**1. Beadandó feladat dokumentáció**

Készítette: Börzsei Marcell László

Neptun-kód: BQHFLX

E-mail: [borzseimarcell032@gmail.com](mailto:borzseimarcell032@gmail.com)

**Feladat:**

**Kitolás**

Készítsünk programot, amellyel a következő két személyes játékot lehet játszani. Adott egy 𝑛 × 𝑛 mezőből álló tábla, amelyen kezdetben a játékosoknak 𝑛 fehér, illetve 𝑛 fekete kavics áll rendelkezésre, amelyek elhelyezkedése véletlenszerű. A játékosok kiválaszthat egy saját kavicsot, amelyet függőlegesen, vagy vízszintesen eltolhat. Eltoláskor azonban nem csak az adott kavics, hanem a vele az eltolás irányában szomszédos kavicsok is eltolódnak, a szélső mezőn lévők pedig lekerülnek a játéktábláról. A játék célja, hogy adott körszámon belül (5𝑛) az ellenfél minél több kavicsát letoljuk a pályáról (azaz nekünk maradjon több kavicsunk a végére). Ha mindkét játékosnak ugyanannyi marad, akkor a játék döntetlen. A program biztosítson lehetőséget új játék kezdésére a táblaméret (3 × 3, 4 × 4, 6 × 6) és így a lépésszám (15, 20, 30) megadásával, és ismerje fel, ha vége a játéknak. Ekkor jelenítse meg, melyik játékos győzött (ha nem döntetlen), majd automatikusan kezdjen új játékot. Ezen felül legyen lehetőség a játék elmentésére, valamint betöltésére.

**Elemzés:**

* A játékot három játékmóddal játszhatjuk: 3x3-as (15 lépés, 9 generált mező), 4x4-es (20 lépés, 16 előre generált mező), 6x6-os (30 lépés, 36 előre generált mező).
* A feladatot egyablakos asztali alkalmazásként Windows Forms grafikus felülettel valósítjuk meg
* Az ablakban elhelyezünk egy menüt a következő menüpontokkal: File (Save Game, Load Saved Game), illetve különböző vezérlő nyomógombokat, amellyel a játékos kiválaszthatja hányszor hányas játéktáblán szeretne játszani, majd a ’Ready’ gombra kattintva el is indíthatja a játékot, ezen felül pedig a játék végén lehetőség van a ’Restart’ gombra kattintva új kört játszani. A vezérlő nyomóbombok alatt megjelenítjük, a fehér és fekete kavicsok számát, amennyi még a játékosok birtokában van, illetve a hátralévő körök számát.
* A játéktáblát egy 3x3-as vagy 4x4-es vagy 6x6-os színes négyzetekből álló rács reprezentálja.
* Mindig a fekete játékos kezdi a játékot. A táblán egy piros színű szegély jelzi az éppen kijelölt kavicsot, amellyel lehet lépni, a kijelölést pedig a ’Space’ lenyomásával tudjuk változtatni. Majd, ha kiválasztottuk a megfelelő kavicsot, amellyel szeretnénk lépni, akkor a nyilak (gombok) lenyomásával jelezhetjük, hogy milyen irányba szeretnénk lépni.
* A játék automatikusan feldob egy dialógusablakot, amikor vége a játéknak (jelzi, hogy esetlegesen ki nyert vagy ha döntetlen a meccs). A mentést és a betöltést a File néven szereplő ’menustripben’ végezhetjük el.

