

THE A-MAZING DS4 RACE

LAB 8 A and 8 B REPORT

ARYAN RAO

SECTION 5

DATE- 11/5/2021

SUBMISSION DATE- 11/9/2021

Problem

Create a real-time game that would be controlled by the DS4 and the game consists of a maze and a character which is to be controlled using the same.

Analysis

The problem consists of various functions defined by us. We must implement mechanisms of ensuring when the character is to be moved left, right, or down.

Design

- Implement the functions
- Check for space to go left, right, or down
- Print the winning and losing messages accordingly

Testing

Made sure that the character moves when it is supposed to, and it doesn't go beyond the boundaries of the maze. It goes left, right, and down when it is supposed to.

Comments

Scanning the correct information from the DS4 is necessary. It might have an impact on the whole code if not done properly.

Questions

1. Describe how you checked if the avatar could safely move down and go left/right.

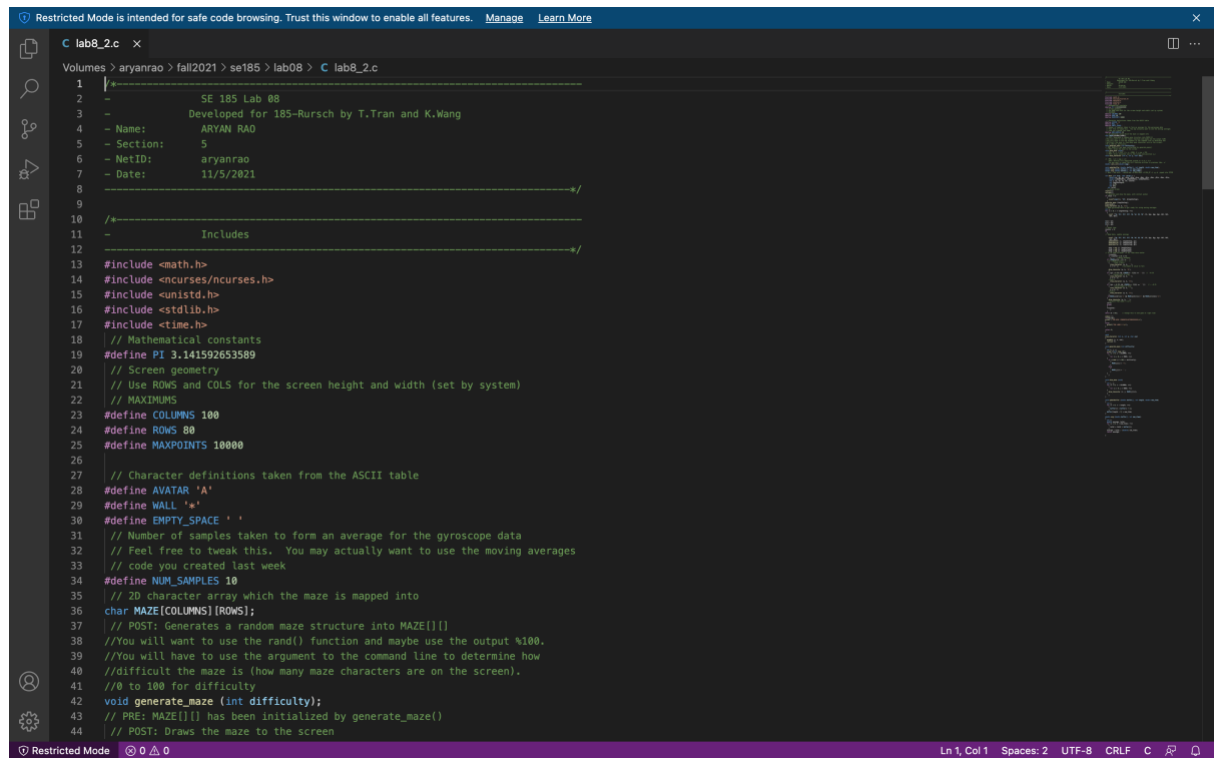
I used the If statements and applied to proper conditions which were if the character encounters the maze or a space, it moves accordingly.

2. Describe what was necessary to check for the player losing the game.

In the If statement, I checked if the avatar encountered a wall, it won't move hence the user loses the game.

