Programozás alapjai 3 – NHF-dokumentáció

Aknakereső

Háttértörténet:

A történetünk egy új kisegér érkezésével kezdődik egy kis városkába, ahol a piciny lakosnak fel kell derítenie, merre nincsenek vérmes cicák. Készített egy alagút rendszert, amivel minden házat el tud érni közvetlenül. Minden háznál van egy ott lakó kutya, aki elmondja hogy van-e a házban cica, illetve jó szaglása miatt azt is, hogy a környező házakban van-e. Ha a kisegér véletlen egy olyan házban bukkan fel, ahol van cica, akkor sajnos a küldetése nem sikerült, és az örök sajtok mezején szaladgálhat tovább. A feladat tehát a szomszédság teljes felderítése, hogy főhősünk nyugodtan tudjon házról házra járni és éldegélni. Sok sikert és jó szórakozást!

Felhasználói kézikönyv:

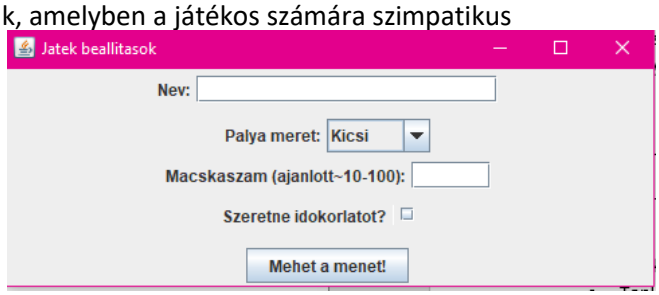
Az aknakeresőmben, a háttértörténet miatt macskakeresés történik, az a lényege, hogy a játékos, anélkül, hogy macskára kattintana felderítse a teljes pályát. Ha macskára nyom, akkor a játékos veszített. Ha minden macskát megjelöl és azon kívül semmit, akkor nyert. Ha időkorlátot is kért, akkor abban az esetben is veszíthet, ha lejár az idő.

Program menete:

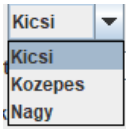
A program indításakor megjelenik egy ablak, amelyben kiválasztható, hogy a játékos mit szeretne tenni:

1. Jatek!
2. Toplista
3. Hattertortenet

1. A Jatek!-ra kattintva, megnyílik egy másik ablak, amelyben a játékos számára szimpatikus pályabeállításokat végezhet.

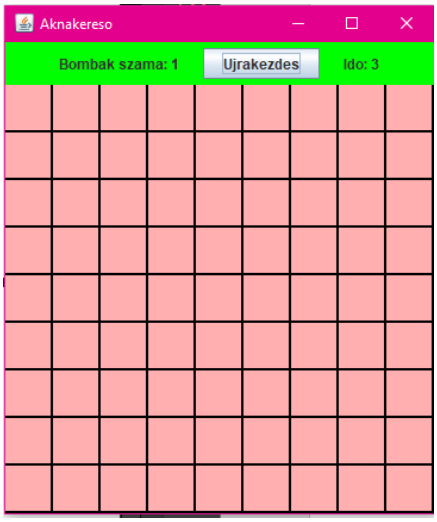


- Nev: Ide írja a játékos a nevét
- Palya meret: Itt 3 lehetőség közül választhat:
  - Kicsi – 9x9
  - Közepes– 16x16
  - Nagy– 30x16

