NHF – Pac-Man

Programozás alapjai 3.

Imets Ákos István
D7H8G6
2021. október 31.

---

1. A program célja

A feladat a Pac-Man nevű játék elkészítése. A játék egy labirintusban játszódik, ahol a játékos egy sárga figurát irányít, célja, hogy a pályán lévő pontokat megegye, és elkerülje az ellenséges szellemeket.

2. Use-case-ek

- A program elindításakor megnyílik a főmenü ahol a játékos megadja a nevét. Ezt követően elindul a játék. A program megjeleníti a játékteret, az aktuális pontszámot valamint az életet.

- A szellemeknek és Pac-Man-nek fix kezdőpontja van. A játékos a nyilak segítségével irányítja Pac-man-t (balra, jobbra, fel, le). Ha Pac-Man megeszik egy pontot, tízzel nő a pontszám.

- A szellemek üldözik Pac-man-t a sajátos stratégiájuk szerint, ha sikerül elkapniuk, Pac-man veszít egy életet, a játék a karakterek kezdőpontjából folytatódik tovább. Ha Pac-Man minden életét elveszíti vége a játéknak, a program visszakerül a főmenübe.

- Szellemek stratégiája: Blinky mindig Pac-Man-t üldözi, Pinky targetje mindig 4 blokkal Pac-Man előtt van, Inky targetje Blinky pozíciójának Pac-Man-re vett középpontos tükörképe, Clyde pedig szintén Pac-Man-t üldözi, azonban ha 12 blokkon belül van hozzá, akkor a saját „sarka” lesz a célpont.

A játék befejeztével a program fájlba menti a játékosok nevét és pontszámát, így a következő indításnál is elérhető lesz a ranglista.

3. Felhasználói útmutató

A programot elindítva a játékos a menübe kerül. Itt lehetősége van megadni a nevét és elindítani a játékot, valamint megtekinteni a top 10 játékos listáját, innen a vissza nyílra kattintva kerülhet vissza a menübe.

A játék a Pac-Man szabályai szerint zajlik, a játékos a nyilakkal irányíthatja Pac-Man-t.

Ha elfogy Pac-Man összes élete, a játékos veszít, ezt a játék egy „Game Over” felirattal jelzi. A menübe a felirat alatt található „Menu” gombbal lehet ismét eljutni.

Fontosabb osztályok és fontosabb függvényeik

Class MenuFrame

actionPerformed

```
public void actionPerformed(ActionEvent e)
```

A megfelelő gomb megnyomására elindul a játék, vagy megnyílik a top 10 játékos listája

Class MapPanel

collision

```
public void collision(Ghost ghost)
```

A függvény dönti el, hogy szellemmel való ütközés esetén mi fog történni

Parameters:

`ghost` - az szellem amivel Pac Man ütközött

endGame

```
public boolean endGame()
```

Ellenörzi Pac Man életét, ennek megfelelően igazzal vagy hamissal tér vissza

Returns:

Igazzal vagy hamissal tér vissza

paint

```
public void paint(Graphics g)
```

Overrides:

`paint` in class `JComponent`

generateMap

```
public void generateMap()
```

Ez a függvény generálja a mezőket, valamint beállítja a mező szomszédait

keyPressed

```
public void keyPressed(KeyEvent e)
```

A nyilakkal való irányításért felelős függvény

Specified by:

`keyPressed` in interface `KeyListener`

Class MovingEntity

respawn

```
public void respawn()
```

A függvény visszahelyezi az entitást a kiinduló poziciójára

onField

```
public int onField(ArrayList<Field> list)
```

A függvény megnézi, hogy melyik mezőn áll a karakter

Parameters:

`list` - mezők listája

Returns:

a mező indexét

canTurn

```
public boolean canTurn(ArrayList<Field> list,

Direction dir)
```

A függvény vizsgálja, hogy az adott mezőn elkanyarodhat-e a karakter

Parameters:

`list` - mezők listája

`dir` - a karakter aktuális iránya

Returns:

Igazzal vagy hamissal tér vissza

canTurn

```
public boolean canTurn(ArrayList<Field> list,

Direction dir, index)
```

eatPellet

```
public int eatPellet(int index,

ArrayList<Field> list)
```

move

```
public void move(ArrayList<Field> list)
```

A karakterek mozgásáért felelős függvény

Parameters:

`list` - , mezők listája

changeDirection

```
public void changeDirection(Direction dir)
```

Az iránynak megfelelő képet állítja be a karaktereknek

Parameters:

`dir` –

Class PacMan

respawn

```
public void respawn()
```

Description copied from class: `MovingEntity`

A függvény visszahelyezi az entitást a kiinduló poziciójára

Overrides:

`respawn` in class `MovingEntity`

hasPower

```
public boolean hasPower()
```

getPowerDuration

```
public int getPowerDuration()
```

setPowerDuration

```
public void setPowerDuration(int powerDuration)
```

isHit

```
public boolean isHit(Ghost ghost)
```

Vizsgálja, hogy Pac Man ütközött-e egy szellemmel

Parameters:

`ghost` - a vizsgált szellem

Returns:
igazzal vagy hamissal tér vissza

death

```
public void death()
```

Csökkenti eggyel Pac Man életét és respawnolja

eat

```
public void eat(Ghost ghost)
```

Az átvett szellem respawn-nol a kiinduló helyére

Parameters:

`ghost` -

eatPellet

```
public int eatPellet(int index,

ArrayList<Field> list)
```

Eltünteti a mezőről a megevett pelletet

Overrides:

`eatPellet` in class `MovingEntity`

Parameters:
`index` - mező indexe

`list` - mezők listája

Returns:
annak a mezőnek az indexe, amelyről megették a pelletet

Class Ghost

respawn

```
public void respawn()
```

Description copied from class: `MovingEntity`

A függvény visszahelyezi az entitást a kiinduló poziciójára

Overrides:

`respawn` in class `MovingEntity`

setTargetEnt

```
public void setTargetEnt(MovingEntity target)
```

setTarget

```
public void setTarget(Position target)
```

ez a függvény gyakorlatilag Blinky üldözési módszerét írja le

Parameters:

`target` - Pac Man pozíciója

getTargetPos

```
public Position getTargetPos()
```

chase

```
public void chase()
```

shortestRoute

```
public void shortestRoute(ArrayList<Field> list,

 Direction dir)
```

Kiszámolja, hogy melyik irányba menjen a szellem, ha a legrövidebb úton akarja elérni a célját

Parameters:

`list` - , mezők listája

`dir` -

frightened

```
public void frightened()
```

Beálltja a szellemek "ijedt" képét, valamint a bázisukat állítja be célpontként

frightenedEnd

```
public void frightenedEnd()
```

isIntersection

```
public boolean isIntersection(ArrayList<Field> list,

Direction dir)
```

Vizsgálja, hogy kereszteződés-e az adott mező

Parameters:

`list` - , mezők listája

`dir` -

Returns: igazzal vagy hamissal tér vissza

isInCorner

```
public boolean isInCorner()
```

Vizsgálja, hogy a szellemek elérték-e a bázis pozíciójukat

Returns:

igazzal vagy hamissal tér vissza

move

```
public void move(ArrayList<Field> list)
```

A szellemek a játék elején elmennek a bázispozíciójukra, majd amint elérték elkezdik üldözni saját stratégia szerint Pac Man-t

Overrides:

`move` in class `MovingEntity`

Parameters:
`list` - , mezők listája

Class Blinky

respawn

```
public void respawn()
```

Description copied from class: `MovingEntity`

A függvény visszahelyezi az entitást a kiinduló poziciójára

Overrides:

`respawn` in class `Ghost`

chase

```
public void chase()
```

Overrides:

`chase` in class `Ghost`

Class Pinky

respawn

```
public void respawn()
```

Description copied from class: MovingEntity

A függvény visszahelyezi az entitást a kiinduló poziciójára

Overrides:

respawn in class Ghost

chase

```
public void chase()
```

Overrides:

chase in class Ghost

setTarget

```
public void setTarget(Position target)
```

Pinky célpontja mindig Pac Man előtti negyedik mező

Overrides:

setTarget in class Ghost

Parameters:
`target` - Pac Man pozíciója

move

```
public void move(ArrayList<Field> list)
```

Description copied from class: Ghost

A szellemek a játék elején elmennek a bázispozíciójukra, majd amint elérték elkezdik üldözni saját stratégia szerint Pac Man-t

Overrides:

move in class Ghost

Parameters:
`list` - , mezők listája

Class Inky

respawn

```
public void respawn()
```

Description copied from class: MovingEntity

A függvény visszahelyezi az entitást a kiinduló poziciójára

Overrides:

`respawn` in class `Ghost`

setTargetEnt

```
public void setTargetEnt(MovingEntity target,
```

`Ghost` targetG)

chase

```
public void chase()
```

Overrides:

`chase` in class `Ghost`

setTarget

```
public void setTarget(Position target,
```

`Position` targetG)

Inky target-je a Blinky helyzetének Pac Man-re középpontosan tükrözött pozíciója

Parameters:

`target` - Pac Man pozíciója

`targetG` - Blinky pozíciója

move

```
public void move(ArrayList<Field> list)
```

Description copied from class: `Ghost`

A szellemek a játék elején elmennek a bázispozíciójukra, majd amint elérték elkezdik üldözni saját stratégia szerint Pac Man-t

Overrides:

`move` in class `Ghost`

Parameters:
`list` - , mezők listája

Class Clyde

respawn

```
public void respawn()
```

Description copied from class: `MovingEntity`

A függvény visszahelyezi az entitást a kiinduló poziciójára

Overrides:

`respawn` in class `Ghost`

setTarget

```
public void setTarget(Position target)
```

Clyde Pac Man-t üldözi, azonban ha 12 mező sugarán belül van, akkor a bázisa lesz a célpont

Overrides:

`setTarget` in class `Ghost`

Parameters:
`target` - Pac Man pozíciója

move

```
public void move(ArrayList<Field> list)
```

Description copied from class: `Ghost`

A szellemek a játék elején elmennek a bázispozíciójukra, majd amint elérték elkezdik üldözni saját stratégia szerint Pac Man-t

Overrides:

`move` in class `Ghost`

Parameters:
`list` - , mezők listája

chase

```
public void chase()
```

Overrides:

`chase` in class `Ghost`

Class Position

distance

```
public double distance(Position targ)
```

Két pont közötti távolságot számolja

Parameters:

`targ` - célpont pozíciója

Returns: két pont közötti távolság

Osztálydiagram