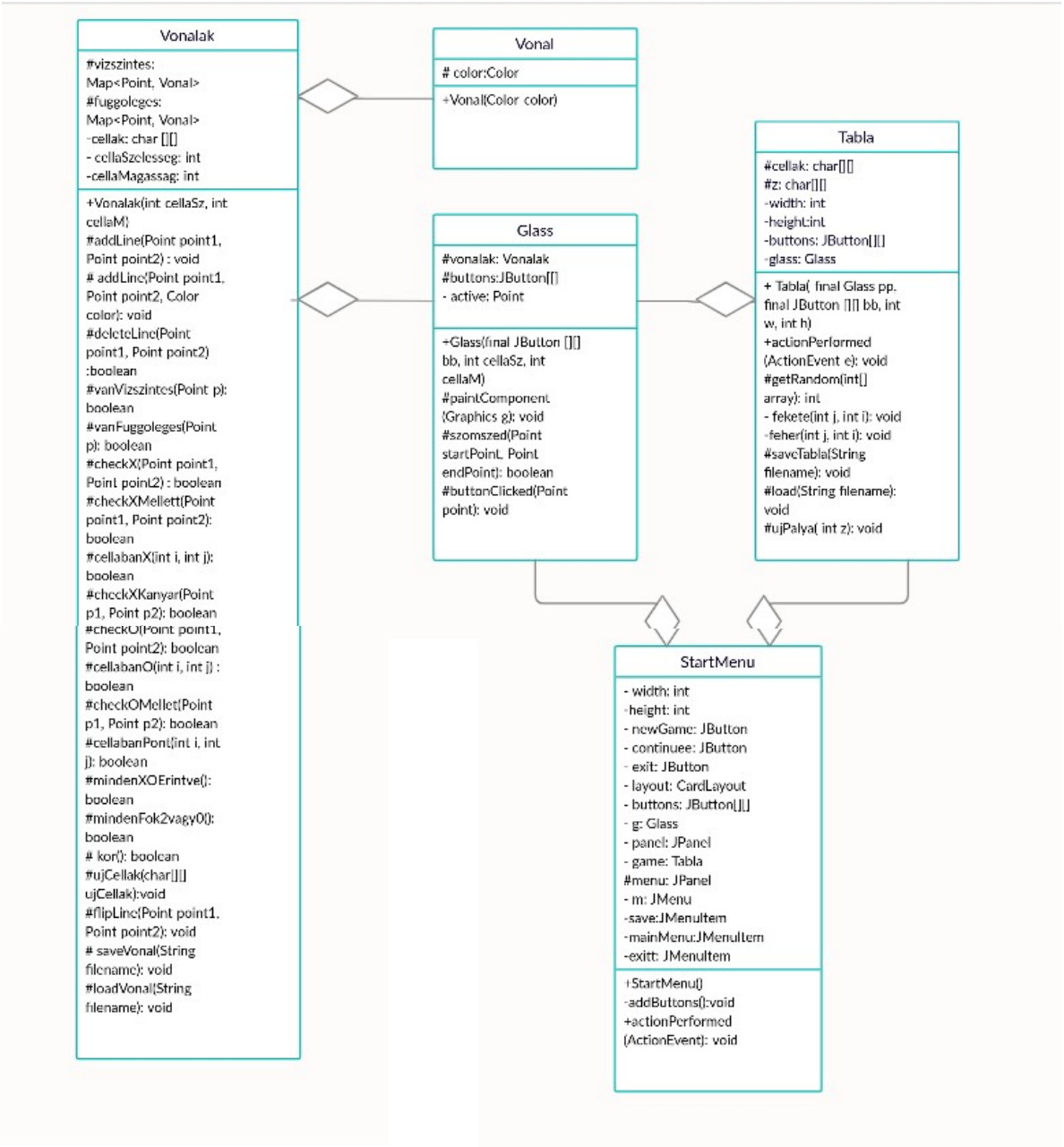
Tehát a . az üres mezőket jelöli, az O a fehér kört tartalmazó mezőket, az X pedig a fekete kört tartalmazó mezőket.

A vonal állapotának elmentéséhez pedig szerializáció segítségével történik.

Kilépés

A kilépés mezőre kattintva a felhasználó kilép a programból.
Amennyiben ezelőtt nem nyomta meg a mentés gombot, az kilépés előtti állás elveszik.

Osztálydiagram



Metódusok

StartMenu.java:

public StartMenu()
    Inicializálja a start menüt.
private void addButtons()
    Hozzáadja a gombokat a megfelelő panelhez.
public void actionPerformed(ActionEvent event)
    A játék működését vezérli a bizonyos gombokra való kattintás esetén.

Tabla.java:

public Tabla(final Glass pp, final JButton [][] bb, int w, int h)
    Incializálja a tábla panelt.
public void actionPerformed(ActionEvent e)
    Megadja, hogy mi történik egy mezőre való kattintáskor, ezen belül ellenőrződik az is, hogy helyesen meg lett e oldva a pálya.
protected int getRandom(int[] array)
    Visszaad egy random elemet a paraméterként kapott tömbből.
private void fekete(int j, int i)
    Az adott gomb hátterének beállítja a fekete kör képét.
private void feher(int j, int i)
    Az adott gomb hátterének beállítja a fehér kör képét.
protected void saveTabla(String filename)
    Elmenti a táblát.
protected void load(String filename)
    Betölti a legutóbb elmentett táblát.
public void ujPalya(int z)
    Fájlból betölt egy random új pályát.

Glass.java:

public Glass(final JButton [][] bb, int cellaSz, int cellaM)
    Inicializálja a glass panelt.
protected void paintComponent(Graphics g)
    Kirajzolja a vonalakat.
protected boolean szomszed(Point startPoint, Point endPoint)
    Megmondja, hogy a paraméterként kapott két pont szomszédos-e.

protected void buttonClicked(Point point)

Ha először kerül lenyomásra egy gomb, csak kijelöli melyik gomb van éppen lenyomva, ha már a második gomb van megnyomva, berajzolja, vagy pedig törli az adott vonalat, ha az megfelel a szabályoknak.

Vonal.java:

public Vonal(Color color)

Létrehoz egy vonalat.

Vonalak.java:

public Vonalak(int cellaSz, int cellaM)

Létrehoz egy vonalak komponenst.

protected void addLine(Point point1, Point point2, Color color)

Hozzáad egy vonalat.

protected boolean deleteLine(Point point1, Point point2)

Kitöröl egy vonalat.

protected boolean vanVizszintes(Point p)

Ellenőrzi, hogy megy-e át vízszintes vonal az adott ponton.

protected boolean vanFuggoleges(Point p)

Ellenőrzi, hogy megy-e át függőleges vonal az adott ponton.

protected boolean checkX(Point point1, Point point2)

Ellenőrzi, hogy a két adott pont közé rajzolt vonal megfelelne-e a fekete körön áthaladás szabályainak.

protected boolean checkXMellett(Point point1, Point point2)

Ellenőrzi, hogy a két adott pont közé rajzolt vonal megfelelne-e a fekete kör mellett lévő vonalak szabályainak.

protected boolean cellabanX(int i, int j)

Megadja, hogy az adott cellában van-e fekete kör.

protected boolean checkXKanyar(Point p1, Point p2)

Ellenőrzi, hogy a két adott pont közé rajzolt vonal megfelelne-e a fekete kör mellett lévő vonalak szabályainak.

protected boolean checkO(Point point1, Point point2)

Ellenőrzi, hogy a két adott pont közé rajzolt vonal megfelelne-e a fehér körön áthaladás szabályainak.

protected boolean cellabanO(int i, int j)

Megadja, hogy az adott cellában van-e fehér kör.

protected boolean checkOMellet(Point p1, Point p2)

Ellenőrzi, hogy a két adott pont közé rajzolt vonal megfelelne-e a fehér kör mellett lévő vonalak szabályainak.

protected boolean cellabanPont(int i, int j)

Megadja, hogy az adott cella üres-e.

protected boolean mindenXOErintve()

Ellenőrzi, hogy minden kört érint-e vonal.

protected boolean mindenFok2vagy0()

Ellenőrzi, hogy minden vonal foka 0 vagy 2.

protected boolean kor()

Ellenőrzi, hogy a vonalak egy összefüggő kört alkotnak-e.

protected void ujCellak(char[][] ujCellak)

Törli a mezők tartalmát.

protected void flipLine(Point point1, Point point2)

Ha van már itt vonal törli, ha nincs berajzolja, miután ellenőrizte, hogy megfelel-e a szabályoknak.

protected void saveVonal(String filename)

Elmenti a vonalakat.

protected void loadVonal(String filename)

Betölti a legutóbb elmentett vonalakat.