**Készítette**:

Újvári Marcell

G7W8FM

**Feladat:**

Készítsünk programot, amellyel a következő két személyes játékot játszhatjuk. A malom játékban két játékos egy 24 mezőből álló speciális játéktáblán játszhatja, ahol a mezők három egymásba helyezett négyzetben helyezkednek (mindegyikben 8, a sarkoknál és a felezőpontoknál), melyek a felezőpontok mentén össze vannak kötve. Kezdetben a tábla üres, és felváltva helyezhetik el rajta bábuikat az üres mezőkre. Az elhelyezés után a játékosok felváltva mozgathatják bábuikat a szomszédos (összekötött) mezőkre. Amennyiben egy játékos nem tud mozgatni, akkor passzolhat a másik játékosnak. Ha valakinek sikerül 3 egymás melletti mezőt elfoglalnia (azaz malmot alakít ki, rakodás, vagy mozgatás közben), akkor leveheti az ellenfél egy általa megjelölt bábuját (kivéve, ha az egy malom része). Az a játékos veszít, akinek először megy 3 alá a bábuk száma a mozgatási fázis alatt.

A program biztosítson lehetőséget új játék kezdésére, mentésére és betöltésére. Ismerje fel, ha vége a játéknak, és jelenítse meg, melyik játékos győzött.

**Elemzés:**

* A játéknak egy állandó méretű játéktáblája van, amire a bábukat lehet elhelyezni. A program indításkor automatikusan egy új játékot kezd.
* A feladatot egyablakos asztali alkalmazásként Windows Forms grafikus felülettel valósítjuk meg.
* Az ablakban egy menü található a következő menüpontokkal: File (New Game, Save Game, Load Game). Az ablak alján megjelenik az éppen soron következő játékos és a tőle elvárt lépés.
* A játéktábla 24 mezőből áll, amit három egymásba helyezett négyzet reprezentál. A mezők a felezőpontok mentén össze vannak kötve.
* A játék automatikusan jelzi a játék végét és kiírja a nyertest. A mentés és betöltés is dialógusablakokkal történik.

**Diagram

Description automatically generatedFelhasználói eset:**

**Tervezés:**

* Programszerkezet:
  + A programnak háromrétegű az architektúrája, View, Model és Persistence.
  + A programot két részre osztjuk: A Persistence és Model csomagok egy felületfüggetlen projektben, míg a View egy Windows Formsos projektben található.
* Perzisztencia:
  + A perzisztencia biztosítja a játék betöltését és mentését, valamint a Malom táblájával kapcsolatos információkat tárolja.
  + A MalomTable osztály egy táblát biztosít. Itt tároljuk minden mező értékét (\_fieldOuterValues, \_fieldMiddleValues, \_fieldInnerValues). A tábla biztosít lehetőséget az állapotok lekérdezésére (IsFilled, CurrentPlayer, GetValue, CurrentNumberOfPieces, Player1NumberOfPieces, Player2NumberOfPieces), valamint az értékek szabályos léptetésére (StepValue) és direkt beállítására (SetValue, SetEmpty)
  + Az adatok tárolásának lehetőségeit az IMalomDataAccess interfész teremti meg. Lehetőséget ad a játék betöltésére (LoadAsync) és mentésére (SaveAsync).
  + A szöveges fájlok kezelését a MalomFileDataAccess végzi el. A fájlkezelés során felmerülő hibákat a MalomDataException jelzi.
  + A program szöveges fájlként tárolja az adatokat.
  + A fájl első sora a jelenlegi játékost jelzi. A második sorban van a lépések száma, a még kézben lévő bábuk, illetve az első és második játékos még meglévő bábui. A többi sorban a mezők értékei találhatóak.
  + A mezők lehetséges értékeit egy felsorolási típusban tároljuk (Values).

Diagram

Description automatically generated

* Modell:
  + A modellt a MalomGameModel osztály valósítja meg. Ez szabályozza a tábla működését, valamint tárolja a játék jelenlegi állapotát (\_gameState). Lehetőséget ad új játék kezdésére (NewGame), bábuk letételére (Step), bábuk mozgatására (MovePiece), malom ellenőrzésére (CheckMill), malom esetén ellenőrzött módon bábu levételére (Mill).
  + A következő kört, a játék végét, malom létrejöttét és a mozgatást esemény jelzi (GameAdvanced, OnMill, GameOver, Move) a nézet felé.
  + A MalomEventArgs argumentum tárolja a jelenlegi játékost, a nyertest, és a játék állapotát. A MoveEventArgs argumentum pedig a mozgatni kívánt bábu jelenlegi helyét és mozgatás utáni helyét tartalmazza a jelenlegi játékos mellett.
  + A modell megkapja az adatkezeléshez szükséges felületet (\_dataAccess) ami alapján mentést (SaveGameAsync) és betöltést (LoadGameAsync) biztosít.
* Nézet:
  + A nézetet a GameForm osztály biztosítja. Ez tárolja a modellt (\_model) és az adatelérés példányát (\_dataAccess).
  + A játéktáblát egy dinamikusan létrehozott gombmezők (\_outerButtons, \_middleButtons, \_innerButtons) és egy elemi grafikával készített tábla reprezentálja. Az ablakban elhelyezkednek a megfelelő menüpontok illetve dialógusablakok és a hozzájuk tartozó eseménykezelők. A játéktábla generálását (GenerateTable, PaintTable) metódusok végzik.

**UML diagram:**

**Diagram

Description automatically generated with medium confidence**

**Tesztelés:**

* A modell működése egységtesztek segítségével lett tesztelve a MalomGameModelTest osztályban:
  + MalomGameModelNewGameTest
    - Új játék indításának tesztelése
  + MalomGameModelNotIsGameOverTest,
  + MalomGameModelIsGameOverTest:
    - Annak tesztelése, hogy vége van-e a játéknak
  + MalomGameModelCanMoveTest
    - Annak tesztelése, hogy a játékos tud-e valahova lépni a játéktáblán
  + MalomGameModelStepExceptionTest,
  + MalomGameModelStepTest:
    - Játékbeli lépés tesztelése
  + MalomChangeGameTest:
    - Az utolsó lépés után észleli-e a modell, hogy mostantól lépnie kell a játékosoknak
  + MalomGameModelMillPlaceTest,
  + MalomGameModelRemFromMillTest,
  + MalomGameModelRemWrongPieceTest,
    - Malom kialakulásának, illetve malom után bábu levételének tesztelése
  + MalomGameModelMoveToOccupiedTest,
  + MalomGameModelMoveTest,
  + MalomGameModelMoveToNonAdjacentTileTest
    - Mozgatás tesztelése
  + MalomGameModelLoadTest
    - Játéktábla betöltésének tesztelése mockolt perzisztencia réteggel