

# Maci Laci

## Feladatléírás

A meséből jól ismert Maci Laci bőrébe bújva a Yellowstone Nemzeti Park megmászhatatlan hegyei és fái között szeretnénk begyűjteni az összes rendelkezésre álló piknik kosarat. Az átjárhatatlan akadályok mellett Yogi élelem szerzését vadőrök nehezítik, akik vízszintesen vagy függőlegesen járőröznek a parkban. Amennyiben Yogi egy egység távolságon belül a vadőr látószögébe kerül, úgy elveszít egy élet pontot. (Az egység meghatározása rád van bízva, de legalább a Yogi sprite-od szélessége legyen.) Ha a 3 élet pontja még nem fogyott el, úgy a park bejáratához kerül, ahonnan indult.

A kalandozás során, számon tartjuk, hogy hány piknik kosarat sikerült összegyűjtenie Lacinak. Amennyiben egy pályán sikerül összegyűjteni az összes kosarat, úgy töltünk be, vagy generálunk egy új játékteret. Abban az esetben, ha elveszítjük a 3 élet pontunkat, úgy jelenjen meg egy felugró ablak, melyben a nevüket megadva el tudják menteni az aktuális eredményüket az adatbázisba. Legyen egy menüpont, ahol a 10 legjobb eredménnyel rendelkező játékost lehet megtekinteni, az elért pontszámukkal, továbbá lehessen bármikor új játékot indítani egy másik menüből.

## Feladat elemzése

A feladatot részekre bonthatjuk.

**Első sorban** a játékpályákat el kell készíteni, melyet a **program .txt** fájlokból fog beolvasni. Minden egyes objektum egy-egy bizonyos karaktert jelenthet. **A pályák megtervezését követően, az objektumok viselkedését ki kell fejteni.**

Feladat leírása alapján szükséges lesz magaa **Yogi Bear**, az **élelem**, amit meg kell szereznie, a **vadőrök**, melyek akár többen is lehetnek, illetve a **hegyek** és a **fák** megjelenítéséhez szükséges objektumok. Míg az utóbbi kettőnek csak a szélessége és magassága a fontos, addig a piknikkosaraknál fontos a felvehetőség. Ezen felül a vadőrök véletlenszerű irányban mozoghatnak, és amennyiben 1 egység távolságra kerülnek a Yogitól, akkor el kell kapniuk a macit. Yogi a WASD-vel mozoghat 4 irányban, felveheti a piknikkosarakat és életerővel rendelkezik.

Amennyiben az objektumainkat megterveztük, úgy **szükséges lesz ezt a modellt megjelenteni**. A megjelenítéshez **minden egyes objektumhoz egy .png képet rendelünk**. A pálya minden egyes „üres” pontján, egy-egy háttérjellegű .png képet helyezünk, és **az összes szereplő egy 2 dimenziós mátrixban mozoghat**.

**Végző soron a játék fő logikáját kell megtervezni**, amennyiben Yogi az egyik pályán összeszedte az összes piknikkosarat, úgy a következő pályát szükséges legenerálni. Ha a Yogi az összes életpontját elvesztette egy ugyanazon pályán, úgy meg kell jeleníteni egy dialógus ablakot, ahol a felhasználó beírhatja a nevét és elmentheti a végeredményét egy adatbázisba.

## **Megvalósítási terv**

**Többrétegű architektúrát fogunk használni.** Az első réteg a modell, a második a nézet, harmadik a resource állomány és végül az adatbázis.

**Az első rétegben, a modellben, létrehozzuk** a pályákon lévő betűjelzések alapján a **megfelelő objektumokat**, ezek lehetnek: Yogi, Vadőr, Piknikkosár, Hegy és / vagy Fa. Minden egyes objektumot felruházunk tulajdonságokkal. Yogi tud mozogni, van életereje, vannak pontjai. A Vadőr véletlenszerűen mozog, ha 1 egység távolságra kerül Yogitól, úgy elfogja. A piknikkosár felvehető. A hegy és a fa pedig akadályként fog szerepelni a játékban.

**A view rétegben két java class helyezkedik el**, ahol az egyik magáráért a játéktérért felel, hogy minden egyes objektum a megfelelő .png képpel jelenjen meg, és a másik ami felel az egész játéklablakért, és ahol kezelhetjük az eseményeket.

**Resource rétegben**, egy ún. Resource loaderrel dolgozunk, mely minden egyes objektumhoz képes lesz betölteni a megfelelő képet. Ebbe a rétegbe helyezzük el az összes .png és .txt állományunkat, a főprogram innen fogja beolvasni a játékpályát és az objektumkat.

