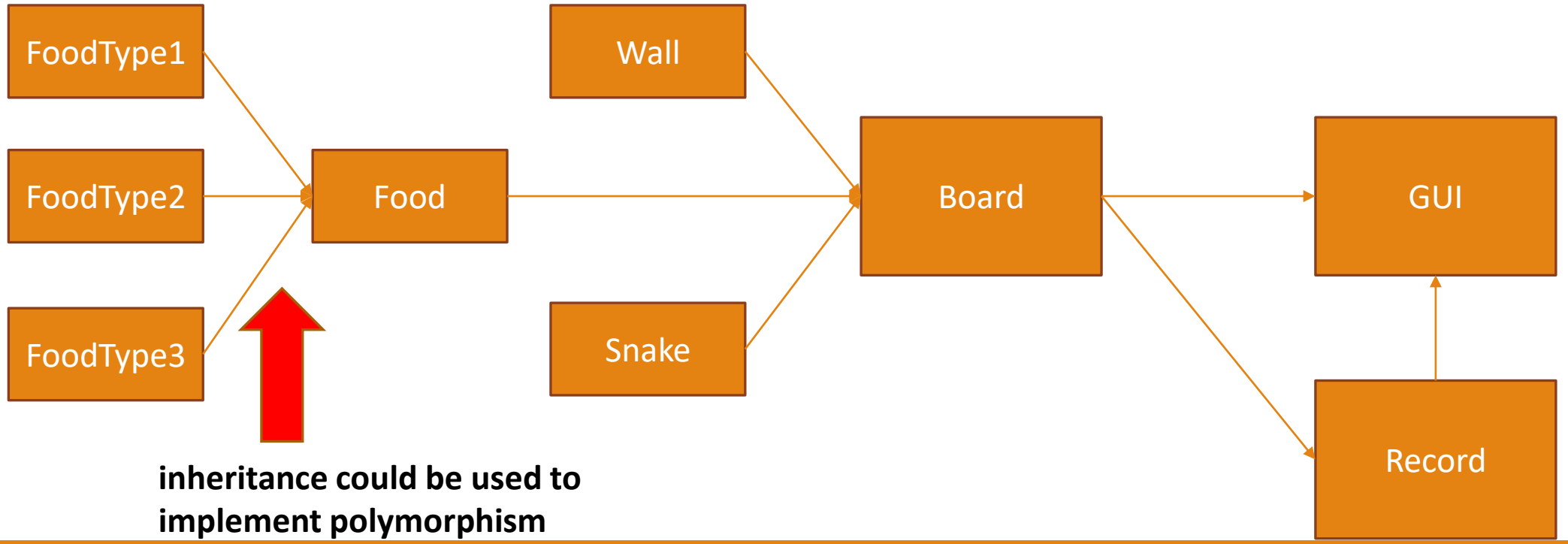
Report and Testing Output

Report: describes the structure and functionality of my system

Testing: demonstrate that the features of my game

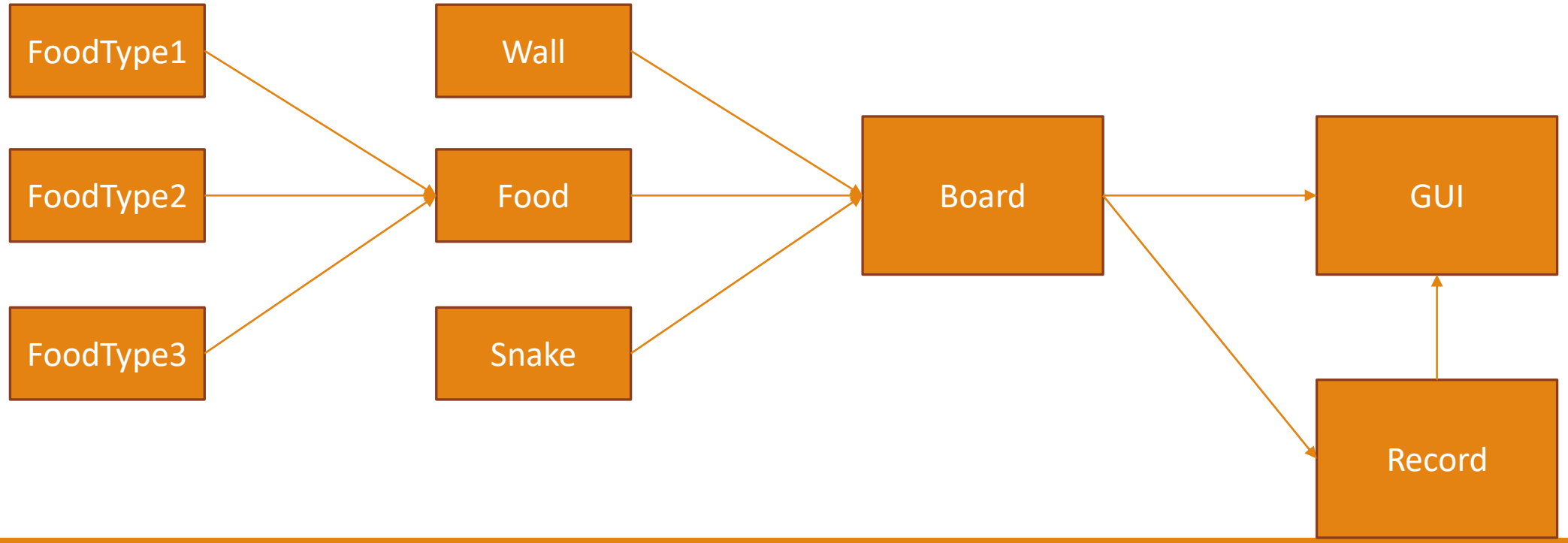
Report

The structure of my game:



Report

There are several entity classes, Wall, Food, Snake, Board and Record.



Report

Board: provide the place for snake to move and eat, it could be called as map or field for this game.

Wall: is the barrier that snake should prevent from hitting.

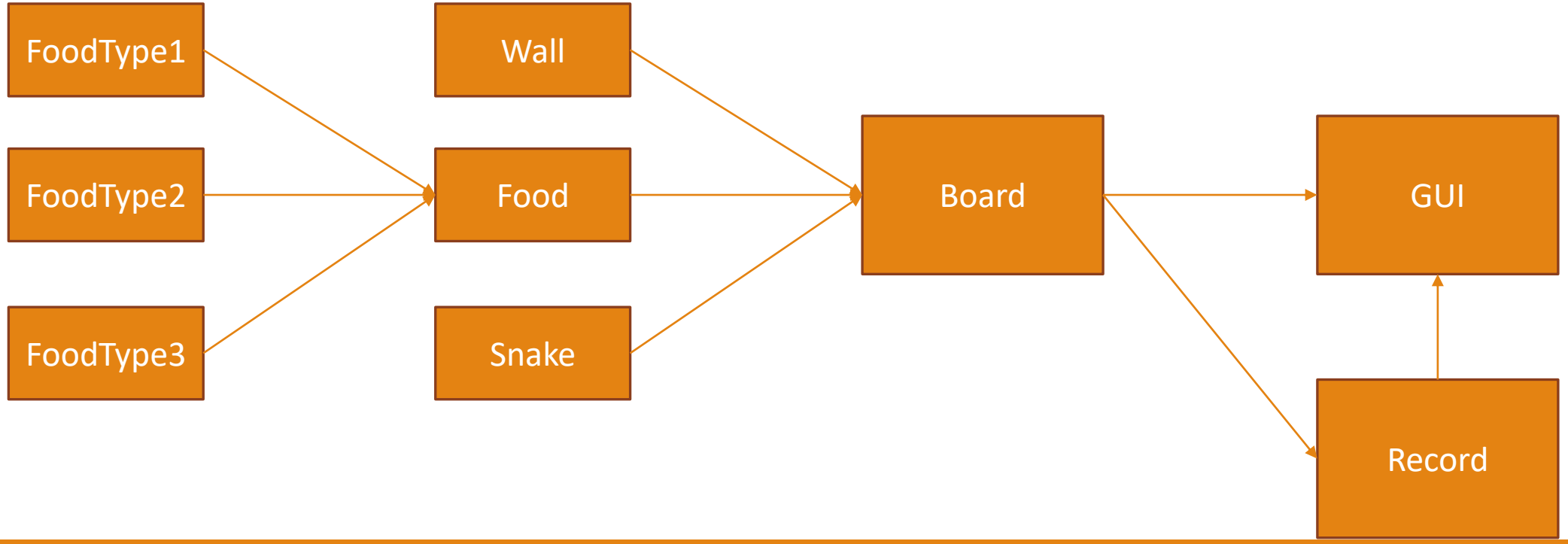
Food, is the reward that could increase score when snake eating it. There are 3 kinds of food that rewards differently and appear with different possibilities.

Snake: moves on the board (or map), it should try to eat more food and prevent from hitting walls or it self.

Record: store the data for every playing and provide ranking information.

Report

GUI class is to show the game in a friendly interface.



Report

GUI class:

there are several important features for this class,

1) show the Board as a 2D map
2) draw the Walls as blocks
3) draw the Food as a distinguished block
4) draw snake as a consecutive blocks
5) handle the controller such as keyboard
6) draw how many score gained
7) show the ranking when game over

Report and Testing Output

Report: describes the structure and functionality of my system

Testing: demonstrate that the features of my game

Testing

Main features:
1, Board that is a 2D-array

2, Barriers (or called Walls) that are blocks

3, Food that may be eaten and randomly be generated again

4, Snake that consists of several blocks and move on the board

5, Associations of the above entities

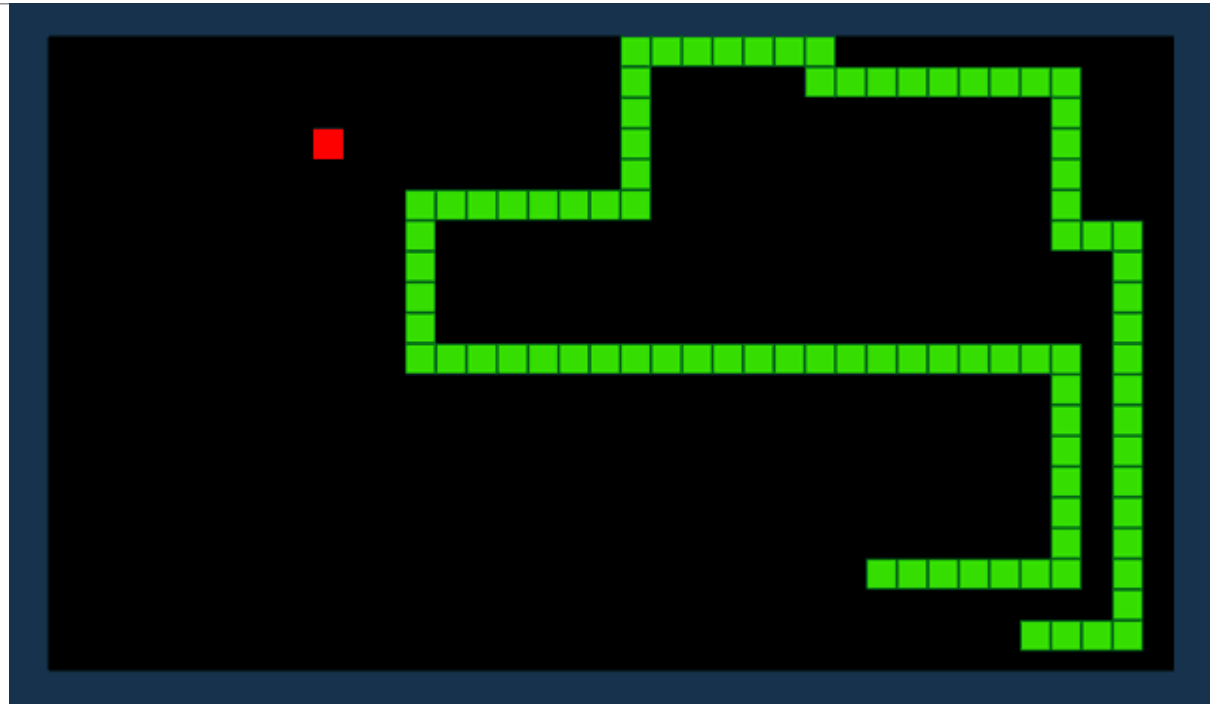


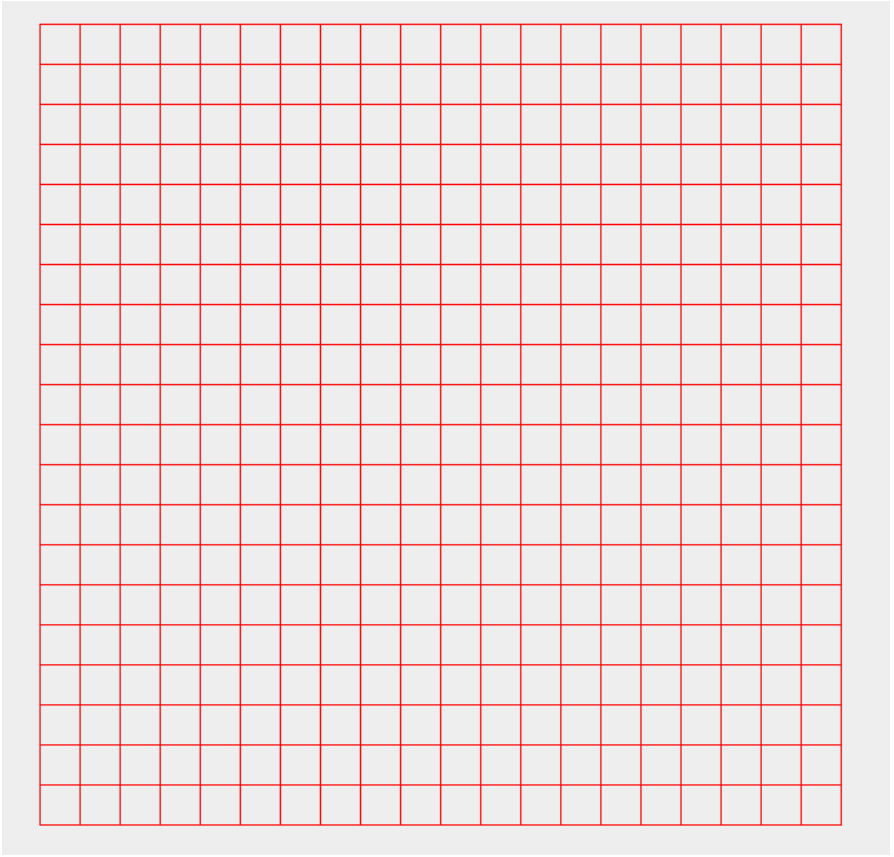
Image From Google:
https://www.google.com/url?sa=i&source=images&cd=&ved=2ahUKEwjCtcX99IvmAhWRnp4KHTiiDjIQjRx6BAgBEAQ&url=https%3A%2F%2Fwww.coolmathgames.com%2F0-snake&psig=AOvVaw0Z1ZZQgKu_OwHPYeAY3ZOi&ust=1574996401646971

Testing - Board

Obviously, it is a 2D array

As suggested in the assignment, “An example grid of squares could be 20 x 20”,

Here, Board could be defined as int [20][20]



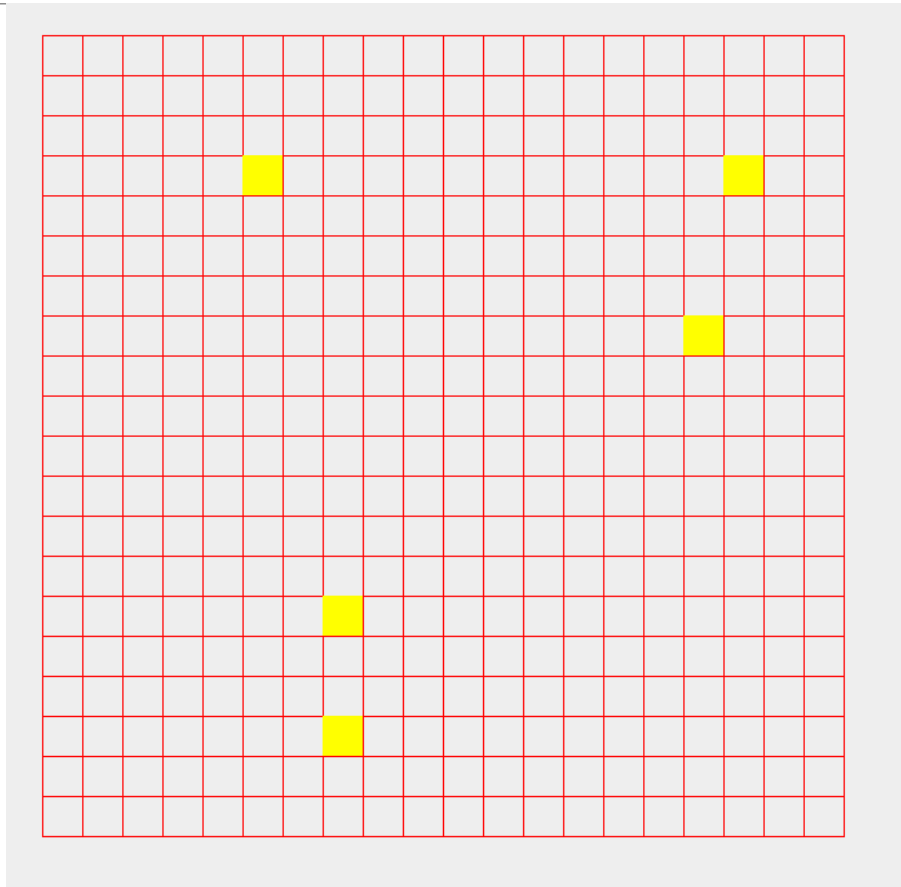
GUI Testing: 2D Array, 20 * 20

Testing - Wall

Wall is the barrier, on the Board there could be several more walls.

As the rule of this game, snake could not move over the walls, and will die when hitting the wall.

Wall could be a block, or grid of the Board



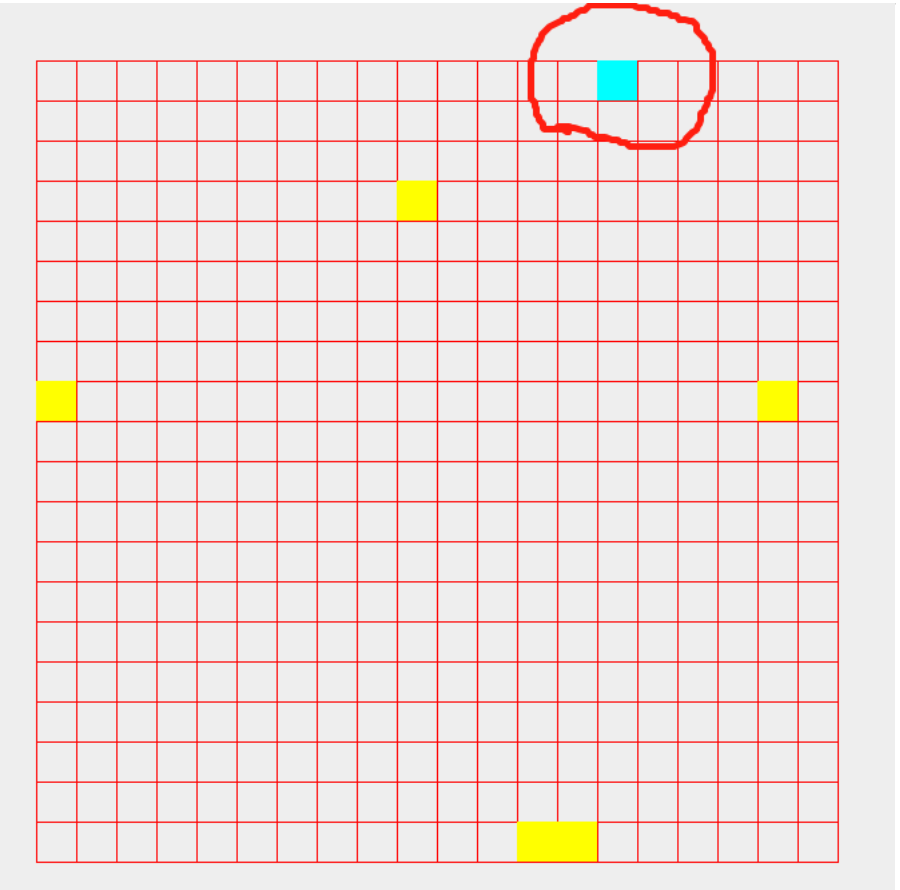
GUI Testing: Walls are the yellow ones

Testing - Food

Wall is the barrier, on the Board there could be several more walls.

As the rule of this game, snake could not move over the walls, and will die when hitting the wall.

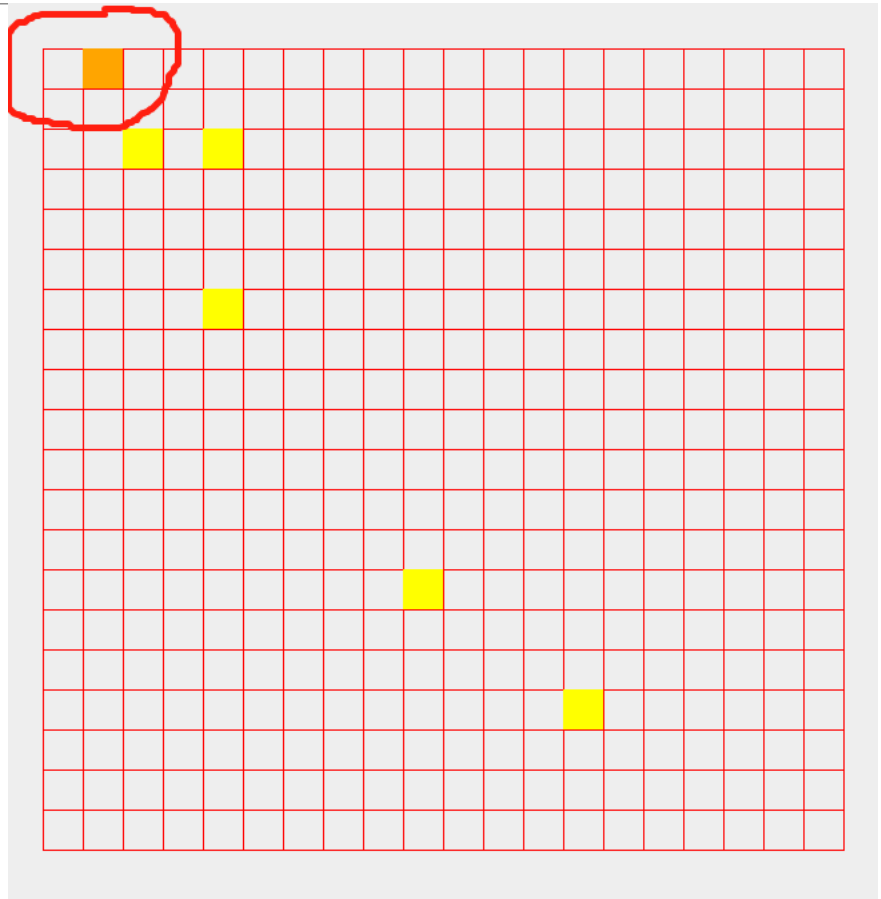
Wall could be a block, or grid of the Board



GUI Testing: The light blue one is Food

Testing – Food Continued

To make the game more interesting, there are more than one kind of Food, different Food has different colors, with different scores when snake eating it.



GUI Testing: The orange one is also Food

Testing – Snake

Snake could move, eat and may hit wall and itself.
Snake is started with only one block that is the head.

To move,

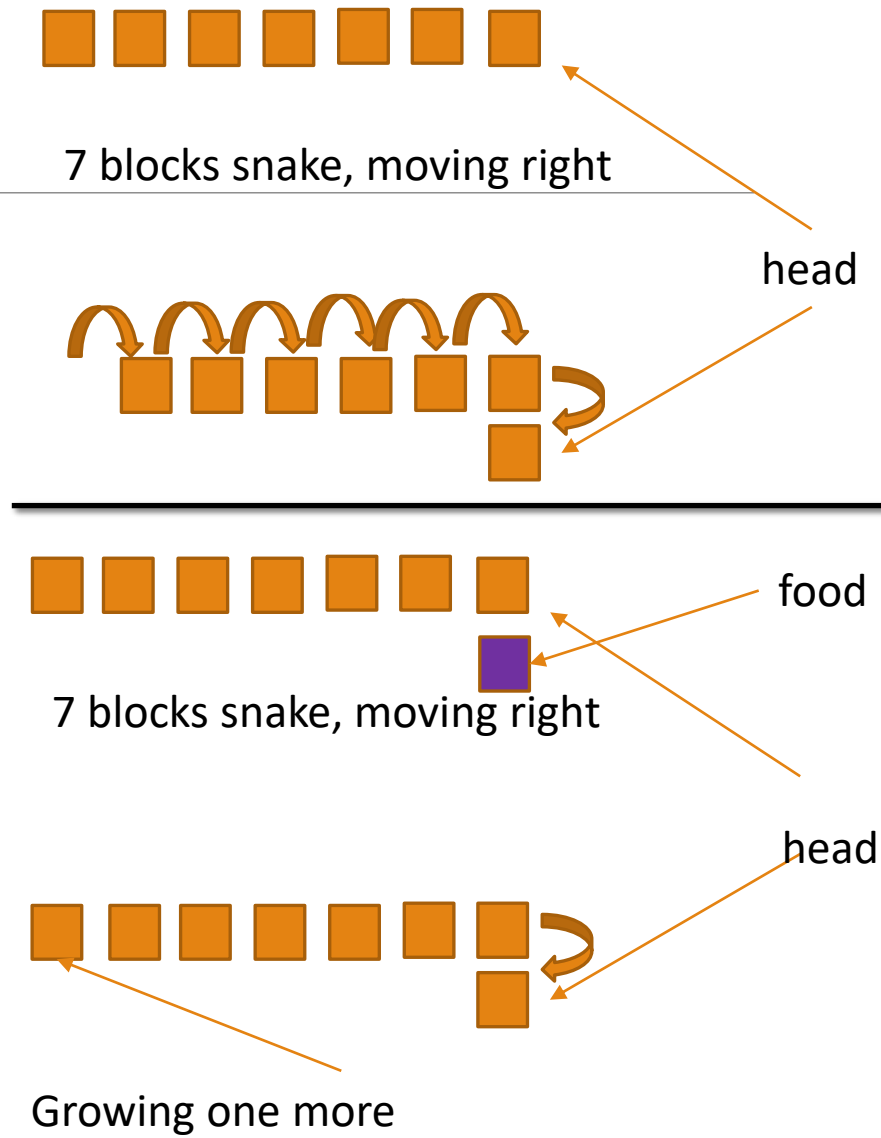
Every time snake moves, the head move to a new position while the following blocks move to the position of previous one.

To eat Food,

when the head of snake hit the Food, the snake will grow with one block adding on the tail of snake.

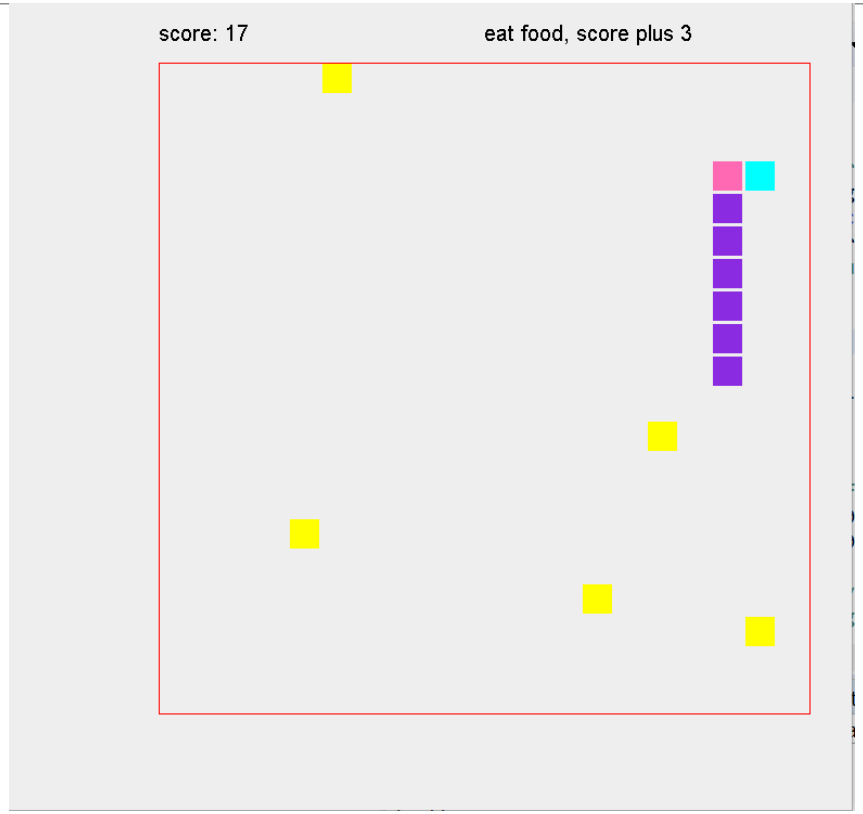
Simultaneously, the score will go up, with different food rewarding different scores.

When hitting wall or itself, game over.

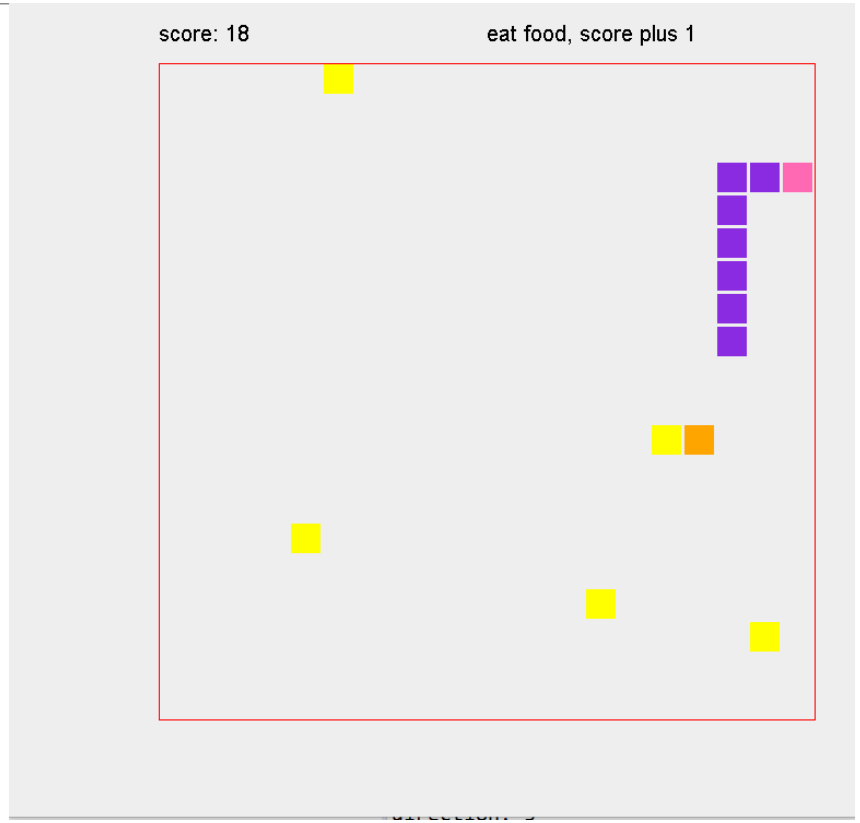


Testing – Snake Continued

This two screenshots show that how snake moves, eats food and grow



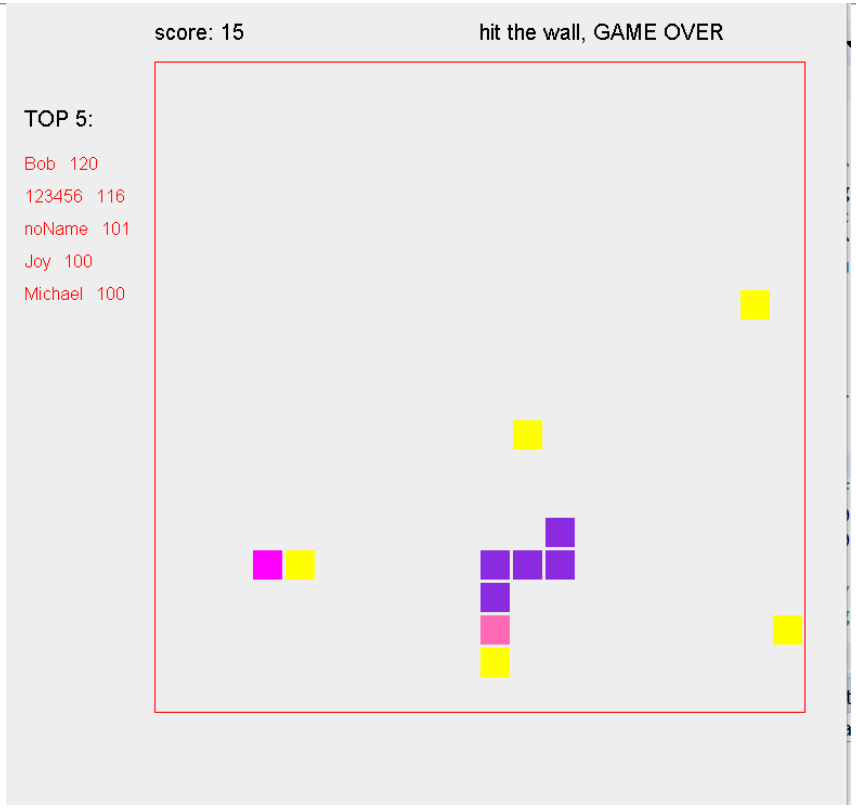
GUI Testing: The light blue is food, after eating, snake grows



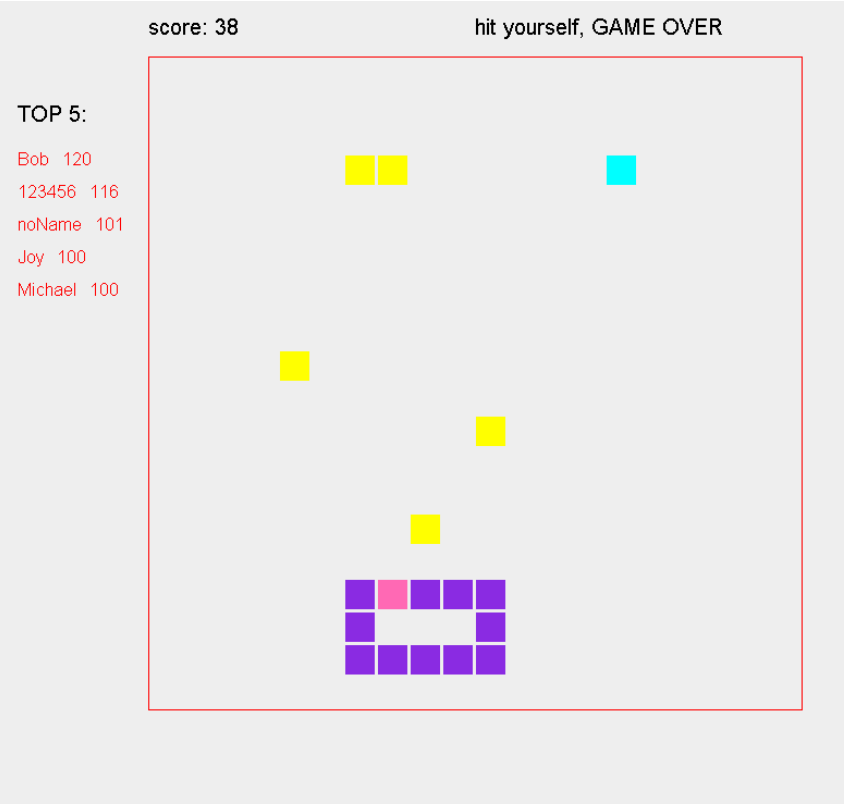
GUI Testing: after eating, a new food appear as the orange one

Testing – Snake Continued

This two screenshots show that when snake hits walls or itself



GUI Testing: when hitting board, GAME OVER



GUI Testing: when hitting itself, GAME OVER

Testing – Associations

Board tend to be static, and initialized as the 2D map.

Add a list of walls into the Board.

Generate Food on Board. (if snake eat it, then a new food will appear.)

Create a snake that is started with only one block (head).

Singleton design pattern is used for snake.

Snake.java

```
45
46      //singleton design pattern
47      private Snake() {
48          block_list = new LinkedList<Block>();
49          Block head = new Block(0, 0, index);
50          index++;
51          block_list.add(head);
52          speed = 1;
53      }
54
55      //used to get the singleton reference of snake
56      public static Snake getInstance() {
57          if(snake == null) {
58              snake = new Snake();
59          }
60          return snake;
61      }
62
```

Testing – GUI Controller

Keyboard is used for controlling the move of snake.

To be more specific, it is:

UP, ↑

DOWN, ↓

LEFT, ←

RIGHT, →

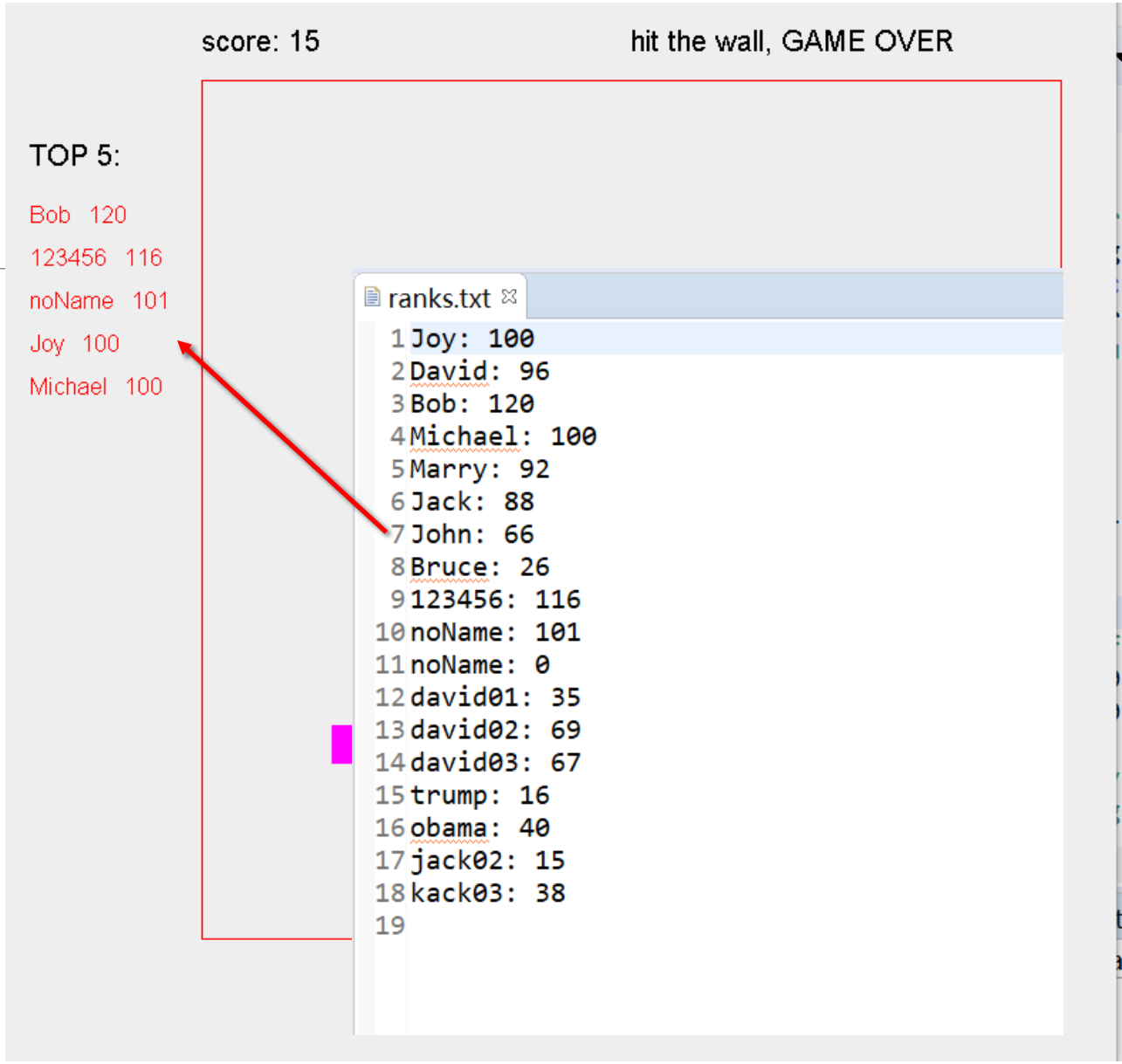
WindowShow.java

```
//add keyboard event listener
this.addKeyListener(new KeyListener() {

    @Override
    public void keyTyped(KeyEvent e) {
        // TODO Auto-generated method stub

    }

    @Override
    public void keyReleased(KeyEvent e) {
        // TODO Auto-generated method stub

    }

    @Override
    public void keyPressed(KeyEvent e) {
        // TODO Auto-generated method stub
        int dir = Board.snake.getDirection();
        switch(e.getKeyCode()) {
        case KeyEvent.VK_UP:
            if(dir == 0) {
                Board.snake.moveForward();
                repaint();
            }else if(dir == 1) {

            }else if(dir == 2) {
                Board.snake.turnRight();
                repaint();
            }else if(dir == 3) {
                Board.snake.turnLeft();
```

Testing - Recording

Every time, when running the game, the gained score will be recorded and top 5 will be displayed.

A txt file is needed here to be used as storage.



Thank you!