SCREENSHOT

SOURCE CODE:



```
1  /*
2  - SE 185 Lab 08
3  - Developed for 185-Rursch by T.Tran and K.Wang
4  - Name: ARYAN RAO
5  - Section: 5
6  - NetID: aryanrao
7  - Date: 11/5/2021
8  */
9
10 /*-----
11 - Includes
12 -----*/
13 #include <math.h>
14 #include <ncurses/ncurses.h>
15 #include <unistd.h>
16 #include <stdlib.h>
17 #include <time.h>
18 // Mathematical constants
19 #define PI 3.141592653589
20 // Screen geometry
21 // Use ROWS and COLS for the screen height and width (set by system)
22 // MAXIMUMS
23 #define COLUMNS 100
24 #define ROWS 80
25 #define MAXPOINTS 10000
26
27 // Character definitions taken from the ASCII table
28 #define AVATAR 'A'
29 #define WALL '*'
30 #define EMPTY_SPACE ' '
31 // Number of samples taken to form an average for the gyroscope data
32 // Feel free to tweak this. You may actually want to use the moving averages
33 // code you created last week
34 #define NUM_SAMPLES 10
35 // 2D character array which the maze is mapped into
36 char MAZE[COLUMNS][ROWS];
37 // POST: Generates a random maze structure into MAZE[][]
38 //You will want to use the rand() function and maybe use the output %100.
39 //You will have to use the argument to the command line to determine how
40 //difficult the maze is (how many maze characters are on the screen).
41 //0 to 100 for difficulty
42 void generate_maze (int difficulty);
43 // PRE: MAZE[][] has been initialized by generate_maze()
44 // POST: Draws the maze to the screen
```

```
Restricted Mode is intended for safe code browsing. Trust this window to enable all features. Manage Learn More

C lab8_2.c x
Volumes > aryanrao > fall2021 > se185 > lab08 > C lab8_2.c
39 //You will have to use the argument to the command line to determine how
40 //difficult the maze is (how many maze characters are on the screen).
41 //0 to 100 for difficulty
42 void generate_maze (int difficulty);
43 // PRE: MAZE[][] has been initialized by generate_maze()
44 // POST: Draws the maze to the screen
45 void draw_maze (void);
46 // PRE: 0 < x < COLS, 0 < y < ROWS, 0 < use < 255
47 // POST: Draws character use to the screen and position x,y
48 void draw_character (int x, int y, char use);
49
50 /* PRE: -1.0 < mag < 1.0
51    POST: Returns tilt magnitude scaled to -1.0 -> 1.0
52    You may want to reuse the roll function written in previous labs. */
53 double calc_roll(double mag);
54
55 void updatebuffer (double buffer[], int length, double new_item);
56 double avg (double buffer[], int num_items);
57 double avg (double buffer[], int num_items);
58 // Main - Run with './ds4rd.exe -d 054c105c4 -0 DS4_BT -t -g -b piped into STDIN
59
60 int main (int argc, char *argv[]) {
61     double gx, gy, gz, xAvg, yAvg, zAvg, xMax, xMin, yMax, yMin, zMax, zMin;
62     double x[MAXPOINTS], y[MAXPOINTS], z[MAXPOINTS];
63     int t, b1, b2, b3, b4, counter;
64     int lengthofavg=0;
65     int a=39;
66     int b=0;
67     int lol=0;
68     // setup screen
69     initscr();
70     refresh();
71     // Generate and draw the maze, with initial avatar
72     if (argc > 1)
73     {
74         sscanf(argv[1], "%d", &lengthofavg);
75     }
76     generate_maze (lengthofavg);
77     draw_maze();
78     draw_character (a, b, 'A');
79     // Read gyroscope data to get ready for using moving averages.
80     int i;
81     for (i = 0; i < lengthofavg; ++i)
82     {
```

```
Restricted Mode is intended for safe code browsing. Trust this window to enable all features. Manage Learn More

C lab8_2.c x
Volumes > aryanrao > fall2021 > se185 > lab08 > C lab8_2.c
79 // Read gyroscope data to get ready for using moving averages.
80 int i;
81 for (i = 0; i < lengthofavg; ++i)
82 {
83     scanf ("%d, %lf, %lf, %lf, %d, %d, %d, %d", &t, &gx, &gy, &gz, &b1, &b2,
84           &b3, &b4);
85
86
87     x[i] = gx;
88     y[i] = gy;
89     z[i] = gz;