A képen diagram, kör, vázlat, rajz látható

Automatikusan generált leírás

**Tervezés:**

* **Programszerkezet:**
* A programot háromrétegű architektúrában valósíjuk meg. A megjelenítés a View, a modell a Model, míg a perzisztencia a Persistence névtérben helyezkedik el.
* A program szerkezetét két projektre osztjuk implementációs megfontolásból: a Persistence és Model csomagok a program felületfüggetlen projektjében, míg a View csomag a Windows Formstól függő projektjében kap helyet.
* **Perzisztencia:**
* Az adatkezelés feladata a SlideTable táblával kapcsolatos információk tárolása, valamint a betöltés/mentés biztosítása.
* A SlideTable osztály egy érvényes táblát biztosít (azaz mindig ellenőrzi a beállított értékek), ahol minden mezőre ismert az értéke. Magukat az értékeket a tábla létrehozásakor véletlen generáljuk. A táblából alapértelmezés szerint 3x3-as, 4x4-es, illetve 6x6-os hozható létre, és ez a konstruktorban paraméterezhető. A tábla lehetőséget az állapotok lekérdezésére (a különböző fieldeknek a getterei), valamint szabályos léptetésre (slide).
* A hosszú távú adattárolás lehetőségeit az IKitolasDataAccess interfész adja meg, amely lehetőséget ad a tábla betöltésére (LoadAsync), valamint mentésére (SaveAsync). A műveleteket hatékonysági okokból aszinkron módon valósítjuk meg.
* Az interfészt szöveges fájl alapú adatkezelésre a KitolasFileDataAccess osztály valósítja meg. A fájlkezelés során fellépő hibákat a KitolasDataException kivétel jelzi.
* A program az adatokat szöveges fájlként tudja eltárolni, melyek az stl kiterjesztést kapják. Ezeket az adatokat a programban bármikor be lehet tölteni, illetve ki lehet menteni az aktuális állást.
* A fájl első sora megadja a tábla méretét, a hátralévő körök számát, a fekete majd a fehér kavicsok számát, illetve a soron következő játékost. A fájl többi része izomorf leképezése a játéktáblának, azaz összesen 3, 4 vagy 6 sor következik (játéktábla méretétől függően), és minden sor 3, 4 vagy 6számot tartalmaz szóközökkel választva. A számok 0, 1 vagy 2 lehetnek, ahol 0 reprezentálja az üres mezőt, 1 a fekete kavicsot és 2 a fehér kavicsot.

**A képen szöveg, képernyőkép, diagram, sor látható

Automatikusan generált leírás**

* **Modell:**
* A modell lényegi részét a Game osztály valósítja meg, amely szabályozza a tábla tevékenységeit, valamint a játék egyéb paramétereit, úgymint maga a játék létrehozása és a kiválasztott kavics számontartása (selectedStone). A típus lehetőséget ad új játék kezdésére (initializeMap), valamint lépésre (SlideMove). Új játéknál megadható a kiinduló játéktábla is, különben automatikusan generálódnak kezdő mezők.
* A mezők állapotváltozásáról a changedDisplay esemény tájékoztat.
* A játék végéről a gameEnd esemény tájékoztat. Az események argumentuma (isGameEndArgs) tárolja a győzelem állapotát, a győztest magát (ha van), illetve a lépések számát.
* A modell példányosításkor megkapja az adatkezelés felületét, amelynek segítségével lehetőséget ad betöltésre (LoadGameAsync) és mentésre (SaveGameAsync)
* A játékmódot a vezérlő gombok által átadott értékeken keresztül kezeljük és állítjuk be.
* **Nézet:**
* A nézetet a Form1 osztály biztosítja, amely tárolja a modell egy példányát (game), valamint az adatelérés konkrét példányát (\_dataAccess).
* A játéktáblát egy dinamikusan létrehozott pictureBox reprezentálja. A felületen létrehozzuk a megfelelő gombokat, vezérlőket, valamint feliratokat, és a hozzájuk tartozó eseménykezelőket. A játéktábla generálását a displayButton ’Click’ eseménye fogja kiváltani, majd a (DrawGameTable) metódus megvalósítani vizuálisan.
* A változások konstans megjelenítéséért és frissítéséért az (UpdatePictureBox) felel.

**Tesztelés:**

* A modell funkcionalitása egységtesztek segítségével lett ellenőrizve a SudokuGameModelTest osztályban.
* Az alábbi tesztesetek kerültek megvalósításra:
  + TestContructor: Új játék indítása, és a SlideTable objektum adattagjainak helyességének ellenőrzése.
  + TestMatrixStonePlaces: A mátrixban a kavicsok helyének ellenőrzése.
  + TestSlide: A SlideTable slide metódusának helyességének ellenőrzése valid illetve invalid lépésekkel.
  + TestInvalidSteps: A slide metódus nem megfelelő (ha a játékos lelépne a tábláról) lépéseinek ellenőrzése.
  + TestChangeSelectedStone: A program kavics léptető funkcionalitásást vizsgáló teszt.
  + TestGameEndings: A program játék végét ellenőrző metódúsát és eseményét ellenőrző teszt.

**A képen szöveg, Párhuzamos, diagram, képernyőkép látható

Automatikusan generált leírás**