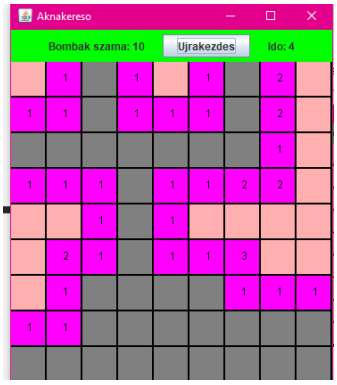


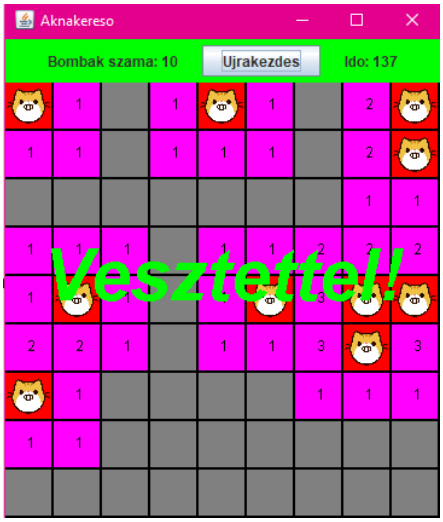
- Macskaszam: Megadhatja a játékos, hogy hány macska legyen a pályán. Ajánlott a 10 és 100 közötti macskaszám, illetve ha túllépi a pálya nagyságát a szám, akkor minden mező macska lesz. Ha a Játékos 0-nál kisebb számot, vagy semmit se ad meg, akkor 10 darab az alapértelmezett macskák száma.
- Szeretne idokorlatot?:
  - Ha nem pipálja be, akkor addig megy a játék amíg nyer, veszít, vagy kilép a játékos, eközben számolja a felhasznált időt.
  - Ha bepipálja, akkor Kicsi pálya esetén 300, Közepes pálya esetén 600 és Nagy pálya esetén 900 másodperce van a játékosnak. Ha az ideje lejár mielőtt sikerülne teljesíteni a pályát, akkor a következő kattintásnál veszít a játékos

-Ha a Mehet a menet! gombra kattint, akkor elindul a játék, feláll a választott paraméter szerinti pálya és elkezd menni a számláló:

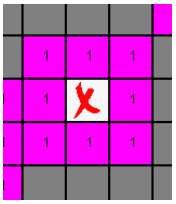


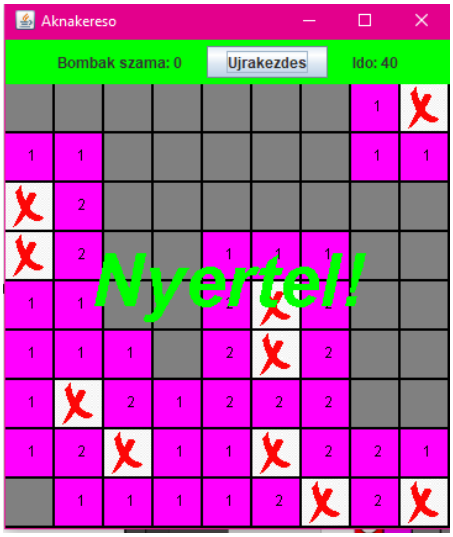
- Ujrakezdes: Erre a gombra kattintva a már megadott paraméterekkel indul egy új játék
- Ha a játékos kattintáskor üres mezőre nyom, akkor minden ezzel üres mező által közvetetten érintkező üres mező felderül, és az ezekkel először határos számok is.



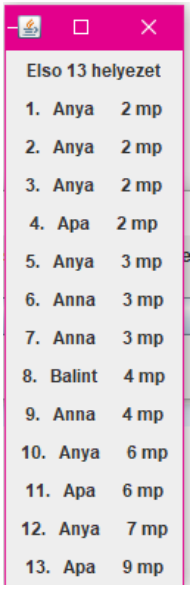


- Ha a játékos kattintáskor számra nyom, akkor csak az adott szám derül fel.
- Ha a játékos kattintáskor macskára nyom, akkor veszített.
- Ha a játékos az egér bal gombjával kattint, akkor megjelöli a mezőt, hogy szeinte ott macska van. Ha újra kattint, eltűnik a jelölés.
- Ha játékos minden macskát megjelöl, és ezenkívül nincs olyan bejelölve ami nem macska, akkor a játékos nyert. Az eredménye lementődik, és ha benne van a top 50-ben akkor a Toplista résznél meg is jelenítődik.