90 }
91 // Event loop
92 counter = 0;
93 do
94 {
95     // Read data, update average
96
97     scanf ("%d, %lf, %lf, %lf, %d, %d, %d, %d", &t, &gx, &gy, &gz, &b1, &b2,
98           &b3, &b4);
99     updatebuffer (x, lengthofavg, gx);
100    updatebuffer (y, lengthofavg, gy);
101    updatebuffer (z, lengthofavg, gz);
102
103    xAvg = avg (x, lengthofavg);
104    yAvg = avg (y, lengthofavg);
105    zAvg = avg (z, lengthofavg);
106    // Is it time to move? if so, then move avatar
107    ++counter;
108    if (counter % 15 == 0)
109    {
110        //delay movement
111        if ((MAZE[a][b + 1]) == ' ')
112        //if ((MAZE[a+1][b]) == ' ')
113        {
114            //Can I fall?
115            draw_character (a, b, ' ');
116            b = b + 1; //increment b value to fall
117            draw_character (a, b, 'A');
118        }
119        if ((gx > 0.25) && ((MAZE[a - 1][b] == ' '))) // >0.25
120        {
121            //moves Left if true
122            draw_character (a, b, ' ');
123            a = a - 1;
124            //b=b-1;
```

```
Restricted Mode is intended for safe code browsing. Trust this window to enable all features. Manage Learn More

C lab8_2.c x
Volumes > aryanrao > fall2021 > se185 > lab08 > C lab8_2.c

122 //b=b-1;
123 draw_character (a, b, 'A');
124 }
125 if ((gx < -0.25) && ((MAZE[a + 1][b] == ' '))) // < -0.25
126 { //moves Right if true
127 draw_character (a, b, ' ');
128 a = a + 1;
129 //b=b+1;
130 draw_character (a, b, 'A');
131 }
132 if (MAZE[a][b+1]=='*' && MAZE[a][b-1]=='*' && MAZE[a+1][b]=='*')
133 {
134 draw_character (a, b, ' ');
135 //printf("YOU LOSE!!!\n");
136 lolol;
137 break;
138 }
139 ++counter;
140 }
141 }
142 while (b < 84); // Change this to end game at right time
143
144 endwin ();
145 if (lol==0){
146 printf ("YOU WIN! CONGRATULATIONSSSSSS\n");
147 }
148 else{
149 printf ("YOU LOSE!!!\n");
150 }
151
152 return 0;
153 }
154
155 void
156 draw_character (int x, int y, char use)
157 {
158 mvaddch (y, x, use);
159 refresh ();
160 }
161
162 void generate_maze (int difficulty)
163 {
164 int i, j, r;
165 srand ((int) time (0));
166
```

```
Restricted Mode is intended for safe code browsing. Trust this window to enable all features. Manage Learn More

C lab8_2.c x
Volumes > aryanrao > fall2021 > se185 > lab08 > C lab8_2.c

159 refresh ();
160 }
161
162 void generate_maze (int difficulty)
163 {
164 int i, j, r;
165 srand ((int) time (0));
166 for (i = 0; i < COLUMNS; ++i)
167 {
168 for (j = 0; j < ROWS; ++j)
169 {
170 if ((rand () % 101) < difficulty)
171 {
172 MAZE[j][i] = '*';
173 }
174 else
175 {
176 MAZE[j][i] = ' ';
177 }
178 }
179 }
180 }
181
182 void draw_maze (void)
183 {
184 int i, j;
185 for (i = 0; i < COLUMNS; ++i)
186 {
187 for (j = 0; j < ROWS; ++j)
188 {
189 draw_character (i, j, MAZE[j][i]);
190 }
191 }
192 }
193
194 void updatebuffer (double buffer[], int length, double new_item)
195 {
196 int i;
197 for (i = 0; i < length; ++i)
198 {
199 buffer[i] = buffer[i + 1];
200 }
201 buffer[length - 1] = new_item;
202 }
```

Restricted Mode is intended for safe code browsing. Trust this window to enable all features. Manage Learn More

C lab8_2.c

Volumes > arianrao > fall2021 > se185 > lab08 > C lab8_2.c

```
190     }
191     }
192 }
193
194 void updatebuffer (double buffer[], int length, double new_item)
195 {
196     int i;
197     for (i = 0; i < length; ++i)
198     {
199         buffer[i] = buffer[i + 1];
200     }
201     buffer[length - 1] = new_item;
202 }
203
204 double avg (double buffer[], int num_items)
205 {
206     int i;
207     double average, total;
208     for (i = 0; i < num_items; ++i)
209     {
210         total = total + buffer[i];
211     }
212     average = total / (double) num_items;
213     return average;
214 }
215
216
```

Ln 1, Col 1 Spaces: 2 UTF-8 CRLF C

OUTPUT:

