1. **Beadandó feladat dokumentáció**

**Készítette:**

Varga Lili Anna

E-mail: [vargalili012@gmail.com](mailto:vargalili012@gmail.com)

**Feladat:**

Készítsünk programot, amellyel az aszteroidák játékot játszhatjuk. A feladatunk az, hogy egy űrhajó segítségével átnavigáljuk egy aszteroidamezőn. Az űrhajóval a képernyő alsó sorában tudunk balra, illetve jobbra navigálni. A képernyő felső sorában meghatározott időközönként véletlenszerű pozícióban jelennek meg az aszteroidák, amelyek folyamatosan közelednek állandó sebességgel a képernyő alja felé. Az idő múlásával egyre több aszteroida jelenik meg egyszerre, így idővel elkerülhetetlenné válik az ütközés. A játék célja az, hogy az űrhajó minél tovább elkerülje az ütközést. A program biztosítson lehetőséget új játék kezdésére, valamint játék szüneteltetésére (ekkor nem telik az idő, és nem mozog semmi a játékban). Ismerje fel, ha vége a játéknak, és jelenítse meg, mennyi volt a játékidő. Ezen felül szüneteltetés alatt legyen lehetőség a játék elmentésére, valamint betöltésére.

**Elemzés:**

* A program indításakor egy menü jelenik meg, melyben új játékot indíthatunk, vagy kiléphetünk a programkor.
* A játék automatikusan feldob egy dialógusablakot, amikor vége a játéknak (ha ütköztünk egy aszteroidával).
* Egy új ablakban elhelyezünk egy menüt a következő menüpontokkal: Játék betöltése, Játék mentése.
* A játéktábla egy új ablak, melyen idővel folyamatosan növekvő aszteroidák jelennek meg. A játékos fel, le, balra és jobbra tud mozogni, hogy elkerülje az aszteroidákat, a pálya határain belül.
* A képen képernyőkép, kör, szöveg, fekete-fehér látható

  Automatikusan generált leírásA felhasználói esetek az 1. ábrán láthatóak.

**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 csomagszerkezete a 2. ábrán látható.
  + 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 Játék pályájával kapcsolatos információk tárolása, valamint a betöltés/mentés biztosítása.
  + A Game.cs egy olyan osztály, amely játék aktuális állapotát tárolja. Ebben szerepel a játékos helyzete, az eddig eltelt idő és az aszteroidák pozíciója.
  + A hosszú távú adattárolás lehetőségeit az IPersistence interfész adja meg, amely lehetőséget ad a tábla betöltésére (Load), valamint mentésére (Save).
  + Az interfészt szöveges fájl alapú adatkezelésre a TextFilePersistence osztály valósítja meg. A fájlkezelés során fellépő hibákat a DataException kivétel jelzi.
  + A program az adatokat szöveges fájlként tudja eltárolni. Ezeket az adatokat a programban bármikor be lehet tölteni, illetve ki lehet menteni az aktuális állást.
  + A képen képernyőkép, fekete látható