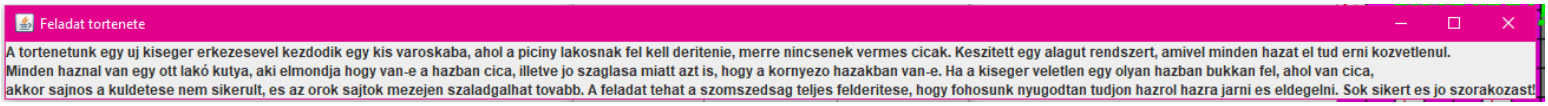




2. A Toplista gombra kattintva megjelenik a top 50 helyezett játékidő alapján, ha van annyi, ha kevesebb, akkor annyi ahány helyezett van.



3. A Hattertortenet gomb hatására megjelenik a játék háttértörténete:



Program megvalósítása:

- JavaSwing segítségével készítettem a grafikai környezetet.
- Eclipse IDE for Java Developers segítségével írtam meg a program kódot.
- Osztályok:
  - **Menu** osztály: Ebben van maga a menü, amiből tovább lehet lépni a toplista megnézésére, át lehet lépni a pálya beállításaihoz és megnézni a feladat történetét.
  - **Palyabeallitasok** osztály: Ebben kéri be a felhasználó által megadott adatokat, majd tovább lehet lépni magára a játékra
  - **Jatek** osztály: Elkészül a pálya majd utána itt zajlik a játék.
  - **Toplista** osztály: Ennek segítségével íratjuk ki a toplistát.
  - **Main** osztály: Ebben van meghívva a menü.
  - **Fejlec** osztály: Itt van az időszámláló, a bombaszámláló, és a játék újraindító.
  - **Feladatleiras** osztály: Itt jelenik meg a feladat háttértörténete.
  - **Helyezett** osztály: A nyertes játékos eredményét tárolja.
  - **Osszehasonlito** osztály: A benne lévő függvény összehasonlítja két nyertes idejét, és aszerint ad vissza értéket.
  - **Mezo** osztály:

Menu:

+Menu(): Elkészíti a menü JFrame grafikus felületét és kiadja a 3 gombot, amiből választani lehet.
+JatekGomb: Ez egy kis class a Menu-n belül, amely azt figyeli, hogyha a játék gombra nyomnak és olyankor egy Palyabeallitasok-at készít.
+ToplistaGomb: Ez egy kis class a Menu-n belül, amely azt figyeli, hogyha a toplista gombra nyomnak és olyankor egy Toplista-t készít, amit láthatóvá tesz.
+FeladatleirasGomb: Ez egy kis class a Menu-n belül, amely azt figyeli, hogyha a feladatleírás gombra nyomnak és olyankor egy Feladatleiras-t készít, amit láthatóvá tesz.

Palyabeallitasok:

+Palyabeallitasok(): Elkészíti a pályabeállítások bekéréséhez szükséges grafikus JFrame felületet.
+getBomb(): Visszaadja a macskák számát stringben.
+mezoSorokSzama(): A bekért méret string segítségével visszaadja a sorok számát.
+mezoOszlopokSzama(): A bekért méret string segítségével visszaadja az oszlopok számát.
+ujJatek(): Készít egy JFramet és annak elkészíti megjelenését, amelybe egy Fejlec-et és egy Jatek-ot helyez el JPanel segítségével. Itt adja át a lekért adatokat, és vizsgálja le, hogy a macskák számát megfelelően adja-e meg.
+IndulJatek: Ez egy kis class a Palyabeallitasok-on belül, amely azt figyeli, hogyha a Mehet a menet! gombra nyomnak és olyankor meghívja az ujJatek() függvényt.

