Game : BugEvolution

Create by

Surasee Suwannapal

Student ID : 6204062660251 Section 5

This report is part of the subject 040613222.

Object Oriented Programming

Department of Computer Science

Faculty of Applied Sciences

King Mongkut's University of Technology North Bangkok

Semester 1 Academic Year 2019

BugEvolution

Introduction

Origin:

This project improve my skill computer programming. Improve my java language and OOP concept and GUI

Type: Java Game Project

Benefits:

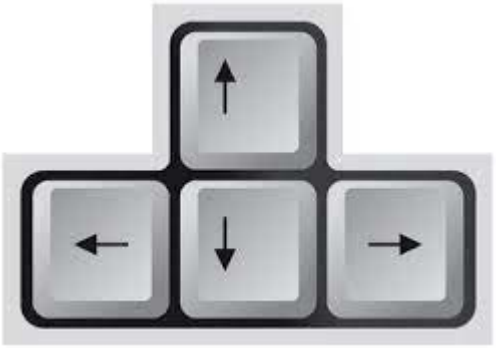
This is a game that makes players have fun and enjoyment. helps to practice meditation.

Proposal:

This is a game where we take on the role of a hero who has been evolved backwards to become a ladybug. We have to keep the all scattered energy sources. And most importantly, be careful of evil spiders, if you got catch You will become their food.

Keyblinding:

-Use arrow to move the ladybug.



-Use spacebar to stop the bug

-Use s to save game(if you save already you can load save on start page.)

How to play:

-Control the ladybug to collect all energy(score) and becareful the evil spider.

-If you collect the red energy the evil spider will become extra energy. If you collect it you will get extra score.

-Strawberry and cherry will spawn on map. If you collect it you get extra score.

-If you collect all energy in state 1 you will go to state2.

-You have 3 life point. If you lose all GameOver.

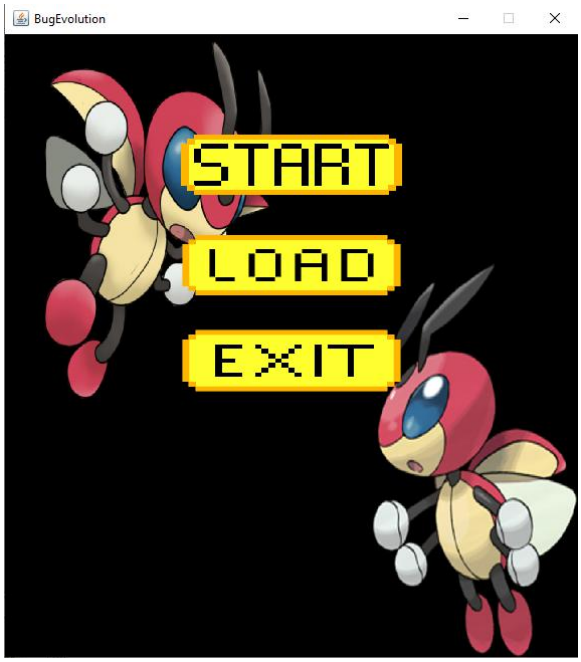
Character:

Ladybug

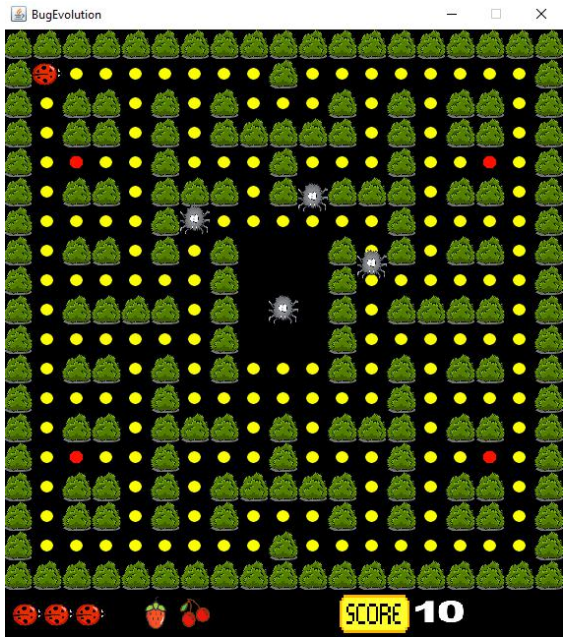
Evil spider

In Game Play:

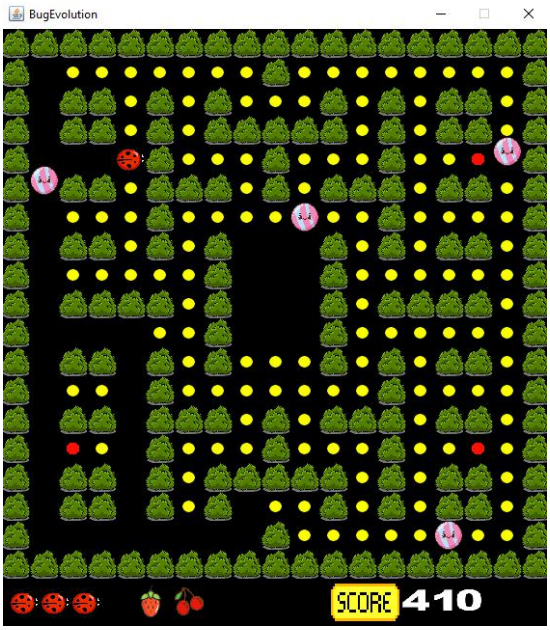
Start page



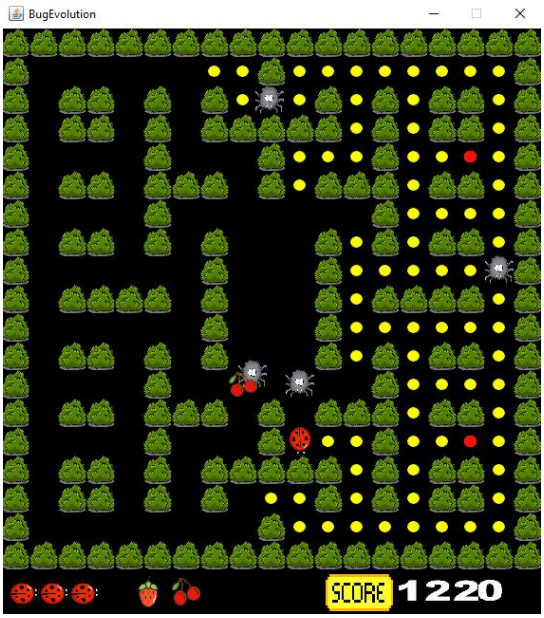
Start Game



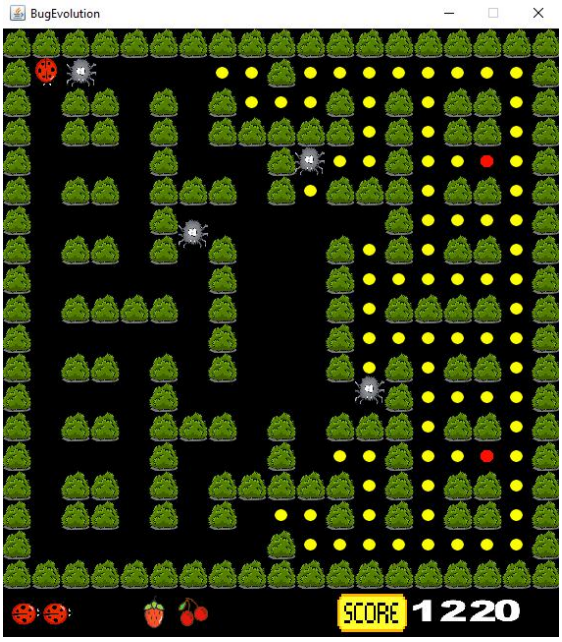
Collect red energy



Cherry and strawberry spawn

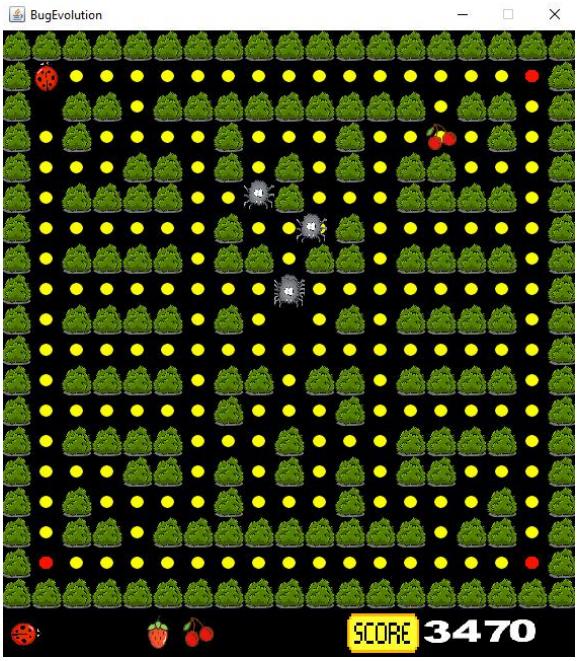


Got catch by Evil spider lost 1 life point



lose 3 life point

State 2



Working table

No.	list	1-15	16-25	16-30
1	Plan the programming of various algorithm functions.			
2	write a program			
3	Check program			
4	Document			
5	Fix problem			