    Automatikusan generált leírásA fájl első két eleme megadja a játékos pozícióját x és y koordináták szerint, a harmadik eleme az eddig eltelt időt tárolja el. A fájl többi része az aszteroidák helyzetét menti el.
* Model
  + A modell lényegi részét a GameModel osztály valósítja meg, amely szabályozza a játék pálya tevékenységeit, valamint egyéb paramétereit, mint például az idő (GameTime) és a játékos (player) pozíciója, illetve az aszteroidák kezelése (asteroids). Az osztály lehetőséget biztosít a játék előrehaladására (AdvanceGame), az aszteroidák mozgásának kezelésére (MoveAsteroid), és új aszteroidák létrehozására (CreateAsteroid).
  + A játékos pozícióját mozgási metódusok (MovePlayerLeft, MovePlayerRight, MovePlayerUp, MovePlayerDown) szabályozzák, amely lehetőséget biztosít a játékos különböző irányokba történő mozgatására a panel határain belül.
  + A játék állapotváltozásairól az AsteroidCreated, AsteroidMoved, AsteroidReMoved események tájékoztatják a felhasználót. Az események argumentumai információkat tartalmaznak az újonnan létrehozott aszteroidákról, illetve a már meglévők pozíciójának változásairól. A játék végéről a GameEnded esemény tájékoztat, amikor egy ütközés történik.
  + A Player osztály tárolja a játékos aktuális pozícióját (Position), és lehetőséget biztosít a játékos mozgatására a képernyő bal, jobb, felső, illetve alsó határain belül. A játékos alapértelmezett mozgási sebessége (speed) 10 egység, ami minden irányban módosítja a játékos pozícióját.
  + Az ütközéseket a CollidesWith metódus kezeli, amely figyeli a játékos és az aszteroidák közötti átfedést. Ha az X és Y koordináták alapján a játékos és egy aszteroida ütköznek, a metódus igaz értéket ad vissza, amely a GameEnded esemény kiváltásához vezet, ezzel véget vetve a játéknak.
  + A játékállapot mentését és betöltését a LoadGame és SaveGame metódusok kezelik, amelyeket az adatkezelés felülete, az IPersistence interfészen keresztül valósít meg. Az IPersistence interfész lehetőséget ad a játékadatok betöltésére (Load) és mentésére (Save). A LoadGame a játékos adatait, az aszteroidák listáját és a játékidőt tölti be a megadott elérési útvonalról, míg a SaveGame elmenti ezeket az adatokat.
* Nézet
  + A nézetet a MainGame osztály biztosítja, amely tárolja a játék modell egy példányát (model) és kezeli a játékfelület grafikus elemeit. A GamePanel elem felelős a játékpanel megjelenítéséért, míg a playerPanel és az asteroids listája az egyes játékobjektumok (a játékos és az aszteroidák) vizuális megjelenítését biztosítja.
  + Az eseménykezelést a gameTimer időzítő segíti, amely szabályos időközönként (100 ms) lépteti előre a játékot (GameTimer\_Tick esemény). A játék előrehaladtával az aszteroidák mozgatását és a játékos pozícióváltozásait a billentyűkezelő (MainGame\_KeyDown) és a modell eseményei (AsteroidCreated, AsteroidMoved, AsteroidReMoved, GameEnded) irányítják.
  + Az aszteroidák megjelenítése és kezelése a modell AsteroidCreated, AsteroidMoved és AsteroidReMoved eseményei alapján történik. Az AsteroidCreated esemény létrehozza az aszteroida panel elemeket, míg az AsteroidMoved frissíti az egyes aszteroidák pozícióját. Amennyiben egy aszteroida elhagyja a képernyőt, a Model\_AsteroidReMoved metódus törli azt a képernyőről.
  + A játék felfüggesztésére és folytatására a PauseMenu biztosít lehetőséget, amely a játék Escape billentyű megnyomásával érhető el. A szünet menü (PauseMenu) felülete külön mentés és betöltés funkciókat biztosít, amelyek lehetővé teszik a játékállapot fájlba mentését és későbbi betöltését.
  + A MenuForm osztály kezeli a főmenü megjelenítését és a játék indulásának irányítását, ahol új játékot indíthatunk (NewGame\_Click) vagy kiléphetünk a programból (ExitButton\_Click).

A képen szöveg, képernyőkép, Betűtípus, tervezés látható

Automatikusan generált leírás

* Tesztelés
  + A játékmodell funkcionalitása egységtesztekkel került ellenőrzésre a GameModelTests osztályban, ahol különböző játékfunkciók helyes működése és a perzisztencia tesztelése történt meg.
  + Az alábbi tesztesetek lettek megvalósítva:
    - AdvanceGame\_CreatesAsteroids: A teszt során a AdvanceGame metódust ismételten meghívva biztosítottuk, hogy aszteroidák jönnek létre a játék során, és azok száma nő.
    - MoveAsteroid: A teszt célja annak ellenőrzése, hogy az MoveAsteroid metódus helyesen mozgatja az aszteroidákat. Amennyiben van aszteroida, akkor a metódus az y-koordinátájukat növeli, miközben az x-koordinátát változatlanul hagyja.
    - GameEndsWhenCollided: Ez a teszteset igazolja, hogy a játék véget ér, ha a játékos és egy aszteroida ütközik. Az aszteroidát a játékos pozíciójába helyezve teszteljük, hogy az GameEnded esemény helyesen kiváltódik.
    - Boundaries: A teszt a játékos mozgási korlátait ellenőrzi. A játékos nem hagyhatja el a képernyő bal, jobb, felső vagy alsó széleit. A teszt során a játékos pozícióját minden irányba addig mozgatjuk, amíg el nem éri a játékterület határait.
    - PersistenceSave: A perzisztencia tesztelése érdekében ellenőrizzük, hogy a SaveGame metódus meghívásra kerül-e a perzisztencia rétegen keresztül a megadott mentési útvonallal.
    - PersistenceLoad: A játék betöltésének tesztje, amely mockolt perzisztencia réteggel történik. A LoadGame meghívása után ellenőrizzük, hogy a játékos, az aszteroidák és a játékidő helyesen kerültek-e visszatöltésre, valamint, hogy a betöltésért felelős perzisztencia metódus meghívódik-e.