**CSCE A385 Programming Assignment 1**

Computer Graphics

Shanan Almario

**Program Notes**

*Program Compilation and Minor Errors*

*Assuming you’re using Visual Studios*

* If there is a collision before the fly even flies, probably clean the solution or try to run it again.
* Sometimes there will be an error that will force a break. When that happens, clean the solution and try to run it again.

*Variable Description*

exit1, exit2: these integers describes where the randomly placed exit will be located

xCoord, yCoord: these doubles hold the x and y coordinates for where the path ends

roomData: this array holds all 16 rooms so that their data can be accessed easily

currentRoom: this variable keeps track of which room the fly is currently inside

*Room Class Description*

Each square on the grid is called a room. This class stores the max and min coordinates as well as the existence of wall edges on the sides of the square.

*Implementation and Thought process*

There is a 4x4 grid where the fly will roam until it hits a wall. The nature of each square is described by the Room class which tells the user the max coordinates of the room and the existence of walls on each of its sides. The coordinates are stored as doubles, and the existence of walls are given as Boolean statements. There is an array holding the information of all 16 squares where the index is (the room number ID – 1). The fly will start in a random room at a random point where it will fly around until it reaches the boundary of the square. This is done in the randomStart function where the random room number is generated. The room’s max coordinates are examined by using if statements to put the fly’s starting point inside the right room at the start. In this function, the currentRoom is also initialized. The checkNode function will check to see if there is a wall or exit at the boundary using the information in roomData. If there isn’t a wall or an exit, the current room will be changed and the coordinates will be changed to match the next room so that the fly can continue flying until it reaches a wall.

|  |  |  |  |
| --- | --- | --- | --- |
| **1** | **2** | **3** | **4** |
| **5** | **6** | **7** | **8** |
| **9** | **10** | **11** | **12** |
| **13** | **14** | **15** | **16** |

**Figure 1: Room Grid Image**

This is the layout of the rooms that the code follows. This is used to construct the code and to decide which room the fly wanders into after reaching the boundary of the previous room.

To find next room after reaching boundary:

Boundary at Max Y – fly exits to the top

newRoom = RoomNum – 4

Boundary at MinY – fly exits to the bottom

newRoom = RoomNum + 4

Boundary at Max X – fly exits to the right

newRoom = RoomNum +1

Boundary at Min X – fly exits to the left

newRoom = RoomNum – 1

**1 2 3 4**

|  |  |  |  |
| --- | --- | --- | --- |
| **17**  **20 21 22 23** | **18** | **19** |  |
| **24** | **25** | **26** |  |
| **31**  **27 28 29 30** | **32** | **33** |  |
| **38**  **34 35 36 37** | **39** | **40** |  |

**16**

**15**

**14**

**13**

**5**

**6**

**7**

**8**

**12 11 10 9**

**Figure 2: Edge Grid Image**

The red numbers are the horizontal lines, and the black numbers are vertical lines. There were numbers assigned to each edge when creating the grid. This was how each edge was numbered.