

Programozás alapjai 3. Házi Feladat

A program rövid leírása

A program egy egyszerű genetikus algoritmust valósít meg, kihasználva a Java által nyújtott grafikai elemeket. A pontok száma illetve a pálya módosítható a program mellet megtalálható testin.txt fájlban. A fájl a következőképpen néz ki:

Első sor a pontok számát adja meg, majd a következő az akadályok számát határozza meg, az ez után következő sorok 4-es csoportokban az akadályok (és a cél) bal felső és jobb alsó koordinátáit adják meg.

A program indulása után a pontok eredetileg véletlenszerűen mozognak, majd minden egyes generáció végén a fitnessz értékük alapján öröklődnek a következő generációba. A program fejlődésének beakadását az új generáció mozgásának kis szintű randomizálása igyekszik megelőzni. A Start/Stop Save elindítja illetve megállítja a mentést. A mentés a test.txt fájlba történik a következőképpen:

A program minden generáció végén megnézi, hogy meg lett-e nyomva a gomb, és ha igen, kiírja a fájlba a generáció sorszámát illetve a minimum lépések számát. Ha megint megnyomjuk a gombot, a mentés leáll.

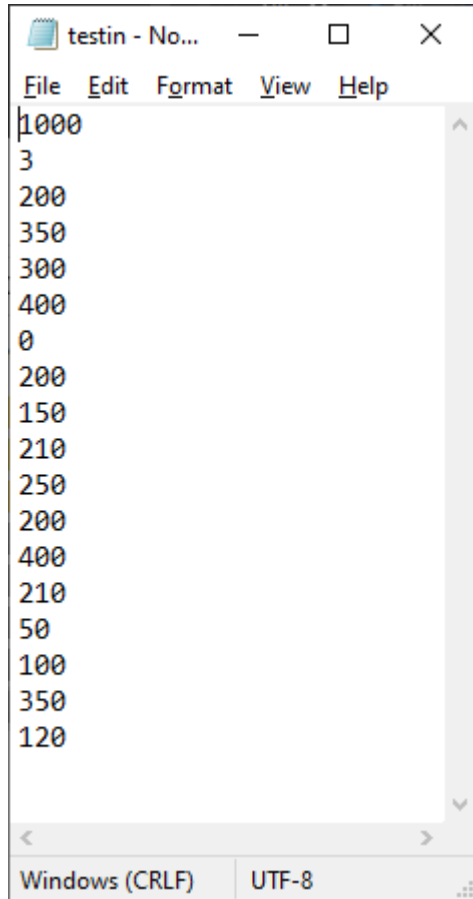
A program egy screenshotja:



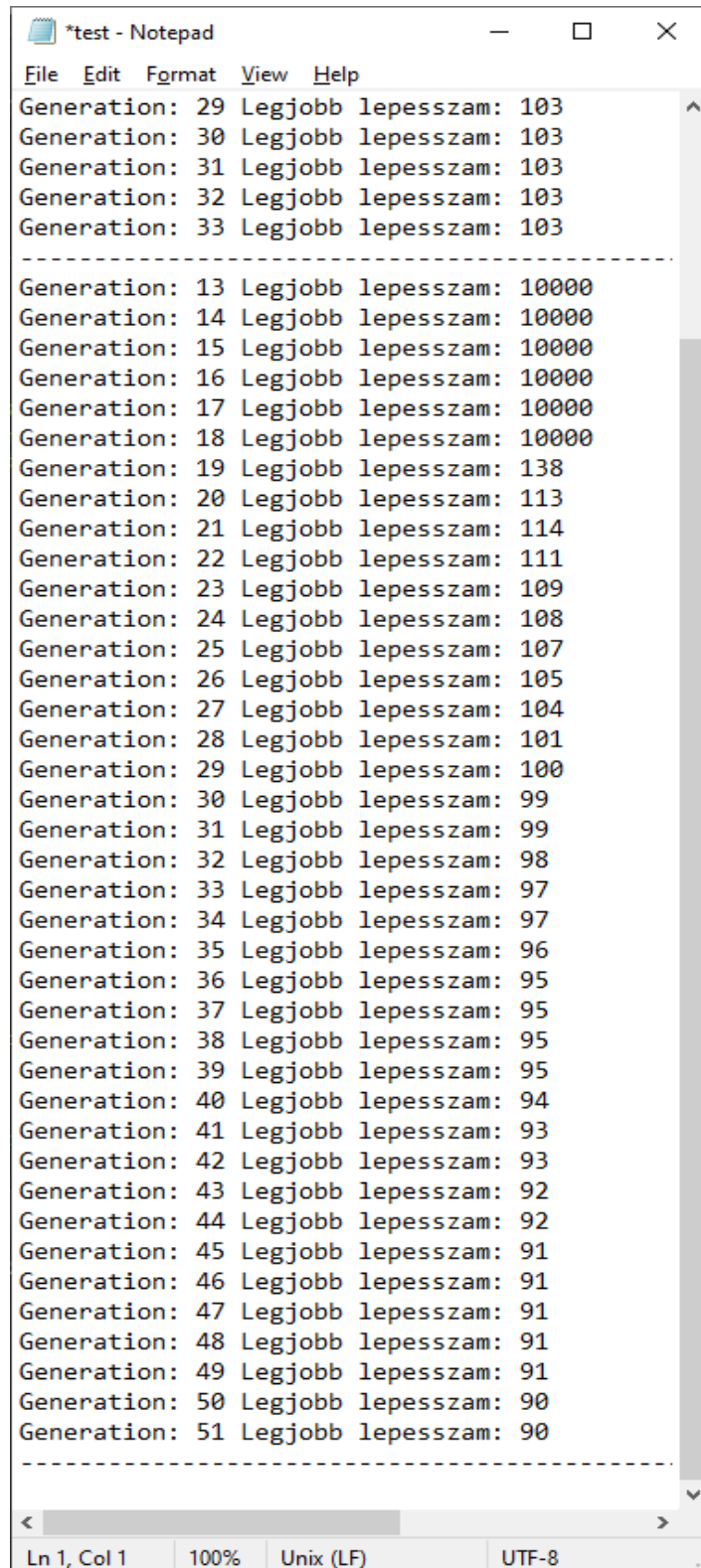
Felhasználói kézikönyv

A pálya testreszabásához a testin.txt fájl végéhez hozzá lehet adni akadályokat a következőképpen: 4 számra van szükség, ami az akadály bal felső sarok x és y, illetve a jobb alsó sarok x és y koordinátái. A fájl első számát módosítva a generáció pontjainak számát módosíthatjuk. A második szám az akadályok száma, ha hozzáadunk akadályt, vagy kiveszünk akadályt, ne felejtsük el ezt módosítani!

A fájl a fentebb látható példapályával:



A program indítás után teljesen magától működik. A Start/Stop Save gomb megnyomásával elindíthatjuk a fájlba mentést. Ekkor a kikapcsolásig minden generáció száma és legjobb lépésszáma ki lesz írva a fájlba. Ennek egy példája:



```
*test - Notepad
File Edit Format View Help
Generation: 29 Legjobb lepeszam: 103
Generation: 30 Legjobb lepeszam: 103
Generation: 31 Legjobb lepeszam: 103
Generation: 32 Legjobb lepeszam: 103
Generation: 33 Legjobb lepeszam: 103
-----
Generation: 13 Legjobb lepeszam: 10000
Generation: 14 Legjobb lepeszam: 10000
Generation: 15 Legjobb lepeszam: 10000
Generation: 16 Legjobb lepeszam: 10000
Generation: 17 Legjobb lepeszam: 10000
Generation: 18 Legjobb lepeszam: 10000
Generation: 19 Legjobb lepeszam: 138
Generation: 20 Legjobb lepeszam: 113
Generation: 21 Legjobb lepeszam: 114
Generation: 22 Legjobb lepeszam: 111
Generation: 23 Legjobb lepeszam: 109
Generation: 24 Legjobb lepeszam: 108
Generation: 25 Legjobb lepeszam: 107
Generation: 26 Legjobb lepeszam: 105
Generation: 27 Legjobb lepeszam: 104
Generation: 28 Legjobb lepeszam: 101
Generation: 29 Legjobb lepeszam: 100
Generation: 30 Legjobb lepeszam: 99
Generation: 31 Legjobb lepeszam: 99
Generation: 32 Legjobb lepeszam: 98
Generation: 33 Legjobb lepeszam: 97
Generation: 34 Legjobb lepeszam: 97
Generation: 35 Legjobb lepeszam: 96
Generation: 36 Legjobb lepeszam: 95
Generation: 37 Legjobb lepeszam: 95
Generation: 38 Legjobb lepeszam: 95
Generation: 39 Legjobb lepeszam: 95
Generation: 40 Legjobb lepeszam: 94
Generation: 41 Legjobb lepeszam: 93
Generation: 42 Legjobb lepeszam: 93
Generation: 43 Legjobb lepeszam: 92
Generation: 44 Legjobb lepeszam: 92
Generation: 45 Legjobb lepeszam: 91
Generation: 46 Legjobb lepeszam: 91
Generation: 47 Legjobb lepeszam: 91
Generation: 48 Legjobb lepeszam: 91
Generation: 49 Legjobb lepeszam: 91
Generation: 50 Legjobb lepeszam: 90
Generation: 51 Legjobb lepeszam: 90
-----
Ln 1, Col 1 100% Unix (LF) UTF-8
```

A program osztályai és fontosabb metódusai

A felsorolásból a getter/setter metódusok, illetve a nem használt metódusok ki vannak hagyva. A legtöbb nem használt metódus csak a Shape implementálása miatt lett definiálva.

Vector

Egy egyedi vektor osztály, ami tartalmazza a következőket:

-x: az x koordináta

-y: az y koordináta

-limit: a limit fölé nem mehet x és y értéke

-fromAngle(double angle) : A brain vektorainak generálására használt függvény, ami kap egy random számot 0-tól 9-ig, és a kapott szám egészrésze alapján állítja be a vektort. Így összesen 9 irányt kaphat (a (0,0)-t is iránynak számoltam.)

Point

A pont osztály tárolja egy pontnak az alábbi adatait

-brain: a pont "agya", lsd. Brain.java

-vel: a pont sebessége

-pos: a pont aktuális pozíciója

-acc: a pont aktuális gyorsulása

-fitness: minél fittebb egy pont, annál valószínűbb, hogy továbböröklődik

-isDed: true - a pont nekiment a falnak/egy akadálnak

-didFinish: true - a pont elérte a célt

-isBest: true - a legjobb pont a generációban (mindig öröklődik)

-calculateFitness(Vector finishA, Vector finishB): Kiszámolja a pont fitnesszt, ahol a kapott vektorok a cél bal felső illetve job alsó sarkai. Két eset van, Az első, ha a pont elérte a célt. Ekkor a lépésszám alapján kap a pont fitnesszértéket. A második, ha nem ért célba. Ekkor a céltól való végső távolság alapján számolja a fitnesszt.

-move(): Mozgatja a pontot. Először az agyban lévő vektorok alapján módosul a pont gyorsulása, majd ezzel az új gyorsulásértékkel módosítjuk a sebességet, végül a sebességet hozzáadjuk a pozícióhoz.

-getDistance(double x, double y, double w, double h): A kapott koordináták által behatárolt területtől számított távolság.

-contains(double x, double y, double w, double h): Megnézi, hogy a kapott koordináták által behatárolt területen belül van-e a pont.

Brain

A pontok "agya"

-directions: egy Vectorokat tartalmazó tömb amiben randomizált vektorokat tárolunk, majd ezt örökli a következő generáció. A randomizált vektorok a pontok gyorsulását módosítják, így módosítva a mozgás irányát.

-step: a megtett lépések száma

-mutate(): A mutationRate-nek megfelelő eséllyel randomizálja az agyat.

-randomize(): Átad egy random számot 0-tól 9-ig a fromAngle() függvénynek. Randomizálja az agyat.

Generation

Egy generáció tárolóosztálya

- points: Point típusokat tároló ArrayList
- numberOfPoints: igazából points.size()
- finishA: A célterület bal felső sarka
- finishB: A célterület jobb felső sarka
- fitnessSum: fitnesssek összege, a szülő kiválasztásához kell
- bestPointID: A legjobb pont indexe
- gen: A generáció sorszáma
- minStep: A legkevesebb lépés amiből befejezhető a pálya

- calculateFitnessSum(): A fitnesszek összegének kiszámolása
- movePoint(int i): A kapott sorszámú pont mozgatása
- calculateFitness(): A fitnesszek kiszámolása
- isEverythingDed(): Megvizsgálja, hogy minden pont célba ért-e/halott-e
- generateChildren(): Megkeresi a legjobb pontot, ez mindenképpen öröklődik. Ezután kiszámolja a fitnesszek összegét és létrehozza az új generációt.
- selectParent(): A fitnessSum alapján random kiválasztja, hogy melyik pont fog öröklődni. Minél nagyobb egy pont fitnessze, annál nagyobb az esély, hogy öröklődik.
- findBest(): lineáris keresés a legjobb pont megtalálásához
- mutateChildren(): Az új generáció mutálása (ez akadályozza meg a fejlődés elakadását)

Obstacle

A pályaelemek osztálya

- pos1: Az elem bal felső sarka
- pos2: Az elem bal alsó sarka

- contains(double x, double y): Megnézi, hogy a kapott koordináta az akadály területén belül van-e

Map

A pálya

- obstacles: Obstacle típusokat tartalmazó ArrayList. Az első eleme mindig a cél, a többi az akadályok.

Display

A program grafikus felülete.

- canvas: A rajzfelület, ami minden mozgás után újrarajzolódik.
- jp: egy JPanel, ami tartalmazza a canvas-t és a gombot.
- saveButton: a mentésre használt gomb
- savePressed: Meg let-e nyomva a saveButton
- SaveButtonActionListener: Ha megnyomjuk a gombot, átírja a savePressed-et, illetve a "kikapcsoláskor" tesz egy elválasztást a fájlba, ahova mentünk.

- initGame(): A program megjelenítése a JFrame-en
- paint(Graphics g, Point p): A kapott pontot kirajzolja g-re

Application

Az applikáció Main-je.

-drawObstacles(Graphics g, Map m): A kapott térképet felrajzolja g-re

A program:

Létrehozza a szükséges classokat, beolvassa az adatokat a fájlból, majd létrehozza a pályát, illetve a generációt. A ciklus belsejében:

Amíg nem végzett/halt meg minden pont, addig mozgatja őket, és buffereléssel kirajzolja a canvasra. Ha minden pont végzett, létrehozza a következő generációt és újakezdi a ciklust. Ha a Start/Stop Save gombot megnyomjuk, minden generáció végén kiírja egy fájlba az adott generáció számát és a minimális lépésszámot.

Class Diagram