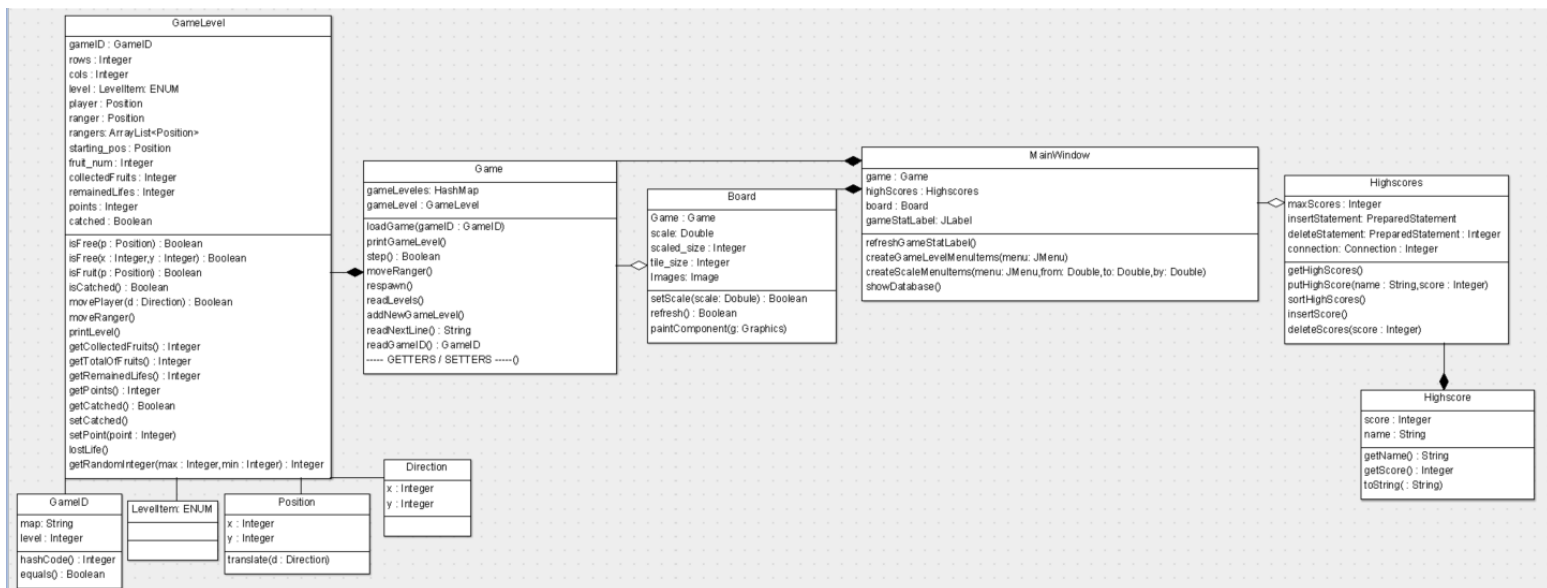
**Végül az adatbázis réteg**, ahol szükségszerűen minimum két java class-szal fogunk dolgozni. Az egyikben meghatározzuk, milyen formában fognak kinézni a sorok. A feladathoz két fontos tulajdonság szükséges, a pontszám, mely az összes összeszedett piknikkosárral egyezik meg, illetve egy név, ami a pontszámhoz egy nevet fog rendelni. Ezt a párost fogjuk elmenteni magába az adatbázisba, és végül megjeleníteni a felhasználó számára.

## Programozási Technológia – 3. beadandó (1. feladat)

Abdurasitov Alekszandr

A49MZV

Ezek alapján az osztálydiagramm az alábbiak szerint fog kinézni:



## Az implementáció (Legfontosabb algoritmusok)

- **Yogi Bear lép:** Elmentjük a játékos jelenlegi pozícióját és a pozíciót, ahová az eseménykezelő alapján szeretne lépni. Ha az az adott pozícióra lehet lépni, abban az esetben a lépés megtörténik, ha azon a ponton piknikkosár van, akkor felveszi a kosarat és kap egy pontot a felhasználó. Végül ellenőrizzük, hogy el lett-e kapva Yogi, akkor is ha lépett és akkor is, ha nem tudott lépni, mivel akadályba ütközött.

