**Közös követelmények:**

* A megvalósításnak, felhasználóbarátnak és könnyen kezelhetőnek kell lennie. Törekedni kell az objektumorientált megoldásra, de nem kötelező a többrétegű architektúra alkalmazása.
* A megjelenítéshez első sorban elemi grafikát kell használni. Az így kirajzolt ’sprite’-ok közül a játékoshoz tartozót billentyűzet segítségével lehessen mozgatni a jelenleg is standard (WASD) billentyűzet kiosztásnak megfelelően. Egyéb funkciókhoz egérhez kapcsolódó esemény vezérlőket is implementálhattok.
* Amennyiben nem algoritmussal generáltatod a játékteret, úgy legalább 10 előre definiált játékteret készíts különböző fájlokban eltárolva. Ügyelj arra, hogy mind az algoritmussal generált játékok esetén, illetve az előre definiált esetekben is végig játszható legyen az adott terület.
* Minden feladathoz tartozik egy időzítő, mely a játék kezdetétől eltelt időt mutatja.
* A dokumentációnak tartalmaznia kell a választott feladat leírását, elemzését, a program szerkezetének leírását (UML osztálydiagrammal), egy implementációs fejezetet a kiválasztott játék szempontjából és/vagy az általad érdekesebbnek gondolt algoritmusok leírásával. (Például: pálya generáláshoz implementált algoritmus.), valamint az eseményeseménykezelő párosításokat és a tevékenység rövid leírását.
* A feladatleírás a minimális követelményeket tartalmazza. A játékok tetszőlegesen bővíthetők.

**Játékszöveg leírás (mini RPG Adventure)**

Egy kalandort irányítva kell a nyirkos kazamatában összegyűjteni a kincseket. A feladat nem ennyire egyszerű, a kalandor dolgát megnehezítik a sötétben járkáló goblinok. Amennyiben a kalandor összefut egy szörnnyel ott leli a végét. A kalandozás során, számon tartjuk, hogy mennyi kincset sikerült a kalandornak összeszedni. Ez egész nagy számokig is elmehet, úgyanis egy kazamata kamrában nem lehet megmondani mennyi kincs lapul és mindig csak egy fedi fel magát. Amint a kalandor felveszi a kincset, egy újabb jelenik meg valahol a kamrában. Abban az esetben, ha elkapnak minket a goblinok, megjelenik egy felugró ablak, melyben a kalandor nevét megadva el tudjuk menteni az aktuális eredményünket az adatbázisba (Név, Nehézség, Összegyűjtött Kincs, Idő). Legyen egy menüpont, ahol a 10 legjobb eredménnyel rendelkező játékost lehet megtekinteni.

**Terv**

Diagram, schematic

Description automatically generated

A MainWindow osztály egy származtatott JFrame osztály, ahol a játék jelentős vizuális része történik. Konstruktorában meg van adva az ablak neve és mérte, valamint a program ikonja. Ezek mellett van egy legördülő menü, ahol kiválaszthatjuk a játék nehézségét, megnézhetjük a játékosok rekordjait, valamint van egy Exit is. A MainWindow létrehoz egy Game osztályt, ahol a backend mágia történik, valamint egy Boardot, ami JPanelből lett származtatva.

A Board osztály jeleníti meg az adott pályát. A Game-nek van egy GameLevel meg egy GameID osztálya. A GameID-ban tárolt adatok szerint a Game a GameLevelen végigmenve létrehozza a pályát az adott Block-okból. Egy pályát egy 25x25-ös mátrixból konstruál, ahol a mátrix külső sorai a falak, a többi meg a kazamata badlója. 5 különböző nehézség van: Easy, Medium, Hard, Expert és Legendary. Ezek közt nem sok különbség van, csak annyi, hogy más-más számú ellenség kerül az adott pályára (4, 5, 7, 10, 15).

A játékos a pálya közepére spawnol. A játék még random sorsol pozíciókat a szörnyeknek és a kincsnek. Amint a játékos megmozdul elindul egy timer és a szörnyek is lépnek egyet másodpercenként egy random generált irányba. Ha a szörnyek beleütköznek a falba, megindulnak visszafele. Egymásba nem ütköznek. Ha a játékos fut beléjük, akkor véget ér a játék. Ha a játékos olyan mezőre lép, amin kincs van, akkor fel tudja szedni azt.

**Tesztelési Terv**

Graphical user interface

Description automatically generated with low confidence