TUDNIVALÓK:
Most is és a következő gyakorlatokon is – akkor is, ha külön nem emeljük ki – az órán meg nem oldott feladatok HÁZI FELADAT-ként megoldandóak!!!
Ez fontos a tárgy sikeres teljesítéséhez!

Játsszunk! ☺

1. feladat:

Az ábrán látható felület belső mérete: 850*460, a vezérlő rész 150 pixel széles.

Induláskor minden üres, csak a háttérkép látszódik. Az „Adatbetöltés” gomb hatására kerülnek be a halak a rendszerbe. Ez azt jelenti, hogy létre is jönnek a hal példányok: mindnek lesz neve, egyértelműen megadható hozzájuk a bal- és jobboldali profilképük és a képméret. Esetünkben a neveket az *adatok.txt* fájl tartalmazza, a képek a kepek mappában találhatóak a nevek sorrendjében, a méretük pedig véletlenül generálható a megadott határok között. Az adatfájlt egy fájlválasztó segítségével keressük meg.
Adatbetöltés után megjelenik a nevük a listafelületen. A „Vízbedobás” gomb csak akkor válik aktívvá, ha sikerült beolvasni az adatokat.

A „Vízbedobás” gomb hatására a listából kiválasztott halak (egyszerre többet is lehet választani) bekerülnek a vízbe, és ott elkezdenek úszkálni. Egyúttal a listából kikerül a nevük.

Az úszás egyelőre ezt jelenti: véletlen sebességgel véletlenszerűen indulnak balra vagy jobbra, és ha az „akvárium” falához érnek, akkor visszafordulnak.

A vízre kattintva megijednek, és vagy egyszerre mind megáll, vagy ha éppen álltak, akkor egyszerre mind nekiindul. Mivel a halak erősen társas lények, ezért az újonnan vízbedobott hal alkalmazkodik a többihez, és ha ők állnak, akkor az új is csak akkor kezd el mozogni, amikor a többi, ha mozognak, akkor ő is mozog.

Oldjuk meg azt is, hogy a hangszóró gombra kattintva szólaljon meg egy háttérzene, majd ismét megnyomva hallgasson el.

b/ A kicsit „élethűbb” mozgás kedvéért oldjuk meg azt, hogy időnként ne csak akkor forduljon meg egyik-másik, amikor falhoz ér, hanem akkor, amikor kedve tartja.

c/ Engedjünk meg a mozgásukban némi le-föl hullámzást is.

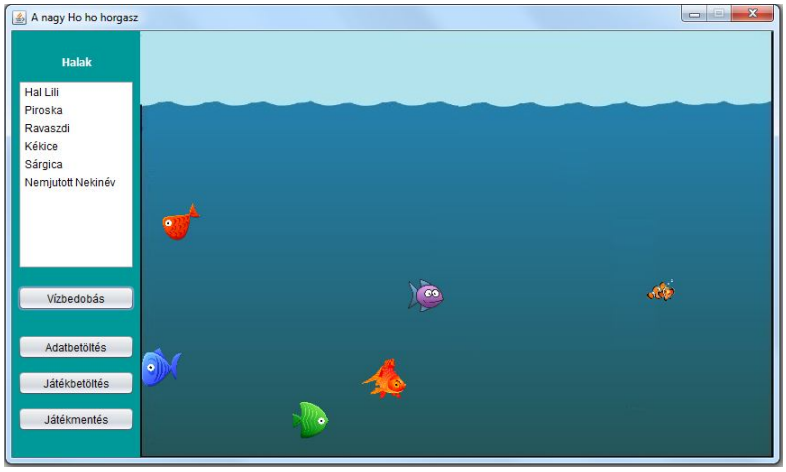
d/ Lehet, hogy mégsem annyira összetartóak, mert most ha úgy kattintunk a vízre, hogy eltaláljuk az egyiket, akkor csak ő álljon meg (vagy induljon el), a többi mozgása ne változzon. Ekkor az új hal a vezér mozgása szerint mozog. (Vezér az, aki legrégebb óta bent van a vízben.)

e/ Lássa el az alkalmazást egy induló felülettel. A felületre kattintva tűnjön el ez a frame, és jelenjen meg helyette a másik.