movePlayer(d: Direction): Bool	
curr := Player	
next := curr.translate(d)	
isFree(next) OR isFruit(next)	
isFruit(next)	SKIP
level[next.y][next.x] = LevelItem.EMPTY	SKIP
collectedFruits+=1	
points+=1	
player := next	
isCaught()	
remainedLives-=1	SKIP
player := startin_pos	
caught := true	
return true	
isCaught()	
remainedLives-=1	SKIP
player := startin_pos	
caught := true	
return false	

stuki.hu

- **isCaught() metódus:** Megnézzük, hogy Yogi egy vadőr látótávolságába kerül-e. Végigiterálunk a vadőrök listáján, és megnézzük minden egyes vadőr, hogy Yogi a vadőr + 1 egység távolságnyra esik el

isCaught(): Bool	
ranger in rangers	
player.x = ranger.x AND player.y = ranger.y	
return true	SKIP
player.x = ranger.x+1 AND player.y = ranger.y	
return true	SKIP
player.x = ranger.x AND player.y = ranger.y+1	
return true	SKIP
player.x = ranger.x AND player.y = ranger.y-1	
return true	SKIP
return false	

stuki.hu

- **moveRangers() metódus:** Az összes vadór mozgatása. Végigiterálunk a vadőrökön, és véletlenszámot generálunk [1;5) között. Generált szám függvényében mozgatjuk a vadőröket attól függően, hogy ahová mozogni oda mozoghat vagy sem, ha nem mozoghat, megnézzük az összes többi irányt és oda mozog, ahová először tud mozogni. Ezt a metódus minden egy másodpercben meghívja a MainWindow egy timer segítségével.

moveRanger()

ranger in rangers				
getRandomDirection := getRandomInteger(5,1) // Randomszám generálás [1;5) között				
getRandomDirection = 1 //Felfelé mozgítás, ha lehetséges, különben mozgás máshová				
isFree(ranger.x-1, ranger.y)				SKIP
ranger.x := ranger.x-1	isFree(ranger.x+1, ranger.y)			
	ranger.x := ranger.x+1	isFree(ranger.x, ranger.y-1)		
		ranger.y := ranger.y-1	isFree(ranger.x, ranger.y+1)	
		ranger.y := ranger.y+1	SKIP	
getRandomDirection = 2 //Lefelé mozgítás, ha lehetséges, különben mozgás máshová				
isFree(ranger.x+1, ranger.y)				SKIP
ranger.x := ranger.x+1	isFree(ranger.x-1, ranger.y)			
	ranger.x := ranger.x-1	isFree(ranger.x, ranger.y-1)		
		ranger.y := ranger.y-1	isFree(ranger.x, ranger.y+1)	
		ranger.y := ranger.y+1	SKIP	
getRandomDirection = 3 //Balra mozgítás, ha lehetséges, különben mozgás máshová				
isFree(ranger.x, ranger.y-1)				SKIP
ranger.y := ranger.y-1	isFree(ranger.x, ranger.y+1)			
	ranger.y := ranger.y+1	isFree(ranger.x-1, ranger.y)		
		ranger.x := ranger.x-1	isFree(ranger.x+1, ranger.y)	
		ranger.x := ranger.x+1	SKIP	
getRandomDirection = 4 //Jobbra mozgítás, ha lehetséges, különben mozgás máshová				
isFree(ranger.x, ranger.y+1)				SKIP
ranger.y := ranger.y+1	isFree(ranger.x, ranger.y-1)			
	ranger.y := ranger.y-1	isFree(ranger.x-1, ranger.y)		
		ranger.x := ranger.x-1	isFree(ranger.x+1, ranger.y)	
		ranger.x := ranger.x+1	SKIP	

stuki.hu

## Esemény-eseménykezelők párosítások

Megvalósítási terv alapján az alábbi eseménykezelők szükségesek

### 1. Mozgatás


#### a. W, A, S, D billentyűk

- i. Attól függően melyik billentyűt nyomjuk le, oda fog mozogni a maci.  
(További kifejtést nem igényel).

### 2. Menü:

#### a.

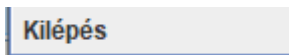
##### i. Játék

1.  (Level 1-10)

- a. Pálya menüpont kiválasztása esetén, lehetőség van kiválasztani 1-10-ig a játékpályákat, kiválasztás után betöltődik az adott pálya.

2.  (1.0x, 1.5x, 2.0x)

- a. Játéklablak ZOOM-ját lehet vele állítani.

3. 

- a. Játéklablak bezárul

##### ii. Eredmények

1. Eredmények menüpontra kattintva megjelenik egy JTable, ahol az adatbázis alapján előáll az eredménylista, ahol TOP 10 eredmény fog megjeleni.

Budapest, 2020.12.07.

Készítette,  
Abdurasitov Alekszandr  
A49MZV