Example of programs:

Constructor: create constructor and call it later

```
1
2   package elements;
3
4   import java.awt.Graphics;
5   import utils.Drawing;
6
7   public class Ball extends Element {
8
9       protected int point;
10
11      // Construtor
12      public Ball(final String imageName, final int point) {
13          super(new String[]{imageName}, 0, 2);
14          this.point = point;
15          this.isTransposable = false;
16      }
17
18      // Construtor
19      public Ball(final String imageName, final int point, final double x, final double y) {
20          super(new String[]{imageName}, 0, 2);
21          this.point = point;
22          this.isTransposable = false;
            this.setPosition(x, y);
24      }
25
26      @Override
        public void autoDraw(Graphics g) {
28          Drawing.draw(g, imageIcon, pos.getY(), pos.getX());
29      }
30  }
31
```

Encapsulation:

achieve Encapsulation

```
126
127     public void removeLife() {
128         life--;
129     }
130
131     public int getLife() {
132         return life;
133     }
134
135     public int getMovDirection() {
136         return movDirection;
137     }
138
139     public void setMovBefDirection(int movBefDirection) {
140         this.movBefDirection = movBefDirection;
141     }
142
143     public int getScore() {
144         return this.score;
145     }
146
147     public void setScore(int score) {
148         this.score = score;
149     }
150
151     public void resetScore() {
152         this.aux_score -= 10000;
153     }
154
155     public void resetTotalScore() {
156         this.aux_score = 0;
157         this.score = 0;
158     }
159
160     public void scorePoints(int points) {
161         this.score += points;
162         this.aux_score += points;
163     }
164
```

Composition:

Fruit refers point and others

```java
package elements;

import java.awt.Graphics;
import utils.Drawing;

public abstract class Fruit extends Element {

    protected int points;
    protected int duration;

    public Fruit(String iconName) {
        super(new String[]{iconName}, 0, 4);

        this.isTransposable = true;
        this.isVisible = false;
    }

    public int getPoints() {
        return points;
    }

    public int getDuration() {
        return duration;
    }

    public void decrementDuration() {
        this.duration--;
    }

    @Override
    public void autoDraw(Graphics g) {
        if (isVisible) {
            Drawing.draw(g, this.imageIcon, pos.getY(), pos.getX());
        }
    }
}

```

Polymorphism:

Subclasses of elements is wall

```
1
2   package elements;
3
4   import java.awt.Graphics;
5   import utils.Drawing;
6
7   public class Wall extends Element {
8
9       public Wall(String image, double x, double y) {
10          super(new String[]{image}, 0, 5);
11          this.isVisible = true;
12          this.isTransposable = false;
            this.setPosition(x, y);
14      }
15
16      @Override
        public void autoDraw(Graphics g) {
18          Drawing.draw(g, imageIcon, pos.getY(), pos.getX());
19      }
20
21  }
22
```

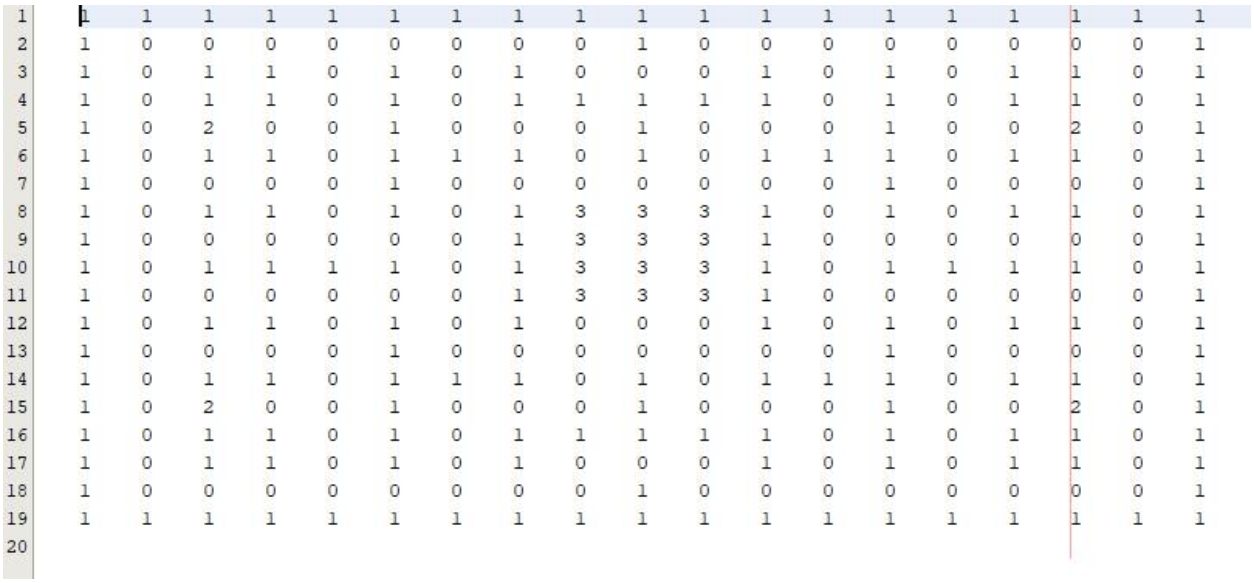
Abstract:

```
15
16      protected ImageIcon[] directions;
17      protected ImageIcon imageIcon;
18      protected Position pos;
19      protected boolean isTransposable;
20      protected boolean isVisible;
21
22      private final int typeElement;
23
24      protected Element(String[] imageName, int dir, int typeElement) {
25          this.pos = new Position(1, 1);
26          this.isTransposable = true;
27
28          directions = new ImageIcon[imageName.length];
29
30          for (int i = 0; i < imageName.length; i++) {
31              directions[i] = getImageIcon(imageName[i]);
32          }
33
34          this.typeElement = typeElement;
35
36          setImageIcon(dir);
37      }
```

Inheritance:

```
10  public class StageGameOver extends Stage {
11
12      private Image imgStart;
13      private Image imgExit;
14      private Image background;
15
16      public StageGameOver() {
17          try {
18              this.imgStart = Toolkit.getDefaultToolkit().getImage(
19                      new java.io.File(".").getCanonicalPath() + Consts.PATH + "button_start.png");
20              this.imgExit = Toolkit.getDefaultToolkit().getImage(
21                      new java.io.File(".").getCanonicalPath() + Consts.PATH + "button_exit.png");
22              this.background = Toolkit.getDefaultToolkit().getImage(
23                      new java.io.File(".").getCanonicalPath() + Consts.PATH + "background_game_over.jpg");
24          } catch (IOException e) {
25              System.err.println("Error GO\n " + e.getMessage());
26          }
27      }
28
29      @Override
30      public void paintScene(Graphics g) {
31          int aux = Consts.CELL_SIZE * Consts.NUM_CELLS;
32          g.fillRect(0, 0, aux, aux + 50);
33          g.drawImage(background, 0, 0, aux, aux + 50, null);
34          g.drawImage(imgStart, (aux / 2) - 210, 350, 200, 60, null);
35          g.drawImage(imgExit, (aux / 2) + 10, 340, 200, 70, null);
36      }
37
38      @Override
39      protected void drawSceneFinal() {
40      }
41
42  }
43
```

Use map from text file



1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
2	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1
3	1	0	1	1	0	1	0	1	0	0	0	1	0	1	0	1	1	0	1
4	1	0	1	1	0	1	0	1	1	1	1	1	0	1	0	1	1	0	1
5	1	0	2	0	0	1	0	0	0	1	0	0	0	1	0	0	2	0	1
6	1	0	1	1	0	1	1	1	0	1	0	1	1	1	0	1	1	0	1
7	1	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	1
8	1	0	1	1	0	1	0	1	3	3	3	1	0	1	0	1	1	0	1
9	1	0	0	0	0	0	0	1	3	3	3	1	0	0	0	0	0	0	1
10	1	0	1	1	1	1	0	1	3	3	3	1	0	1	1	1	1	0	1
11	1	0	0	0	0	0	0	1	3	3	3	1	0	0	0	0	0	0	1
12	1	0	1	1	0	1	0	1	0	0	0	1	0	1	0	1	1	0	1
13	1	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	1
14	1	0	1	1	0	1	1	1	0	1	0	1	1	1	0	1	1	0	1
15	1	0	2	0	0	1	0	0	0	1	0	0	0	1	0	0	2	0	1
16	1	0	1	1	0	1	0	1	1	1	1	1	0	1	0	1	1	0	1
17	1	0	1	1	0	1	0	1	0	0	0	1	0	1	0	1	1	0	1
18	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1
19	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
20																			

Problem

This project have some problem. When I start creating program. I create an orderly and call it step by step. Save and load function have some problem with bug animation. State 3 did not finish. Export to jar file images no found. I think imgs folder is no in src folder.