Segítség: A frame-n lévő panelről így lehet hivatkozni a tartalmazó frame-re:

```
JFrame frame =
    (JFrame) SwingUtilities.getRoot(this);
```

f/ Ha mégsem csak gyönyörködni szeretne a halakban, akkor az eltalált halat tüntesse el az akváriumból. (Ha jobban tetszik, kevésbé csicsás akváriumban is úszkáltathatja őket.) Úgy is módosíthatja, hogy az eltalált halat majd ismét vissza lehet dobni, vagyis a neve kerüljön vissza a listába.



g/ Oldja meg, hogy el tudja menteni az aktuális állapotot, illetve vissza is tudja tölteni azt.

Az igazi kihívás az, ha objektumként akarja menteni (egyetlen mentendő objektumpéldány a halakat tartalmazó lista), ugyanis az Image nem szerializálható, csak az ImageIcon osztály. (De persze, kitalálhat olyan megoldást is, hogy a képeket ne kelljen szerializálni.)

h/ Oldja meg (vagy alakítsa át) úgy, hogy a Hal osztály a JComponent leszármazottja legyen. (A panelt BorderLayout-ra kell állítani, a takarást a komponens z order tulajdonságának beállításával lehet szabályozni – setComponentZOrder().)

i/ Oldja meg executor segítségével – ez csak a köv. előadáson szerepel majd ☹.

j/ Oldja meg úgy, hogy a vezérlőpanel is független legyen a Hal osztálytól, azaz használjon generikusokat.

k/ Az adatokat adatbázisból töltse be – a megadott SQL fájl Derby adatbázishoz való.

Segítség:

Az mp3 fájlok kezelésére egy külső, javazoom nevű library-t használunk. Ezt hozzá kell szerkesztenünk a programunkhoz. Legjobb megoldás, ha rábízzuk ezt a munkát a **Maven** programra. Ezért induláskor Maven projektet hozunk létre, és ebben a pom.xml fájlban beállítjuk a függőségeket, jelenleg csak a javazoom használatához szükségeset. Ezt innen lehet bemásolni:
https://mvnrepository.com/artifact/javazoom/jlayer/1.0.1

Ennyivel kell bővíteni a generált fájlt:

```
    <dependencies>
        <!-- https://mvnrepository.com/artifact/javazoom/jlayer -->
        <dependency>
            <groupId>javazoom</groupId>
            <artifactId>jlayer</artifactId>
            <version>1.0.1</version>
        </dependency>

    </dependencies>

</project>
```

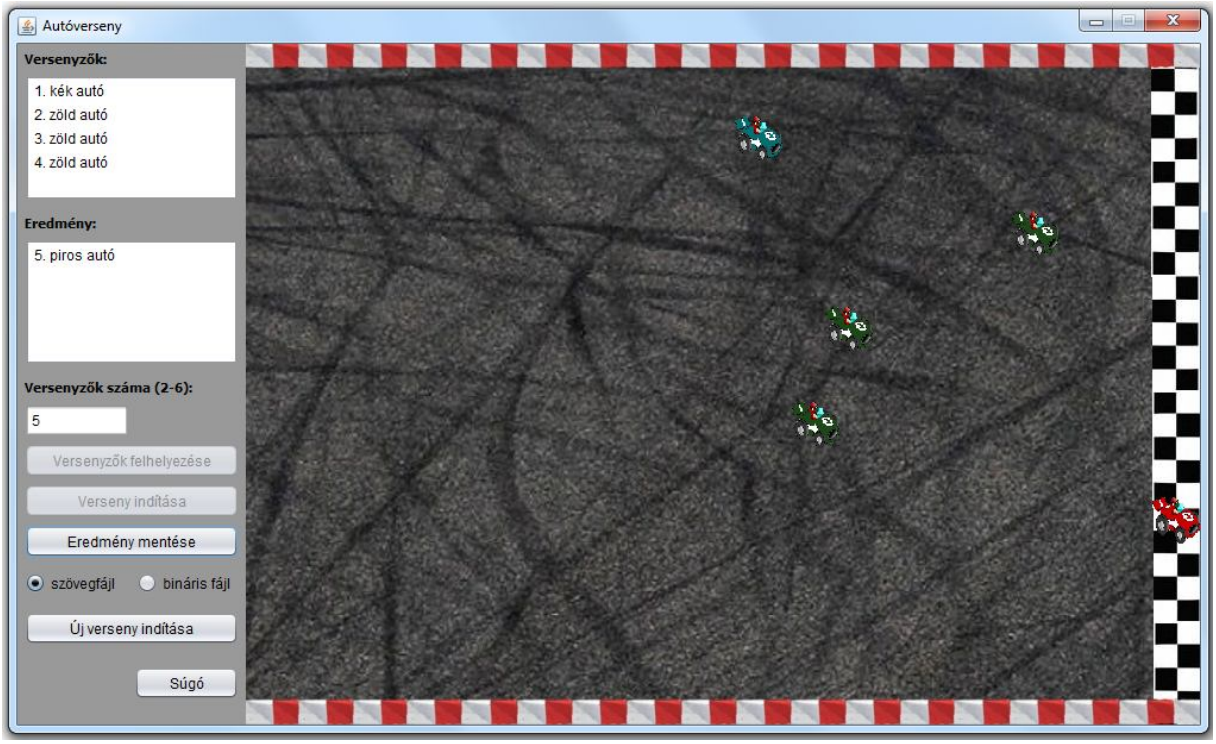
A projektet lezáró tag benne van a generált xml-ben, ide csak azért másoltam, hogy lássa, ez elé kell másolnunk.

Az adatokat betöltheti egy adatbázisból is – a megadott SQL fájl Derby adatbázishoz való.

Ha kedve van eljátszadozni a zenével, íme egy ötlet:
http://stackoverflow.com/questions/16882354/how-to-play-pause-a-mp3-file-using-the-javazoom-jlayer-library

Esetleg utánanézhet a javafx.scene Media, illetve MediaPlayer megoldásának is – ehhez NetBeansben nem is kell Maven projekt.

Ugyancsak kipróbálhatja a már említett, .odd zenéket kezelő lwjgl és slick2d csomagokat – ezeket szintén Maven segítségével.
Ez kicsit több utánajárást igényel, mivel még „gyalog” is le kell szedni fájlokat:
http://wiki.lwjgl.org/wiki/Setting_Up_LWJGL_with_IntelliJ_IDEA.html
illetve az lwjgl natív kód letölthető innen :
https://sourceforge.net/projects/java-game-lib/files/Official%20Releases/LWJGL%202.9.3/
ebből is a native/megfelelő OS mappát kell célnak megadni és úgy működik.



2. feladat:
Írjon egy autóversenyzős programot. Az ábra magárét beszél, de a feladatleírás:
A képen látható felület mérete 1000*600-as, a baloldali vezérlő rész 200 pixel széles.

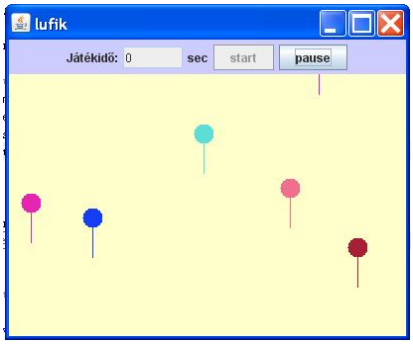
Megadható a versenyzők száma (nyilván korlátozni kell, de NE égesse be a korlátozásra vonatkozó értéket). Ekkor még csak a Versenyzők felhelyezése gomb aktív. Felhelyezve őket aktívvá válik a Verseny indítása nyomógomb, evvel lehet elindítani a versenyt – a többi gomb inaktív. A verseny eredménye a második listában látható a beérkezés sorrendjében.
Ha mindenki beérkezett, aktívvá válik az Eredmény mentése gomb és az Új verseny indítása gomb is.
Ha akarjuk, elmenthetjük az eredményt egy fájlba, mégpedig a választástól függően vagy szövegfájlba, vagy binárisba (objektumként).
Az Új verseny indítása gomb hatására minden visszaáll alapállapotba.

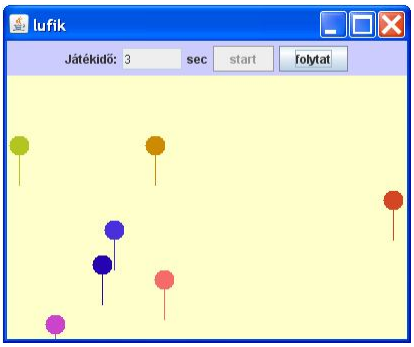
A Súgó gomb mindig aktív, bármikor lehet segítséget kérni. Megnyomása után rövid leírást olvashatunk a versenyről.

b/ Az ábrán látható változatban az autókat egyedi sorszám alapján különböztettük meg, és csak a színük jellemezte őket. Módosítsa a feladatot úgy, hogy a program indulásakor azonnal betöltődnek nevek egy neveket tartalmazó adatfájlból, a versenyzők felrakásakor pedig ezekből a nevekből véletlenszerűen rendelünk egyet-egyet egy-egy autóhoz. Minden név csak egyszer szerepelhet. A versenyzők listájában az autókhoz rendelt nevek jelenjenek meg.

Képek: feladatok/ autoverseny_kepek mappa, vagy keressen magának.

3. feladat



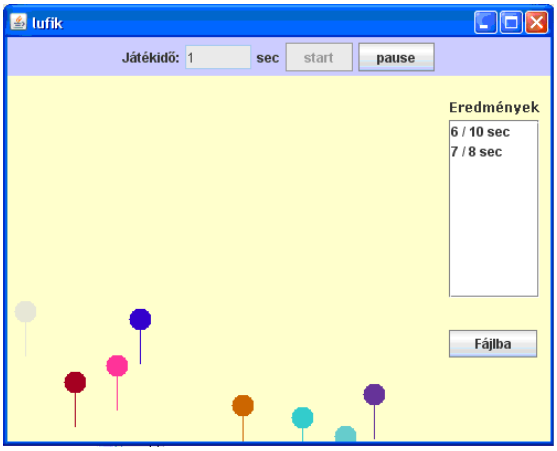


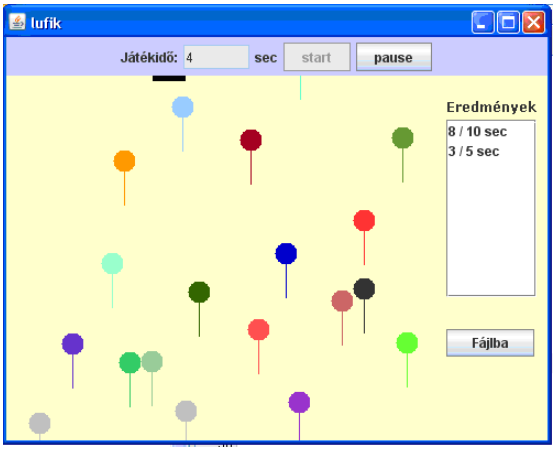
Miután beállítottuk a játékidőt, a start gombra kattintva indulhat a játék. Ez azt jelenti, hogy bizonyos időközönként véletlen helyről, véletlen színű és véletlen gyorsaságú lufik kezdenek szálldogálni, amelyeket egérkattintással kell elkapnunk.
A pause gomb hatására felfüggeszthetjük a játékot – természetesen ekkor találatunk sem lehet, majd ismét ugyanerre a gombra kattintva folytathatjuk. (A gomb felirata értelemszerűen változik.)

Amíg le nem járt a játékidő, addig a start gomb is és a beviteli szövegmező is inaktív. Ha lejárt az idő, akkor a panel aljára írja ki a találatok számát.
Utána a start gomb és a beviteli mező ismét aktívvá válik, és kezdődhet újra a játék.

Természetesen illik arra is figyelni, hogy a beviteli mezőbe ne kerülhessen hibás adat, ellenkező esetben küldjön hibajelzést.

b/ Bővítse ki a feladatot az ábrán látható módon.
Vagyis rakjon fel egy listát is, amelybe a játszma végén bekerül az eredmény találat / idő formában. Figyeljen rá, hogy a listára ne szálljon lufi.
A játszma végén továbbra is jelenjen meg a találatok száma, és a Fájlba gomb hatására a lista teljes tartalmát mentse fájlba.
(Ha sok ideje van, avval is kísérletezhet, hogy nem az eredményeket, hanem az aktuális lufiállapotokat menti fájlba.)





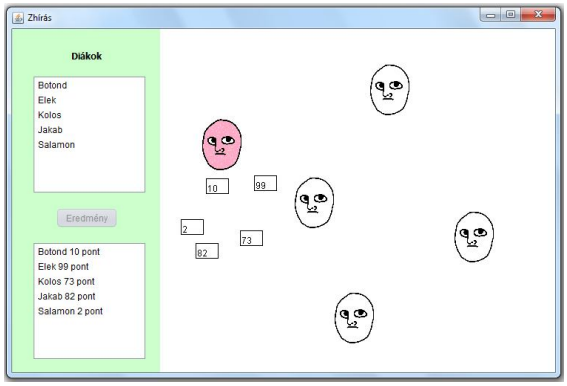
c/ Egészítse ki a feladatot úgy, hogy a sárga panel tetején egy ütő (vagy lap) mozog balra, jobbra. Az ütőt az „A”, „S” billentyűkkel lehet irányítani.

Indítsa be a fantáziáját, és adjon funkciót az ütőnek, vagyis történjen valami a hatására. (Pl. üsse vissza a lufit, vagy számolja, hogy hányat talált el, stb.)

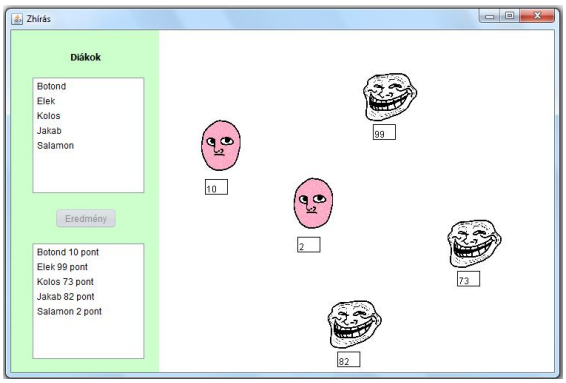
A **teacher the troll** vizsgafeladat erre a két zh feladatra volt válasz ☺

1. Közismert, hogy a facebook sehogy sem hagyja nyugton a diákokat. Itt is magához akarja hívni őket. Sajnos, nem is kell sok hozzá, és a diákok mennek vele. Olyannyira, hogy hamarosan még trollkodási vizsgára is ráveszi őket. Ezt a folyamatot követhetjük végig ebben az alkalmazásban.
Egy 750*500-as felület 550 széles jobboldali részén induláskor néhány másodpercig látható, hogy a facebook kérleli a diákot. Pár másodperc múlva azonban kötélnek is áll, és elmegy vele, azaz a kép szép lassan jobbra kiúszik. Az utána maradó fehér felületen gyülekeznek a vizsgára váró diákok: egy-egy kattintásra megjelenik egy-egy diák, a diák neve pedig a baloldali felső listában lesz olvasható. A kattintás helye a kép középpontja legyen.
A neveket a program indulásakor automatikusan olvassa be egy adott fájlból. Lehetőleg úgy oldja meg, hogy a program jar-ból is indítható legyen külön adatfájl hozzáadása nélkül, és tudja kezelni az ékezetes karaktereket is. Ha nem megy a fájlból olvasás, akkor adjon meg konstans módon egy névlistát, és innen vegye a neveket.

Legfőljebb csak annyi diák jelenhet meg, ahányan a névsorban szerepelnek, és mindenki csak egyszer.



Ha már nem akarunk több diákot megjeleníteni, akkor kattintsunk az Eredmény gombra. Ennek hatására a vizsgázó emberek elkezdik megkapni az eredményt, mégpedig úgy, hogy mindegyikőjük számára generálódik egy 0 és max érték közötti pontszám, és ez a pontszám egy téglalap alakú fehér lapon véletlen sebességgel elindul a rajzfelület bal oldalának közepéről az illető diák felé. Amikor egy diák megkapja az eredményt (a zh odaér hozzá – a zh bal felső sarka a trollkép bal alsó sarkától negyed szélességgel jobbra van), akkor az eredménytől függően megváltozik az arca.
Ha a pontszám egy, az összes diákra egyformán érvényes határ alatt van, akkor elpirul, ha fölötte van, akkor kacagni kezd.



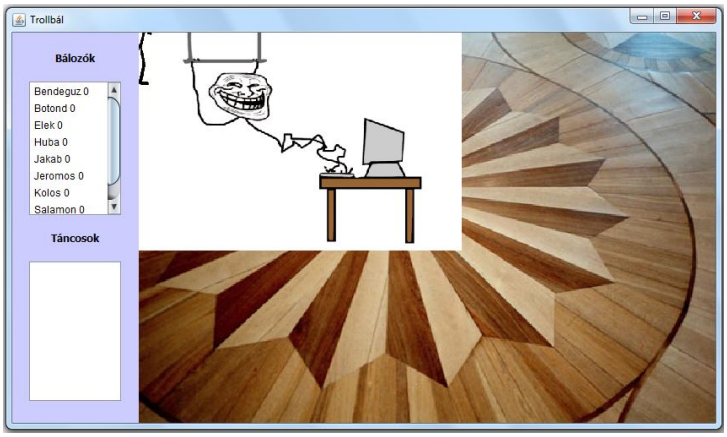
A zh-eredményt csak egyszer lehet elkérni, ezért a gomb a gombnyomás után váljon inaktívvá.
(Az esetlegesen nem író emberek további felrakását nem kell megakadályozni, hiszen miért ne találkozhatnának a többiekkel, csak ők már nem kapnak pontot.)

Segítség:
Az adatfájl: nevek.txt.
A képek: facebook.jpg, troll.gif, trolluj.gif és trolluj2.gif.

A képen látható méretek: troll: 80*80-as, a téglalap 30*20-as, a szöveg bal sarka pedig 2-2 pixellel van beljebb a téglalap bal alsó sarkától.
Vegye figyelembe, hogy minden zh ugyanakkora, és ugyanonnan indul.

Azt, hogy másfajta szöveg jelenjen meg az ugyanahhoz az emberekhez tartozó listában, úgy tudja megoldani, hogy két különböző modellt hoz létre, és más-más modellt rendel a két listához.

2. Furcsállottam, hogy miért a hét közepén van a gólyabál. Kérdésemre azt válaszolták, hogy azért, mert hét végén foglalt a terem. Utánajártam, hogy kik használták, és íme, troll-bál volt. Reprodukálja, azaz írjon Java programot a következőkre:



Egy 600*500-as méretű táncparkett mellett egy 150-szer 500-as felület jelzi, hogy kik a bálozók, és éppen kik táncolnak.
Induláskor automatikusan betöltődik egy adatfájlból a résztvevők névsora. A nevek melletti szám azt jelzi, hogy hányszor táncoltak. A még a serényen dolgozó trollt látjuk, de ez a kép pár másodperces várakozás után fokozatosan csökkenve eltűnik a táncparkett bal felső sarkában.
Lehetőleg úgy oldja meg a fájlból való olvasást, hogy a program jar-ból is indítható legyen külön adatfájl hozzáadása nélkül, és tudja kezelni az ékezetes karaktereket is. Ha nem megy a fájlból olvasás, akkor adjon meg konstans módon egy névlistát, és innen vegye a neveket.

A bálozók listájában lévő névre kattintva (egyszerre csak egy embert lehessen kiválasztani!), az illető elindul a táncparkett bal felső sarkából a parketta egy véletlenül választott pontjába, amikor odaér, akkor megáll, és elkezd táncolni (a táncoló figura animált gif). Véletlen ideig táncol, majd visszatér a kiinduló pontba. Arra ügyeljen, hogy a táncos figura ne lógjon ki a rajzfelületről, vagyis úgy válassza a véletlen-generálás határait, hogy figyelembe veszi a rajzfelület aktuális méretét is és a kép méretét is.

A kiválasztott táncos neve átkerül a táncolók listájába, és a táncai száma értelemszerűen növekszik.
Amikor visszaért a kiinduló helyére, akkor neve kikerül a táncosok listájából, és bekerül a bálozók listájába.

Segítség:
Az adatfájl: nevek.txt.
A képek: parketta.jpg, indulotroll.gif, illetve trolluj.gif és trolluj2.gif.

A képen látható troll mérete: 80*80-as, a gyaloglás egyes lépései közben 10 ms telik el, táncolni 10 és 30 másodperc között táncol, de használhat más értékeket is, csak NE drótozza be őket.
Az induló kép kezdőméretét ne a konstruktorban, hanem később, pl. a szál indításakor adja meg.