Jatek:

+Jatek(String, int, int, int, boolean, String): Elkészíti a játékhoz szükséges grafikus JPanel felületet, beállítja a kapott attribútumok segíttségével a saját attribútumait, meghívja a szamElhelyezo() függvényt és elindít egy időzítőt.
+szamElhelyezo(): Elhelyezi a macskákat és megadja a macskákkal szomszédos mezők értékét.
+getIdo(): Visszaadja a megkapott méret string segítségével az időt.
+getBomba(): Visszaadja a macskák számát.
+setVesztett(): Beállítja a vesztett atrributúm értékét true-ra.
+getVesztett(): Visszaadja, hogy vesztett-e a játékos.
+getNyert(): Visszaadja, hogy nyert-e a játékos.
+jatekosNyert(Graphics): Leállítja az időzítőt, kiírja, hogy vesztett a játékos, felfedi a teljes pályát, lementi a játékos eredményét a toplistába és meghívja a fajlbaIr() függvényt.
+jatekosVeszitett(Graphics): Kiírja, hogy vesztett a játékos és felfedi a teljes pályát.
+paintComponent(Graphics): Megfesti a teljes pályát és az aktuális tartalmát a mezőknek.
+felderit(int, int): A megkapott számok segítségével felderíti azon mezőket, amelyeket fel kell és false-ra állítja a felderített mezők felsőréteg attribútumát.
+egerrelKattint: Ez egy kis class a Jatek-on belül, amely azt figyeli, az egérrel való kattintást. A bal egérgomb esetén felderít és a felsőréteg változóját a mezőnek false-ra állítja. A jobb egérgomb esetén megjelöl, vagy ha már jelölve van, akkor leveszi a jelölést a mezőről. A végén megvizsgálja, hogy nyert-e a játékos.
+actionPerformed(ActionEvent e): Az időt kezeli és leállítja a az időzítőt, ha nyer a játékos.

Toplista:

+Toplista(): Elkészíti a grafikus JFrame felületét, beolvassa az adott fájlt, sorba rendezi, majd kiírja a grafikus felületre JPanel és JLabel segítségével.
+getNyertesek(): Visszaadja a nyertesek adatait.
+ fajlbolOlvas(): Beolvassa a toplista.ser-ből a nyertesek eredményeit egy Helyzettekből álló ArrayList-be.
+fajlbaIr(): A Helyzettekből álló ArrayList-et kiírja a toplista.ser-be

Fejlec:

+ Fejlec(Jatek, Palyabeallitasok, JFrame, String, boolean): Elkészíti a fejléc grafikus JPanel felületét, a megkapott adatokra beállítja a saját adatait, a méret alapján megadja az idő limitet, ha kér a játékos limitet és elindítja az időszámlálót.
+ujJatek(): Meghívja a pályabeállításokra az abban az osztályban lévő ujJatek() függvényt és bezárja az aktuális játékot.
+ megNyom: Ez egy kis class a Fejlec-en belül, amely azt figyeli, hogyha az Ujrakezdes gombra nyomnak és olyankor meghívja az ujJatek() függvényt.
+ actionPerformed(ActionEvent): Kezeli az időzítő leállítását nyerés, vagy veszteség esetén, és számolja az időt.

Feladatleiras:

+Feladatleiras(): Elkészíti a grafikus JFrame felületet és kiírja a feladatleírását.

Helyezett:

+Helyezett(String, int): Beállítja, hogy a kapott adatokkal egyenlő a Helyezett osztály attribútumai.
+int getIdo(): Visszaadja az időt.
+String toString(): A Helyezettet stringként adja vissza.

Osszehasonlito:

+int compare(Helyezett h1, Helyezett h2): Összehasonlítja két nyertes eredményét, és attól függően ad vissza egy számot.

Mezo:

+Mezo(): Beállítja az értéket 0-ra, a felsőréteget true-ra és a jelölést false-ra.
+ getErtek(): Visszaadja a mező értékét.
+ setErtek(int): Beállítja a mező értékét a megkapot integerre.
+ Novel(): Növeli 1-el a mező értékét.
+ getFelsoreteg_e(): Visszaadja, hogy a mező felsőrétege meg van-e még.
+ setFelsoreteg_e(boolean): Beállítja a felső réteg állapotát a megkapott booleanra.
+ getJelolve_e(): Visszaadja, hogy jelölve van e a mező.
+ setJelolve_e(boolean): Beállítja a jelöltség állapotát a megkapott booleanra.
+ bomba_e(): A mező értékét megvizsgálva vissza adja, hogy a mezőn macska van-e.

Osztálydiagram: