

Table of contents

1	<u>C++ Files</u>
1.1	<u>MainGameLoop.cpp</u>
1.2	<u>CreateObject.cpp</u>
1.3	<u>DrawWindows.cpp</u>
1.4	<u>ImportMaps.cpp</u>
1.5	<u>Interactions.cpp</u>
1.6	<u>MazeFilter.cpp</u>
1.7	<u>Object.cpp</u>
1.8	<u>PlayerMovement.cpp</u>
1.9	<u>RoomController.cpp</u>
2	<u>Header Files</u>
2.1	<u>CreateObject.h</u>
2.2	<u>DrawWindows.h</u>
2.3	<u>GlobalVars.h</u>
2.4	<u>ImportMaps.h</u>
2.5	<u>Interactions.h</u>
2.6	<u>MazeFilter.h</u>
2.7	<u>Object.h</u>
2.8	<u>PlayerMovement.h</u>
2.9	<u>RoomController.h</u>

C++ files

MainGameLoop.cpp

```
1  /* This is the Main Game loop. The code is repeated while loopGame is true. The switch
2  Statement checks to see the state of the game (if it is in menu, in a fight, or in the overworld)
3  and calls the corresponding function that is called each frame. */
4
5  // Includes
6  #include <stdio.h>
7  #include <Windows.h>
8  #include <curses.h>
9  #include "DrawWindows.h"
10 #include "ImportMaps.h"
11 #include "GlobalVars.h"
12 #include "RoomController.h"
13 #include "Interactions.h"
14
15 // Prototypes
16 void ResizeWindow(int _x, int _y);
17
18 // Variables
19 int loopGame = 1, gameState = -1, save = 1;
20 int roomX = 1, roomY = 1, Step = 0, money = 0;
21 bool enter = false, inv = false;
22
23 char* global_item[4] = { "Bomb", "Lantern", "Hammer", "mk II Lantern" };
24
25 Level currentLevel;
26
27 using namespace std;
28
29 void ResizeWindow(int _x, int _y) //Take the new sizes parameters
30 {
31     resize_term(68,200); //The Lines and Columns needed
32     HWND console = GetConsoleWindow();
33     RECT r;
34     GetWindowRect(console, &r); //stores the console's current dimensions
35     MoveWindow(console, r.left, r.top, _x, _y, TRUE);
36     ShowWindow( console, SW_MINIMIZE); //Minimize first, incase it is already maximized
37     ShowWindow( console, SW_MAXIMIZE); //because maximizing twice messes it up
38 }
39
```

```
40 int main()
41 {
42     initscr(); //Start Public Domain Curses
43
44     if(has_colors() == FALSE) //Cekck if colors are supported
45     {
46         printf("Your terminal does not support color\n"); //Apologize
47         printf("Closing Game, sorry\n");
48         getch();
49         loopGame = 0; //Never start the loop or initialize colors, if they arent supported
50     }
51
52     loopGame = ImportAsciiArt("Assets\\SpriteSheet.txt");
53     noecho();
54     curs_set(0); //Invisible cursor
55     keypad(stdscr, TRUE); //Additional input
56     nonl();
57
58     ResizeWindow(1600,816); //This resizes the window first
59
60     while(loopGame) //Loop while true
61     {
62         start_color(); //Start color
63
64         init_pair(1, COLOR_BLACK, COLOR_WHITE); //WHITE
65         init_pair(2, COLOR_WHITE, COLOR_BLACK); //Hightlighted Text
66
67         switch (gameState)
68         {
69             case -1:
70                 loopGame = DrawBackground("Assets\\TowerBackground.txt", 1);
71                 break;
72
73             case 0: //Main Menu
74                 loopGame = DrawBackground("Assets\\TowerBackground.txt", 0);
75                 loopGame = MainMenu();
76                 break;
77
78             case 1: //Overworld
79                 loopGame = RoomUpdate();
80                 break;
81
82         }
83     }
84     endwin();
85     return 0; //End the game
```

```

86 }
87
88 bool EnterWasPressed()
89 {
90     if (enter == false && (GetAsyncKeyState(VK_RETURN) && 0x8000))
91     {
92         enter = true;
93         return true;
94     }
95     else if (enter == true && (!GetAsyncKeyState(VK_RETURN) && 0x8000))
96     {
97         enter = false;
98         return false;
99     }
100     return false;
101 }
102
103 bool InventoryWasPressed()
104 {
105     if (inv == false && (GetAsyncKeyState('I') && 0x8000))
106     {
107         inv = true;
108         return true;
109     }
110     else if (inv == true && (!GetAsyncKeyState('I') && 0x8000))
111     {
112         inv = false;
113         return false;
114     }
115     return false;
116 }

```

CreateObject.cpp

```

1 //Includes
2 #include "Object.h"
3 #include "DrawWindows.h"
4 #include "GlobalVars.h"
5 #include "Interactions.h"
6
7 //Prototypes
8 void SetAnimations(int a, int b, int c, int d);
9 void DefineObjects(int object, int y, int x);
10
11 //Variables
12 const int SPRITE_HEIGHT = 6;

```

```
13 const int SPRITE_WIDTH = 9;
14 int ani[4];
15
16 void SetAnimations( int a, int b, int c, int d)
17 {
18     ani[0] = a;
19     ani[1] = b;
20     ani[2] = c;
21     ani[3] = d;
22 }
23
24 void DefineObjects(int object, int y, int x)
25 {
26     int index = (y*22)+x;
27     SetAnimations(0,1,2,1);
28
29     currentLevel.puzzleObject[index] = DefineObject(200,-2,ani,create_win(SPRITE_HEIGHT,SPRITE_WIDTH,y*SPRITE_HEIGHT,x*SPRITE_WIDTH),0,0,0,0,0,0);
30     currentLevel.puzzleObject[index].DrawSprite(currentLevel.puzzleObject[index],0);
31
32     switch (object)
33     {
34     case -16:
35         SetAnimations(0,0,1,1);
36         currentLevel.roomObject[index] = DefineObject(83,1,ani,create_win(SPRITE_HEIGHT,SPRITE_WIDTH,y*SPRITE_HEIGHT,x*SPRITE_WIDTH),0,0,y,x,0,object);
37         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
38         break;
39     case -15: //Ice
40         currentLevel.roomObject[index] = DefineObject(105,-1,ani,create_win(SPRITE_HEIGHT,SPRITE_WIDTH,y*SPRITE_HEIGHT,x*SPRITE_WIDTH),0,0,y,x,0,object);
41         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
42         break;
43     case -14: //Cobblestone 4
44         currentLevel.roomObject[index] = DefineObject(180,-1,ani,create_win(SPRITE_HEIGHT,SPRITE_WIDTH,y*SPRITE_HEIGHT,x*SPRITE_WIDTH),0,0,y,x,0,object);
45         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
46         break;
47     case -13: //Cobblestone 3
48         currentLevel.roomObject[index] = DefineObject(179,-1,ani,create_win(SPRITE_HEIGHT,SPRITE_WIDTH,y*SPRITE_HEIGHT,x*SPRITE_WIDTH),0,0,y,x,0,object);
49         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
50         break;
51     case -12: //Cobblestone 2
52         currentLevel.roomObject[index] = DefineObject(178,-1,ani,create_win(SPRITE_HEIGHT,SPRITE_WIDTH,y*SPRITE_HEIGHT,x*SPRITE_WIDTH),0,0,y,x,0,object);
53         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
54         break;
55     case -11: //Cobblestone 1
56         currentLevel.roomObject[index] = DefineObject(177,-1,ani,create_win(SPRITE_HEIGHT,SPRITE_WIDTH,y*SPRITE_HEIGHT,x*SPRITE_WIDTH),0,0,y,x,0,object);
57         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
58         break;
```

```

59     case -10: //Bridge Left
60         currentLevel.roomObject[index] = DefineObject(86,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
61         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
62         break;
63     case -9: //Bridge Right
64         currentLevel.roomObject[index] = DefineObject(87,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
65         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
66         break;
67     case -8: //Bridge Middle
68         currentLevel.roomObject[index] = DefineObject(88,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
69         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
70         break;
71     case -7: //Bridge Horizontal Top
72         currentLevel.roomObject[index] = DefineObject(89,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
73         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
74         break;
75     case -6: //Bridge Horizontal Middle
76         currentLevel.roomObject[index] = DefineObject(205,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
77         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
78         break;
79     case -5: //Bridge Horizontal Bottom
80         currentLevel.roomObject[index] = DefineObject(90,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
81         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
82         break;
83     case -4: //Ladder Right
84         currentLevel.roomObject[index] = DefineObject(197,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
85         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
86         break;
87     case -3: //Ladder Left
88         currentLevel.roomObject[index] = DefineObject(198,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
89         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
90         break;
91     case -2: //Ladder Vertical
92         currentLevel.roomObject[index] = DefineObject(153,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
93         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
94         break;
95     case -1: //Blank Space
96         currentLevel.roomObject[index] = DefineObject(200,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
97         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
98         break;
99     case 0: //Blank Space
100         currentLevel.roomObject[index] = DefineObject(200,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
101         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
102         break;
103     case 1: //Chest Unlocked
104         currentLevel.roomObject[index] = DefineObject(25,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);

```

```

105         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
106         break;
107     case 2: //Chest Locked
108         currentLevel.roomObject[index] = DefineObject(26,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
109         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
110         break;
111     case 3: //Chest Opened
112         currentLevel.roomObject[index] = DefineObject(27,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
113         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
114         break;
115     case 4: //Horizontal Table Left
116         currentLevel.roomObject[index] = DefineObject(28,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
117         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
118         break;
119     case 5: //Horizontal Table Middle
120         currentLevel.roomObject[index] = DefineObject(29,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
121         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
122         break;
123     case 6: //Horizontal Table Right
124         currentLevel.roomObject[index] = DefineObject(30,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
125         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
126         break;
127     case 7: //Vertical Table Top
128         currentLevel.roomObject[index] = DefineObject(31,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
129         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
130         break;
131     case 8: //Vertical Table Middle
132         currentLevel.roomObject[index] = DefineObject(32,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
133         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
134         break;
135     case 9: //Vertical Table Bottom
136         currentLevel.roomObject[index] = DefineObject(33,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
137         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
138         break;
139     case 10: //Empty Chair
140         currentLevel.roomObject[index] = DefineObject(34,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
141         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
142         break;
143     case 11: //Woman Seated Facing Left
144         currentLevel.roomObject[index] = DefineObject(35,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
145         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
146         break;
147     case 12: //Woman Seated Facing Right
148         currentLevel.roomObject[index] = DefineObject(36,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
149         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
150         break;

```

```
151     case 13: //Man Seated Facing Up
152         currentLevel.roomObject[index] = DefineObject(37, -1, ani, create_win(Sprite_Height, Sprite_Width, y*Sprite_Height, x*Sprite_Width), 0, 0, y, x, 0, object);
153         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index], 0);
154         break;
155     case 14: //Man Seated Facing Left
156         currentLevel.roomObject[index] = DefineObject(38, -1, ani, create_win(Sprite_Height, Sprite_Width, y*Sprite_Height, x*Sprite_Width), 0, 0, y, x, 0, object);
157         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index], 0);
158         break;
159     case 15: //Man Seated Facing Right
160         currentLevel.roomObject[index] = DefineObject(39, -1, ani, create_win(Sprite_Height, Sprite_Width, y*Sprite_Height, x*Sprite_Width), 0, 0, y, x, 0, object);
161         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index], 0);
162         break;
163     case 16: //Vertical Table on a Brick Wall
164         currentLevel.roomObject[index] = DefineObject(40, -1, ani, create_win(Sprite_Height, Sprite_Width, y*Sprite_Height, x*Sprite_Width), 0, 0, y, x, 0, object);
165         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index], 0);
166         break;
167     case 17: //Brick Wall
168         currentLevel.roomObject[index] = DefineObject(41, -1, ani, create_win(Sprite_Height, Sprite_Width, y*Sprite_Height, x*Sprite_Width), 0, 0, y, x, 0, object);
169         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index], 0);
170         break;
171     case 18: //Window on a Brick Wall
172         currentLevel.roomObject[index] = DefineObject(42, -1, ani, create_win(Sprite_Height, Sprite_Width, y*Sprite_Height, x*Sprite_Width), 0, 0, y, x, 0, object);
173         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index], 0);
174         break;
175     case 19: //Door Closed
176         currentLevel.roomObject[index] = DefineObject(43, -1, ani, create_win(Sprite_Height, Sprite_Width, y*Sprite_Height, x*Sprite_Width), 0, 0, y, x, 0, object);
177         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index], 0);
178         break;
179     case 20: //Door Open
180         currentLevel.roomObject[index] = DefineObject(44, -1, ani, create_win(Sprite_Height, Sprite_Width, y*Sprite_Height, x*Sprite_Width), 0, 0, y, x, 0, object);
181         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index], 0);
182         break;
183     case 21: //Cave Door
184         currentLevel.roomObject[index] = DefineObject(45, -1, ani, create_win(Sprite_Height, Sprite_Width, y*Sprite_Height, x*Sprite_Width), 0, 0, y, x, 0, object);
185         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index], 0);
186         break;
187     case 22: //Fence Going Left
188         currentLevel.roomObject[index] = DefineObject(46, -1, ani, create_win(Sprite_Height, Sprite_Width, y*Sprite_Height, x*Sprite_Width), 0, 0, y, x, 0, object);
189         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index], 0);
190         break;
191     case 23: //Fence Going Up
192         currentLevel.roomObject[index] = DefineObject(47, -1, ani, create_win(Sprite_Height, Sprite_Width, y*Sprite_Height, x*Sprite_Width), 0, 0, y, x, 0, object);
193         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index], 0);
194         break;
195     case 24: //Fence Going Right
196         currentLevel.roomObject[index] = DefineObject(48, -1, ani, create_win(Sprite_Height, Sprite_Width, y*Sprite_Height, x*Sprite_Width), 0, 0, y, x, 0, object);
```



```

197         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
198         break;
199     case 25: //Top of a Shaded Wall
200         currentLevel.roomObject[index] = DefineObject(50,-1,ani,create_win(SPRITE_HEIGHT,SPRITE_WIDTH,y*SPRITE_HEIGHT,x*SPRITE_WIDTH),0,0,y,x,0,object);
201         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
202         break;
203     case 26: //Brick Wall Right
204         currentLevel.roomObject[index] = DefineObject(65,-1,ani,create_win(SPRITE_HEIGHT,SPRITE_WIDTH,y*SPRITE_HEIGHT,x*SPRITE_WIDTH),0,0,y,x,0,object);
205         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
206         break;
207     case 27: //Brick Wall Left
208         currentLevel.roomObject[index] = DefineObject(68,-1,ani,create_win(SPRITE_HEIGHT,SPRITE_WIDTH,y*SPRITE_HEIGHT,x*SPRITE_WIDTH),0,0,y,x,0,object);
209         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
210         break;
211     case 28: //Brick Wall Bottom
212         currentLevel.roomObject[index] = DefineObject(73,-1,ani,create_win(SPRITE_HEIGHT,SPRITE_WIDTH,y*SPRITE_HEIGHT,x*SPRITE_WIDTH),0,0,y,x,0,object);
213         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
214         break;
215     case 29: //Brick top left inside corner
216         currentLevel.roomObject[index] = DefineObject(67,-1,ani,create_win(SPRITE_HEIGHT,SPRITE_WIDTH,y*SPRITE_HEIGHT,x*SPRITE_WIDTH),0,0,y,x,0,object);
217         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
218         break;
219     case 30: //Brick top right inside corner
220         currentLevel.roomObject[index] = DefineObject(64,-1,ani,create_win(SPRITE_HEIGHT,SPRITE_WIDTH,y*SPRITE_HEIGHT,x*SPRITE_WIDTH),0,0,y,x,0,object);
221         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
222         break;
223     case 31: //Brick bottom left inside corner
224         currentLevel.roomObject[index] = DefineObject(69,-1,ani,create_win(SPRITE_HEIGHT,SPRITE_WIDTH,y*SPRITE_HEIGHT,x*SPRITE_WIDTH),0,0,y,x,0,object);
225         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
226         break;
227     case 32: //Brick bottom right inside corner
228         currentLevel.roomObject[index] = DefineObject(66,-1,ani,create_win(SPRITE_HEIGHT,SPRITE_WIDTH,y*SPRITE_HEIGHT,x*SPRITE_WIDTH),0,0,y,x,0,object);
229         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
230         break;
231     case 33: //Brick top left outside corner
232         currentLevel.roomObject[index] = DefineObject(74,-1,ani,create_win(SPRITE_HEIGHT,SPRITE_WIDTH,y*SPRITE_HEIGHT,x*SPRITE_WIDTH),0,0,y,x,0,object);
233         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
234         break;
235     case 34: //Brick top right outside corner
236         currentLevel.roomObject[index] = DefineObject(75,-1,ani,create_win(SPRITE_HEIGHT,SPRITE_WIDTH,y*SPRITE_HEIGHT,x*SPRITE_WIDTH),0,0,y,x,0,object);
237         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
238         break;
239     case 35: //Brick bottom left outside corner
240         currentLevel.roomObject[index] = DefineObject(76,-1,ani,create_win(SPRITE_HEIGHT,SPRITE_WIDTH,y*SPRITE_HEIGHT,x*SPRITE_WIDTH),0,0,y,x,0,object);
241         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
242         break;

```

```

243     case 36: //Brick bottom right outside corner
244         currentLevel.roomObject[index] = DefineObject(77,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
245         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
246         break;
247     case 37: //Brick Wall Door Facing Right
248         currentLevel.roomObject[index] = DefineObject(70,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
249         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
250         break;
251     case 38: //Brick Wall Door Facing Left
252         currentLevel.roomObject[index] = DefineObject(71,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
253         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
254         break;
255     case 39: //Brick Wall Door Facing Up
256         currentLevel.roomObject[index] = DefineObject(72,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
257         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
258         break;
259     case 40: //Sign
260         currentLevel.roomObject[index] = DefineObject(78,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
261         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
262         break;
263     case 41: //Bed Top
264         currentLevel.roomObject[index] = DefineObject(79,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
265         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
266         break;
267     case 42: //Bed Bottom
268         currentLevel.roomObject[index] = DefineObject(80,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
269         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
270         break;
271     case 43: //Tree
272         currentLevel.roomObject[index] = DefineObject(91,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
273         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
274         break;
275     case 44: //Bush
276         currentLevel.roomObject[index] = DefineObject(92,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
277         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
278         break;
279     case 45: //Pushing Block
280         currentLevel.roomObject[index] = DefineObject(93,-2,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,0);
281         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
282         currentLevel.puzzleObject[index] = DefineObject(93,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
283         currentLevel.puzzleObject[index].DrawSprite(currentLevel.puzzleObject[index],0);
284         break;
285     case 46: //Church middle top
286         currentLevel.roomObject[index] = DefineObject(109,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
287         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
288         break;

```

```

289     case 47: //Church middle empty
290         currentLevel.roomObject[index] = DefineObject(110,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
291         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
292         break;
293     case 48: //Church middle cross
294         currentLevel.roomObject[index] = DefineObject(111,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
295         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
296         break;
297     case 49: //Church middle door
298         currentLevel.roomObject[index] = DefineObject(112,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
299         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
300         break;
301     case 50: //Church left top
302         currentLevel.roomObject[index] = DefineObject(113,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
303         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
304         break;
305     case 51: //Church left middle
306         currentLevel.roomObject[index] = DefineObject(114,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
307         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
308         break;
309     case 52: //Church right top
310         currentLevel.roomObject[index] = DefineObject(115,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
311         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
312         break;
313     case 53: //Church right middle
314         currentLevel.roomObject[index] = DefineObject(116,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
315         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
316         break;
317     case 54: //Church side bottom
318         currentLevel.roomObject[index] = DefineObject(117,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
319         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
320         break;
321     case 55: //Water Pattern 1
322         SetAnimations(0,0,1,1);
323         currentLevel.roomObject[index] = DefineObject(81,1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
324         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
325         break;
326     /*case 56: //Water Corner Top Left
327         currentLevel.roomObject[index] = DefineObject(82,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
328         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
329         break;
330     case 57: //Water Corner Top Right
331         currentLevel.roomObject[index] = DefineObject(83,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
332         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
333         break;
334     case 58: //Water Corner Bottom Left

```

```
335         currentLevel.roomObject[index] = DefineObject(84,-1,ani,create_win(SPRITE_HEIGHT,SPRITE_WIDTH,y*SPRITE_HEIGHT,x*SPRITE_WIDTH),0,0,y,x,0,object);
336         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
337         break;
338     case 59: //Water Corner Bottom Right
339         currentLevel.roomObject[index] = DefineObject(85,-1,ani,create_win(SPRITE_HEIGHT,SPRITE_WIDTH,y*SPRITE_HEIGHT,x*SPRITE_WIDTH),0,0,y,x,0,object);
340         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
341         break;
342     case 60: //VOID - CHANGE TO HOLE
343         currentLevel.roomObject[index] = DefineObject(90,-1,ani,create_win(SPRITE_HEIGHT,SPRITE_WIDTH,y*SPRITE_HEIGHT,x*SPRITE_WIDTH),0,0,y,x,0,object);
344         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
345         break;*/
346     case 61: //Brick Wall Full
347         currentLevel.roomObject[index] = DefineObject(120,-1,ani,create_win(SPRITE_HEIGHT,SPRITE_WIDTH,y*SPRITE_HEIGHT,x*SPRITE_WIDTH),0,0,y,x,0,object);
348         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
349         break;
350     case 62: //Brick Window Full
351         currentLevel.roomObject[index] = DefineObject(121,-1,ani,create_win(SPRITE_HEIGHT,SPRITE_WIDTH,y*SPRITE_HEIGHT,x*SPRITE_WIDTH),0,0,y,x,0,object);
352         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
353         break;
354     case 63: //Brick Door Full
355         currentLevel.roomObject[index] = DefineObject(122,-1,ani,create_win(SPRITE_HEIGHT,SPRITE_WIDTH,y*SPRITE_HEIGHT,x*SPRITE_WIDTH),0,0,y,x,0,object);
356         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
357         break;
358     case 64: //Mod Wall?
359         currentLevel.roomObject[index] = DefineObject(123,-1,ani,create_win(SPRITE_HEIGHT,SPRITE_WIDTH,y*SPRITE_HEIGHT,x*SPRITE_WIDTH),0,0,y,x,0,object);
360         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
361         break;
362     case 65: //Bookshelf
363         currentLevel.roomObject[index] = DefineObject(147,-1,ani,create_win(SPRITE_HEIGHT,SPRITE_WIDTH,y*SPRITE_HEIGHT,x*SPRITE_WIDTH),0,0,y,x,0,object);
364         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
365         break;
366     case 66: //Cliff face
367         currentLevel.roomObject[index] = DefineObject(148,-1,ani,create_win(SPRITE_HEIGHT,SPRITE_WIDTH,y*SPRITE_HEIGHT,x*SPRITE_WIDTH),0,0,y,x,0,object);
368         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
369         break;
370     case 67: //Cliff Left Angle Bottom
371         currentLevel.roomObject[index] = DefineObject(149,-1,ani,create_win(SPRITE_HEIGHT,SPRITE_WIDTH,y*SPRITE_HEIGHT,x*SPRITE_WIDTH),0,0,y,x,0,object);
372         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
373         break;
374     case 68: //Cliff Left Angle Top
375         currentLevel.roomObject[index] = DefineObject(150,-1,ani,create_win(SPRITE_HEIGHT,SPRITE_WIDTH,y*SPRITE_HEIGHT,x*SPRITE_WIDTH),0,0,y,x,0,object);
376         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
377         break;
378     case 69: //Cliff Right Angle Bottom
379         currentLevel.roomObject[index] = DefineObject(151,-1,ani,create_win(SPRITE_HEIGHT,SPRITE_WIDTH,y*SPRITE_HEIGHT,x*SPRITE_WIDTH),0,0,y,x,0,object);
380         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
```

```
381         break;
382     case 70: //Cliff Right Angle Top
383         currentLevel.roomObject[index] = DefineObject(152,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
384         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
385         break;
386     case 71: //Cliff Right Side
387         currentLevel.roomObject[index] = DefineObject(195,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
388         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
389         break;
390     case 72: //Barn 1
391         currentLevel.roomObject[index] = DefineObject(154,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
392         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
393         break;
394     case 73: //Barn 2
395         currentLevel.roomObject[index] = DefineObject(155,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
396         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
397         break;
398     case 74: //Barn 3
399         currentLevel.roomObject[index] = DefineObject(156,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
400         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
401         break;
402     case 75: //Barn 4
403         currentLevel.roomObject[index] = DefineObject(157,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
404         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
405         break;
406     case 76: //Barn 5
407         currentLevel.roomObject[index] = DefineObject(158,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
408         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
409         break;
410     case 77: //Barn 6
411         currentLevel.roomObject[index] = DefineObject(159,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
412         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
413         break;
414     case 78: //Barn 7
415         currentLevel.roomObject[index] = DefineObject(160,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
416         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
417         break;
418     case 79: //Barn 8
419         currentLevel.roomObject[index] = DefineObject(161,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
420         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
421         break;
422     case 80: //Barn 9
423         currentLevel.roomObject[index] = DefineObject(162,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
424         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
425         break;
426     case 81: //Barn 10
```

```
427         currentLevel.roomObject[index] = DefineObject(163,-1,ani,create_win(SPRITE_HEIGHT,SPRITE_WIDTH,y*SPRITE_HEIGHT,x*SPRITE_WIDTH),0,0,y,x,0,object);
428         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
429         break;
430     case 82: //Barn 11
431         currentLevel.roomObject[index] = DefineObject(164,-1,ani,create_win(SPRITE_HEIGHT,SPRITE_WIDTH,y*SPRITE_HEIGHT,x*SPRITE_WIDTH),0,0,y,x,0,object);
432         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
433         break;
434     case 83: //Barn 12
435         currentLevel.roomObject[index] = DefineObject(165,-1,ani,create_win(SPRITE_HEIGHT,SPRITE_WIDTH,y*SPRITE_HEIGHT,x*SPRITE_WIDTH),0,0,y,x,0,object);
436         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
437         break;
438     case 84: //Barn 13
439         currentLevel.roomObject[index] = DefineObject(166,-1,ani,create_win(SPRITE_HEIGHT,SPRITE_WIDTH,y*SPRITE_HEIGHT,x*SPRITE_WIDTH),0,0,y,x,0,object);
440         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
441         break;
442     case 85: //Barn 14
443         currentLevel.roomObject[index] = DefineObject(167,-1,ani,create_win(SPRITE_HEIGHT,SPRITE_WIDTH,y*SPRITE_HEIGHT,x*SPRITE_WIDTH),0,0,y,x,0,object);
444         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
445         break;
446     case 86: //Barn 15
447         currentLevel.roomObject[index] = DefineObject(168,-1,ani,create_win(SPRITE_HEIGHT,SPRITE_WIDTH,y*SPRITE_HEIGHT,x*SPRITE_WIDTH),0,0,y,x,0,object);
448         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
449         break;
450     case 87: //Horse 1
451         currentLevel.roomObject[index] = DefineObject(170,-1,ani,create_win(SPRITE_HEIGHT,SPRITE_WIDTH,y*SPRITE_HEIGHT,x*SPRITE_WIDTH),0,0,y,x,0,object);
452         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
453         break;
454     case 88: //Horse 2
455         currentLevel.roomObject[index] = DefineObject(171,-1,ani,create_win(SPRITE_HEIGHT,SPRITE_WIDTH,y*SPRITE_HEIGHT,x*SPRITE_WIDTH),0,0,y,x,0,object);
456         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
457         break;
458     case 89: //Wood Wall top
459         currentLevel.roomObject[index] = DefineObject(172,-1,ani,create_win(SPRITE_HEIGHT,SPRITE_WIDTH,y*SPRITE_HEIGHT,x*SPRITE_WIDTH),0,0,y,x,0,object);
460         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
461         break;
462     case 90: //Wood Wall left
463         currentLevel.roomObject[index] = DefineObject(173,-1,ani,create_win(SPRITE_HEIGHT,SPRITE_WIDTH,y*SPRITE_HEIGHT,x*SPRITE_WIDTH),0,0,y,x,0,object);
464         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
465         break;
466     case 91: //Wood Wall right
467         currentLevel.roomObject[index] = DefineObject(174,-1,ani,create_win(SPRITE_HEIGHT,SPRITE_WIDTH,y*SPRITE_HEIGHT,x*SPRITE_WIDTH),0,0,y,x,0,object);
468         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
469         break;
470     case 92: //Wood Wall bottom
471         currentLevel.roomObject[index] = DefineObject(175,-1,ani,create_win(SPRITE_HEIGHT,SPRITE_WIDTH,y*SPRITE_HEIGHT,x*SPRITE_WIDTH),0,0,y,x,0,object);
472         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
```

```
473         break;
474     case 93: //Wood corner 1
475         currentLevel.roomObject[index] = DefineObject(176,-1,ani,create_win(SPRITE_HEIGHT,SPRITE_WIDTH,y*SPRITE_HEIGHT,x*SPRITE_WIDTH),0,0,y,x,0,object);
476         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
477         break;
478     case 94: //Wood corner 2
479         currentLevel.roomObject[index] = DefineObject(177,-1,ani,create_win(SPRITE_HEIGHT,SPRITE_WIDTH,y*SPRITE_HEIGHT,x*SPRITE_WIDTH),0,0,y,x,0,object);
480         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
481         break;
482     case 95: //Wood corner 3
483         currentLevel.roomObject[index] = DefineObject(178,-1,ani,create_win(SPRITE_HEIGHT,SPRITE_WIDTH,y*SPRITE_HEIGHT,x*SPRITE_WIDTH),0,0,y,x,0,object);
484         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
485         break;
486     case 96: //Wood corner 4
487         currentLevel.roomObject[index] = DefineObject(179,-1,ani,create_win(SPRITE_HEIGHT,SPRITE_WIDTH,y*SPRITE_HEIGHT,x*SPRITE_WIDTH),0,0,y,x,0,object);
488         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
489         break;
490     case 97: //Dog
491         currentLevel.roomObject[index] = DefineObject(181,-1,ani,create_win(SPRITE_HEIGHT,SPRITE_WIDTH,y*SPRITE_HEIGHT,x*SPRITE_WIDTH),0,0,y,x,0,object);
492         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
493         break;
494     case 101: //Smashable Block
495         currentLevel.roomObject[index] = DefineObject(182,-1,ani,create_win(SPRITE_HEIGHT,SPRITE_WIDTH,y*SPRITE_HEIGHT,x*SPRITE_WIDTH),0,0,y,x,0,object);
496         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
497         break;
498     case 102: //Decorative Boulder
499         currentLevel.roomObject[index] = DefineObject(183,-1,ani,create_win(SPRITE_HEIGHT,SPRITE_WIDTH,y*SPRITE_HEIGHT,x*SPRITE_WIDTH),0,0,y,x,0,object);
500         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
501         break;
502     case 103: //Well
503         currentLevel.roomObject[index] = DefineObject(101,-1,ani,create_win(SPRITE_HEIGHT,SPRITE_WIDTH,y*SPRITE_HEIGHT,x*SPRITE_WIDTH),0,0,y,x,0,object);
504         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
505         break;
506     case 104: //Campfire
507         currentLevel.roomObject[index] = DefineObject(184,-1,ani,create_win(SPRITE_HEIGHT,SPRITE_WIDTH,y*SPRITE_HEIGHT,x*SPRITE_WIDTH),0,0,y,x,0,object);
508         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
509         break;
510     case 105: //Cliff Left Side
511         currentLevel.roomObject[index] = DefineObject(185,-1,ani,create_win(SPRITE_HEIGHT,SPRITE_WIDTH,y*SPRITE_HEIGHT,x*SPRITE_WIDTH),0,0,y,x,0,object);
512         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
513         break;
514     case 106: //Cliff Right Side
515         currentLevel.roomObject[index] = DefineObject(186,-1,ani,create_win(SPRITE_HEIGHT,SPRITE_WIDTH,y*SPRITE_HEIGHT,x*SPRITE_WIDTH),0,0,y,x,0,object);
516         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
517         break;
518     case 107: //Cliff Back Left Corner
```

```

519         currentLevel.roomObject[index] = DefineObject(187,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
520         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
521         break;
522     case 108: //Cliff Back Right Corner
523         currentLevel.roomObject[index] = DefineObject(188,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
524         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
525         break;
526     case 109: //Cliff Back Border
527         currentLevel.roomObject[index] = DefineObject(189,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
528         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
529         break;
530     case 110: //Cliff Side Height
531         currentLevel.roomObject[index] = DefineObject(190,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
532         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
533         break;
534     case 111: //Ladder Up
535         currentLevel.roomObject[index] = DefineObject(204,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
536         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
537         break;
538     case 112: //Bridge Right
539         currentLevel.roomObject[index] = DefineObject(180,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
540         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
541         break;
542     /*case 113:          //Bridge Middle
543         currentLevel.roomObject[index] = DefineObject(90,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
544         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
545         break;*/
546     case 114: //Indoor Cliff Top Left
547         currentLevel.roomObject[index] = DefineObject(191,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
548         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
549         break;
550     case 115: //Indoor Cliff Left
551         currentLevel.roomObject[index] = DefineObject(192,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
552         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
553         break;
554     case 116: //Indoor Cliff Bottom Left
555         currentLevel.roomObject[index] = DefineObject(193,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
556         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
557         break;
558     case 117: //Indoor Cliff Top Right
559         currentLevel.roomObject[index] = DefineObject(194,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
560         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
561         break;
562     case 118: //Indoor Cliff Right
563         currentLevel.roomObject[index] = DefineObject(195,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
564         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);

```



```

565         break;
566     case 119: //Indoor Cliff Bottom Right
567         currentLevel.roomObject[index] = DefineObject(196,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
568         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
569         break;
570     case 120: //Indoor Cliff Ladder Right
571         currentLevel.roomObject[index] = DefineObject(197,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
572         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
573         break;
574     case 121: //Indoor Cliff Ladder Left
575         currentLevel.roomObject[index] = DefineObject(198,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
576         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
577         break;
578     /*case 122:          //Roof Chimney
579         currentLevel.roomObject[index] = DefineObject(199,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
580         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
581         break;*/
582     case 123: //Statue 1
583         currentLevel.roomObject[index] = DefineObject(210,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
584         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
585         break;
586     case 124: //Statue 2
587         currentLevel.roomObject[index] = DefineObject(211,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
588         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
589         break;
590     case 125: //Statue 3
591         currentLevel.roomObject[index] = DefineObject(212,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
592         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
593         break;
594     case 126: //Statue 4
595         currentLevel.roomObject[index] = DefineObject(213,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
596         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
597         break;
598     case 127: //Statue 5
599         currentLevel.roomObject[index] = DefineObject(214,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
600         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
601         break;
602     case 128: //Statue 6
603         currentLevel.roomObject[index] = DefineObject(215,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
604         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
605         break;
606     case 129: //Anvil
607         SetAnimations(0,0,1,1);
608         currentLevel.roomObject[index] = DefineObject(230,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
609         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
610         break;

```

```
611     case 130: //House
612         currentLevel.roomObject[index] = DefineObject(135,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
613         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
614         break;
615     case 131: //House
616         currentLevel.roomObject[index] = DefineObject(136,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
617         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
618         break;
619     case 132: //House
620         currentLevel.roomObject[index] = DefineObject(137,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
621         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
622         break;
623     case 133: //House
624         currentLevel.roomObject[index] = DefineObject(138,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
625         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
626         break;
627     case 134: //House
628         currentLevel.roomObject[index] = DefineObject(139,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
629         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
630         break;
631     case 135: //House
632         currentLevel.roomObject[index] = DefineObject(140,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
633         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
634         break;
635     case 136: //House
636         currentLevel.roomObject[index] = DefineObject(141,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
637         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
638         break;
639     case 137: //House
640         currentLevel.roomObject[index] = DefineObject(142,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
641         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
642         break;
643     case 138: //House
644         currentLevel.roomObject[index] = DefineObject(143,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
645         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
646         break;
647     case 139: //House
648         currentLevel.roomObject[index] = DefineObject(144,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
649         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
650         break;
651     case 140: //House
652         currentLevel.roomObject[index] = DefineObject(145,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
653         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
654         break;
655     case 141: //House
656         currentLevel.roomObject[index] = DefineObject(146,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
```

```
657         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
658         break;
659     case 142: //Alchemist Logo
660         currentLevel.roomObject[index] = DefineObject(131,-1,ani,create_win(SPRITE_HEIGHT,SPRITE_WIDTH,y*SPRITE_HEIGHT,x*SPRITE_WIDTH),0,0,y,x,0,object);
661         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
662         break;
663     case 143: //Survivalist Logo
664         currentLevel.roomObject[index] = DefineObject(134,-1,ani,create_win(SPRITE_HEIGHT,SPRITE_WIDTH,y*SPRITE_HEIGHT,x*SPRITE_WIDTH),0,0,y,x,0,object);
665         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
666         break;
667     case 144: //Locksmith Logo
668         currentLevel.roomObject[index] = DefineObject(133,-1,ani,create_win(SPRITE_HEIGHT,SPRITE_WIDTH,y*SPRITE_HEIGHT,x*SPRITE_WIDTH),0,0,y,x,0,object);
669         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
670         break;
671     case 145: //Bomb Crafter Logo
672         currentLevel.roomObject[index] = DefineObject(132,-1,ani,create_win(SPRITE_HEIGHT,SPRITE_WIDTH,y*SPRITE_HEIGHT,x*SPRITE_WIDTH),0,0,y,x,0,object);
673         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
674         break;
675     case 146: //Odd Corner Brick
676         currentLevel.roomObject[index] = DefineObject(5,-1,ani,create_win(SPRITE_HEIGHT,SPRITE_WIDTH,y*SPRITE_HEIGHT,x*SPRITE_WIDTH),0,0,y,x,0,object);
677         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
678         break;
679     case 147: //Odd Corner Brick
680         currentLevel.roomObject[index] = DefineObject(6,-1,ani,create_win(SPRITE_HEIGHT,SPRITE_WIDTH,y*SPRITE_HEIGHT,x*SPRITE_WIDTH),0,0,y,x,0,object);
681         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
682         break;
683     case 148: //Odd Corner Brick
684         currentLevel.roomObject[index] = DefineObject(7,-1,ani,create_win(SPRITE_HEIGHT,SPRITE_WIDTH,y*SPRITE_HEIGHT,x*SPRITE_WIDTH),0,0,y,x,0,object);
685         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
686         break;
687     case 149: //Odd Corner Brick
688         currentLevel.roomObject[index] = DefineObject(8,-1,ani,create_win(SPRITE_HEIGHT,SPRITE_WIDTH,y*SPRITE_HEIGHT,x*SPRITE_WIDTH),0,0,y,x,0,object);
689         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
690         break;
691     case 150: //Odd Corner Brick
692         currentLevel.roomObject[index] = DefineObject(9,-1,ani,create_win(SPRITE_HEIGHT,SPRITE_WIDTH,y*SPRITE_HEIGHT,x*SPRITE_WIDTH),0,0,y,x,0,object);
693         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
694         break;
695     case 151: //Odd Corner Brick
696         currentLevel.roomObject[index] = DefineObject(10,-1,ani,create_win(SPRITE_HEIGHT,SPRITE_WIDTH,y*SPRITE_HEIGHT,x*SPRITE_WIDTH),0,0,y,x,0,object);
697         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
698         break;
699     case 152: //Odd Corner Brick
700         currentLevel.roomObject[index] = DefineObject(11,-1,ani,create_win(SPRITE_HEIGHT,SPRITE_WIDTH,y*SPRITE_HEIGHT,x*SPRITE_WIDTH),0,0,y,x,0,object);
701         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
702         break;
```

```

703     case 153: //Odd Corner Brick
704         currentLevel.roomObject[index] = DefineObject(12,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
705         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
706         break;
707     case 154: //Table Corner 1
708         currentLevel.roomObject[index] = DefineObject(218,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
709         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
710         break;
711     case 155: //Table Corner 2
712         currentLevel.roomObject[index] = DefineObject(219,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
713         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
714         break;
715     case 156: //TEST SPRITE 1
716         SetAnimations(0,1,2,3);
717         currentLevel.roomObject[index] = DefineObject(224,1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
718         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
719         break;
720     case 157: //TEST SPRITE 2
721         currentLevel.roomObject[index] = DefineObject(225,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
722         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
723         break;
724     case 158: //TEST SPRITE 3
725         currentLevel.roomObject[index] = DefineObject(226,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
726         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
727         break;
728     case 159: //TEST SPRITE 4
729         currentLevel.roomObject[index] = DefineObject(227,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
730         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
731         break;
732 //Chairs
733
734     case 98:         //Seat facing left
735         currentLevel.roomObject[index] = DefineObject(34,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
736         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
737         break;
738     case 99:         //Seat facing right
739         currentLevel.roomObject[index] = DefineObject(34,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
740         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
741         break;
742     case 100: //Seat facing up
743         currentLevel.roomObject[index] = DefineObject(34,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
744         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
745         break;
746
747     //Interactable Doors (301-400)
748

```

```
749 case 301: //D1
750     currentLevel.roomObject[index] = DefineObject(71,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
751     currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
752     break;
753 case 302: //D2
754     currentLevel.roomObject[index] = DefineObject(70,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
755     currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
756     break;
757 case 303: //D3
758     currentLevel.roomObject[index] = DefineObject(70,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
759     currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
760     break;
761 case 304: //D4
762     currentLevel.roomObject[index] = DefineObject(71,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
763     currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
764     break;
765 case 305: //D5
766     currentLevel.roomObject[index] = DefineObject(72,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
767     currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
768     break;
769 case 306: //D6
770     currentLevel.roomObject[index] = DefineObject(140,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
771     currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
772     break;
773 case 307: //D7
774     currentLevel.roomObject[index] = DefineObject(43,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
775     currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
776     break;
777 case 308: //D8
778     currentLevel.roomObject[index] = DefineObject(72,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
779     currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
780     break;
781 case 309: //D9
782     currentLevel.roomObject[index] = DefineObject(45,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
783     currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
784     break;
785 case 310: //D10
786     currentLevel.roomObject[index] = DefineObject(202,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
787     currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
788     break;
789 case 311: //Alchemist IN
790     currentLevel.roomObject[index] = DefineObject(140,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
791     currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
792     break;
793 case 312: //Alchemist OUT
794     currentLevel.roomObject[index] = DefineObject(72,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
```

```

795         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
796         break;
797     case 313: //Survivalist IN
798         currentLevel.roomObject[index] = DefineObject(140,-1,ani,create_win(SPRT_HEIGHT,SPRT_WIDTH,y*SPRT_HEIGHT,x*SPRT_WIDTH),0,0,y,x,0,object);
799         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
800         break;
801     case 314: //Survivalist OUT
802         currentLevel.roomObject[index] = DefineObject(72,-1,ani,create_win(SPRT_HEIGHT,SPRT_WIDTH,y*SPRT_HEIGHT,x*SPRT_WIDTH),0,0,y,x,0,object);
803         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
804         break;
805     case 315: //Locksmith IN
806         currentLevel.roomObject[index] = DefineObject(140,-1,ani,create_win(SPRT_HEIGHT,SPRT_WIDTH,y*SPRT_HEIGHT,x*SPRT_WIDTH),0,0,y,x,0,object);
807         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
808         break;
809     case 316: //Locksmith OUT
810         currentLevel.roomObject[index] = DefineObject(72,-1,ani,create_win(SPRT_HEIGHT,SPRT_WIDTH,y*SPRT_HEIGHT,x*SPRT_WIDTH),0,0,y,x,0,object);
811         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
812         break;
813     case 317: //Bombshop IN
814         currentLevel.roomObject[index] = DefineObject(140,-1,ani,create_win(SPRT_HEIGHT,SPRT_WIDTH,y*SPRT_HEIGHT,x*SPRT_WIDTH),0,0,y,x,0,object);
815         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
816         break;
817     case 318: //Bombshop OUT
818         currentLevel.roomObject[index] = DefineObject(72,-1,ani,create_win(SPRT_HEIGHT,SPRT_WIDTH,y*SPRT_HEIGHT,x*SPRT_WIDTH),0,0,y,x,0,object);
819         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
820         break;
821
822 /*
823 //Chests (501-600)
824
825 case 501: //Chest 1: Locked chest in playerhousebedroom1
826     mapGrid[(x-2)/SPRT_WIDTH][(y-2)/SPRT_HEIGHT] = 501;
827     if (chestsOpened[0] == 1){
828         DrawSprite(8,2,x,y,false,false,0);
829     }
830     else DrawSprite(8,1,x,y,false,false,0);
831     break;
832 case 502: //Chest 2: Unlocked chest in playerhousebedroom3
833     mapGrid[(x-2)/SPRT_WIDTH][(y-2)/SPRT_HEIGHT] = 502;
834     if (chestsOpened[1] == 1){
835         DrawSprite(8,2,x,y,false,false,0);
836     }
837     else DrawSprite(8,0,x,y,false,false,0);
838     break;
839 case 503: //Chest 3: Unlocked chest in 0,-1
840     mapGrid[(x-2)/SPRT_WIDTH][(y-2)/SPRT_HEIGHT] = 503;

```

```

841         if (chestsOpened[2] == 1){
842             DrawSprite(8,2,x,y,false,false,0);
843         }
844         else DrawSprite(8,0,x,y,false,false,0);
845         break;
846     case 504: //Chest 4: Unlocked chest in 0,-1
847         mapGrid[(x-2)/SPRT_WIDTH][(y-2)/SPRT_HEIGHT] = 504;
848         if (chestsOpened[3] == 1){
849             DrawSprite(8,2,x,y,false,false,0);
850         }
851         else DrawSprite(8,0,x,y,false,false,0);
852         break;
853     case 505:
854         mapGrid[(x-2)/SPRT_WIDTH][(y-2)/SPRT_HEIGHT] = 505;
855         if (chestsOpened[4] == 1){
856             DrawSprite(8,2,x,y,false,false,0);
857         }
858         else DrawSprite(8,0,x,y,false,false,0);
859         break;
860     case 506:
861         mapGrid[(x-2)/SPRT_WIDTH][(y-2)/SPRT_HEIGHT] = 506;
862         if (chestsOpened[5] == 1){
863             DrawSprite(8,2,x,y,false,false,0);
864         }
865         else DrawSprite(8,0,x,y,false,false,0);
866         break;
867 */
868 //Signs (601-700)
869
870     case 601: //Sign 1: Your house
871         currentLevel.roomObject[index] = DefineObject(78,-1,ani,create_win(SPRT_HEIGHT,SPRT_WIDTH,y*SPRT_HEIGHT,x*SPRT_WIDTH),0,0,y,x,0,object);
872         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
873         break;
874     case 602: //Sign 2: Castletown ahead
875         currentLevel.roomObject[index] = DefineObject(78,-1,ani,create_win(SPRT_HEIGHT,SPRT_WIDTH,y*SPRT_HEIGHT,x*SPRT_WIDTH),0,0,y,x,0,object);
876         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
877         break;
878
879 //NPCs (701 - 1000)
880
881     case 701: //NPC 1: King/Duke
882         currentLevel.roomObject[index] = DefineObject(124,-1,ani,create_win(SPRT_HEIGHT,SPRT_WIDTH,y*SPRT_HEIGHT,x*SPRT_WIDTH),0,0,y,x,0,object);
883         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
884         break;
885     case 702: //NPC 2: Blacksmith
886         SetAnimations(0,0,1,1);

```

```
887         currentLevel.roomObject[index] = DefineObject(228,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
888         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
889         break;
890     case 703: //NPC 3: Conspicuous Stranger
891         currentLevel.roomObject[index] = DefineObject(99,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
892         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
893         break;
894     case 704: //Lantern Man
895         if (lantern)
896         {
897             DefineObjects(105, y, x);
898         }
899         else
900         {
901             currentLevel.roomObject[index] = DefineObject(99,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
902             currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
903         }
904         break;
905     case 706: //Bonfires
906         SetAnimations(0,0,1,1);
907         currentLevel.roomObject[index] = DefineObject(235,1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
908         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
909         break;
910     case 708: //Breakable Wall
911         currentLevel.roomObject[index] = DefineObject(182,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
912         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
913         break;
914 // Ladders UP
915     case 800: //Ladder 0
916         currentLevel.roomObject[index] = DefineObject(204,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
917         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
918         break;
919     case 801: //Ladder 1
920         currentLevel.roomObject[index] = DefineObject(204,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
921         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
922         break;
923     case 802: //Ladder 2
924         currentLevel.roomObject[index] = DefineObject(204,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
925         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
926         break;
927     case 803: //Ladder 3
928         currentLevel.roomObject[index] = DefineObject(204,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
929         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
930         break;
931     case 804: //Ladder 4
932         currentLevel.roomObject[index] = DefineObject(204,-1,ani,create_win(Sprite_Height,Sprite_Width,y*Sprite_Height,x*Sprite_Width),0,0,y,x,0,object);
```



```

933         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
934         break;
935     case 805: //Ladder 5
936         currentLevel.roomObject[index] = DefineObject(204,-1,ani,create_win(SPRITE_HEIGHT,SPRITE_WIDTH,y*SPRITE_HEIGHT,x*SPRITE_WIDTH),0,0,y,x,0,object);
937         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
938         break;
939     case 806: //Ladder 6
940         currentLevel.roomObject[index] = DefineObject(204,-1,ani,create_win(SPRITE_HEIGHT,SPRITE_WIDTH,y*SPRITE_HEIGHT,x*SPRITE_WIDTH),0,0,y,x,0,object);
941         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
942         break;
943     case 807: //Ladder 7
944         currentLevel.roomObject[index] = DefineObject(204,-1,ani,create_win(SPRITE_HEIGHT,SPRITE_WIDTH,y*SPRITE_HEIGHT,x*SPRITE_WIDTH),0,0,y,x,0,object);
945         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
946         break;
947     case 808: //Ladder 8
948         currentLevel.roomObject[index] = DefineObject(204,-1,ani,create_win(SPRITE_HEIGHT,SPRITE_WIDTH,y*SPRITE_HEIGHT,x*SPRITE_WIDTH),0,0,y,x,0,object);
949         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
950         break;
951     case 809: //Ladder 9
952         currentLevel.roomObject[index] = DefineObject(204,-1,ani,create_win(SPRITE_HEIGHT,SPRITE_WIDTH,y*SPRITE_HEIGHT,x*SPRITE_WIDTH),0,0,y,x,0,object);
953         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
954         break;
955     case 810: //Ladder 10
956         currentLevel.roomObject[index] = DefineObject(204,-1,ani,create_win(SPRITE_HEIGHT,SPRITE_WIDTH,y*SPRITE_HEIGHT,x*SPRITE_WIDTH),0,0,y,x,0,object);
957         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
958         break;
959     case 811: //Ladder 11
960         currentLevel.roomObject[index] = DefineObject(204,-1,ani,create_win(SPRITE_HEIGHT,SPRITE_WIDTH,y*SPRITE_HEIGHT,x*SPRITE_WIDTH),0,0,y,x,0,object);
961         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
962         break;
963     case 812: //Ladder 12
964         currentLevel.roomObject[index] = DefineObject(204,-1,ani,create_win(SPRITE_HEIGHT,SPRITE_WIDTH,y*SPRITE_HEIGHT,x*SPRITE_WIDTH),0,0,y,x,0,object);
965         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
966         break;
967
968 // Ladders DOWN
969     case 813: //Ladder top
970         currentLevel.roomObject[index] = DefineObject(199,-1,ani,create_win(SPRITE_HEIGHT,SPRITE_WIDTH,y*SPRITE_HEIGHT,x*SPRITE_WIDTH),0,0,y,x,0,object);
971         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
972         break;
973     case -801: //Ladder 1
974         currentLevel.roomObject[index] = DefineObject(199,-1,ani,create_win(SPRITE_HEIGHT,SPRITE_WIDTH,y*SPRITE_HEIGHT,x*SPRITE_WIDTH),0,0,y,x,0,object);
975         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
976         break;
977     case -802: //Ladder 2
978         currentLevel.roomObject[index] = DefineObject(199,-1,ani,create_win(SPRITE_HEIGHT,SPRITE_WIDTH,y*SPRITE_HEIGHT,x*SPRITE_WIDTH),0,0,y,x,0,object);

```

```

979         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
980         break;
981     case -803:        //Ladder 3
982         currentLevel.roomObject[index] = DefineObject(199,-1,ani,create_win(SPRITE_HEIGHT,SPRITE_WIDTH,y*SPRITE_HEIGHT,x*SPRITE_WIDTH),0,0,y,x,0,object);
983         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
984         break;
985     case -804:        //Ladder 4
986         currentLevel.roomObject[index] = DefineObject(199,-1,ani,create_win(SPRITE_HEIGHT,SPRITE_WIDTH,y*SPRITE_HEIGHT,x*SPRITE_WIDTH),0,0,y,x,0,object);
987         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
988         break;
989     case -805:        //Ladder 5
990         currentLevel.roomObject[index] = DefineObject(199,-1,ani,create_win(SPRITE_HEIGHT,SPRITE_WIDTH,y*SPRITE_HEIGHT,x*SPRITE_WIDTH),0,0,y,x,0,object);
991         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
992         break;
993     case -806:        //Ladder 6
994         currentLevel.roomObject[index] = DefineObject(199,-1,ani,create_win(SPRITE_HEIGHT,SPRITE_WIDTH,y*SPRITE_HEIGHT,x*SPRITE_WIDTH),0,0,y,x,0,object);
995         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
996         break;
997     case -807:        //Ladder 7
998         currentLevel.roomObject[index] = DefineObject(199,-1,ani,create_win(SPRITE_HEIGHT,SPRITE_WIDTH,y*SPRITE_HEIGHT,x*SPRITE_WIDTH),0,0,y,x,0,object);
999         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
1000        break;
1001     case -808:        //Ladder 8
1002         currentLevel.roomObject[index] = DefineObject(199,-1,ani,create_win(SPRITE_HEIGHT,SPRITE_WIDTH,y*SPRITE_HEIGHT,x*SPRITE_WIDTH),0,0,y,x,0,object);
1003         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
1004         break;
1005     case -809:        //Ladder 9
1006         currentLevel.roomObject[index] = DefineObject(199,-1,ani,create_win(SPRITE_HEIGHT,SPRITE_WIDTH,y*SPRITE_HEIGHT,x*SPRITE_WIDTH),0,0,y,x,0,object);
1007         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
1008         break;
1009     case -810:        //Ladder 10
1010         currentLevel.roomObject[index] = DefineObject(199,-1,ani,create_win(SPRITE_HEIGHT,SPRITE_WIDTH,y*SPRITE_HEIGHT,x*SPRITE_WIDTH),0,0,y,x,0,object);
1011         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
1012         break;
1013     case -811:        //Ladder 11
1014         currentLevel.roomObject[index] = DefineObject(199,-1,ani,create_win(SPRITE_HEIGHT,SPRITE_WIDTH,y*SPRITE_HEIGHT,x*SPRITE_WIDTH),0,0,y,x,0,object);
1015         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
1016         break;
1017     case 999: //Score
1018         currentLevel.roomObject[index] = DefineObject(200,-2,ani,create_win(SPRITE_HEIGHT,SPRITE_WIDTH,y*SPRITE_HEIGHT,x*SPRITE_WIDTH),0,0,y,x,0,999);
1019         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
1020         break;
1021     }
1022 }

```

DrawWindows.cpp

```
1 //Includes
2 #include <curses.h>
3 #include <string.h>
4 #include <Windows.h>
5 #include <fstream>
6 #include <string>
7 #include "GlobalVars.h"
8 #include "ImportMaps.h"
9 #include "Object.h"
10 #include "Interactions.h"
11
12 //Prototypes
13 WINDOW *create_win(int height, int width, int starty, int startx);
14 void DrawMenuOptions(WINDOW *local_win);
15 void destroy_win(WINDOW *local_win);
16 void PrintInMiddle(WINDOW *window, int y, int width, char* string, int color);
17 int do_selection(int sel, int men, WINDOW *local_win);
18 void SetAnimations( int a, int b, int c, int d);
19 void DrawMessage(int y, int x, int w, int h, char* message);
20
21 Object Player;
22
23 //Variables
24 WINDOW *my_win;
25 const int SPRITE_HEIGHT = 6;
26 const int SPRITE_WIDTH = 9;
27 int playerx, playery, ac_playery, ac_playerx;
28 char* word;
29
30 int selection = 0, ch = 0, whichMenu = 0;
31 int player_ani[4] = {0,1,2,1};
32
33 char* string;
34 //Options for the menus stored in arrays. If a menu is longer than the 4 options, a back
35 //"more" option should be made available
36 char* MAIN_MENU[] = { " Start New Game ",
37     " Continue Saved Game ",
38     " Maze Race ",
39     " Exit Game " };
40 char* START_NEW_GAME[] = { " Overwrite Save 1 ",
41     " Overwrite Save 2 ",
42     " Overwrite Save 3 ",
43     " Back " };
44 char* CONTINUE[] = { " Continue Save 1 ",
45     " Continue Save 2 ",
46     " Continue Save 3 ",
```

```

47     " Back " };
48
49 int MainMenu()
50 {
51     int row, col;
52     getmaxyx(stdscr, row, col); //Get the max rows and columns
53
54     my_win = create_win(11, 40, (row/2)-15, (col/2)-20); //Create a window
55     wborder(my_win, '|', '|', '-', '-', '*', '*', '*', '*');
56
57     DrawMenuOptions(my_win); //Draw the options
58
59     ch = getch();
60
61     //Test for which key has been pressed
62     switch(ch)
63     {
64     case KEY_UP:
65         selection--;
66         break;
67     case KEY_DOWN:
68         selection++;
69         break;
70     case 13: //because KEY_ENTER returns \n
71         return do_selection((selection % 4), whichMenu, my_win);
72         break;
73     case 27: //Escape
74         gameState = -1;
75         destroy_win(my_win);
76     default:
77         break;
78     }
79
80     return 1;
81 }
82
83 WINDOW *create_win(int height, int width, int starty, int startx)
84 { //This creates a window of specified height and width at the y and x coordinates
85     WINDOW *local_win;
86     local_win = newwin(height, width, starty, startx);
87     box(local_win, 0, 0);
88     //custom border for the window
89     wborder(local_win, '|', '|', '|', '|', '|', '|', '|', '|', '|');
90     wnoutrefresh(local_win);
91     return local_win;
92 }

```

```
93
94 void destroy_win(WINDOW *local_win)
95 { //Destroy the window function
96     wborder(local_win, ' ', ' ', ' ', ' ', ' ', ' ', ' ', ' ', ' ', ' ', ' ');
97     werase(local_win);
98     wnoutrefresh(local_win);
99     delwin(local_win);
100 }
101
102 void DrawMenuOptions(WINDOW *local_win)
103 {
104     if (selection == 0) //This is needed because for some reason,
105     { //modulos doesnt work with negative numbers that aren't of the multiple you are testing..
106         selection = 4;
107     }
108
109     for (int i=0; i < 4; i++) //Check which menu to draw, and draw all 4 options
110     {
111         if ((selection % 4) == i) //Selected
112         {
113             switch(whichMenu)
114             {
115                 case 0: // Main Menu
116                     PrintInMiddle(my_win, 2 + (i*2), 40, MAIN_MENU[i], 1);
117                     break;
118
119                 case 1: // NewGame Menu
120                     PrintInMiddle(my_win, 2 + (i*2), 40, START_NEW_GAME[i], 1);
121                     break;
122
123                 case 2: // ContinueGame Menu
124                     PrintInMiddle(my_win, 2 + (i*2), 40, CONTINUE[i], 1);
125                     break;
126
127             }
128         }
129         else //Not selected
130         {
131             switch(whichMenu)
132             {
133                 case 0: // Main Menu
134                     PrintInMiddle(my_win, 2 + (i*2), 40, MAIN_MENU[i], 2);
135                     break;
136
137                 case 1: // NewGame Menu
138                     PrintInMiddle(my_win, 2 + (i*2), 40, START_NEW_GAME[i], 2);
```

```

139             break;
140
141         case 2: // ContinueGame Menu
142             PrintInMiddle(my_win, 2 + (i*2), 40, CONTINUE[i], 2);
143             break;
144
145     }
146 }
147
148 wrefresh(local_win); //Refresh the window
149 }
150
151 void PrintInMiddle(WINDOW *local_win, int y, int width, char* string, int color)
152 { // A simple function to print in the given color, in the middle of the given window (stdscr is no window)
153     wattron(local_win, COLOR_PAIR(color));
154
155     int middleX = (width/2)-strlen(string)/2;
156
157     mvwprintw(local_win,y,middleX, "%s", string);
158
159     wattroff(local_win, COLOR_PAIR(color));
160     wrefresh(local_win); //Refresh the window
161 }
162
163 int do_selection(int sel, int men, WINDOW *local_win) //Do the action needed for the menu
164 {
165     switch(men)
166     {
167     case 0: //Main Menu MENU
168         switch(sel)
169         {
170         case 0://Start New Game
171             printf("\a");
172             whichMenu = 1;
173             selection = 0;
174             break;
175
176         case 1://Continue Saved Game
177             printf("\a");
178             whichMenu = 2;
179             selection = 0;
180             break;
181
182         case 2://MazeRace
183             printf("\a");
184             gameState = 1;

```

```

185         destroy_win(local_win);
186         //Create the player
187         playerx = 1 * SPRITE_WIDTH;
188         playery = 1 * SPRITE_HEIGHT;
189         Player = DefineObject(13,-1,player_ani,create_win(SPRITE_HEIGHT,SPRITE_WIDTH,playery,playerx),0,0,playery,playerx,0,0);
190         //Draw the Map
191         currentLevel.GenerateMaze();
192         break;
193
194     case 3://Exit Game
195         printf("\a");
196         return 0; //This value is sent all the way back to the MainGameLoop.cpp
197         break; //and means the game loop is to be exited, and the game is quit
198
199     }
200     break;
201
202 case 1: //Start New Game MENU
203     switch(sel)
204     {
205     case 0://Overwrite save 1
206         save = 1;
207         printf("\a");
208         gameState = 1;
209         destroy_win(local_win);
210         //Create the player
211         playerx = 10 * SPRITE_WIDTH;
212         playery = 4 * SPRITE_HEIGHT;
213         Player = DefineObject(13,-1,player_ani,create_win(SPRITE_HEIGHT,SPRITE_WIDTH,playery,playerx),0,0,playery,playerx,0,0);
214         //Draw the Map
215         return currentLevel.ImportMap(roomX,roomY,"playerhousebedroom1");
216         break;
217
218     case 1://Overwrite save 2
219         save = 2;
220         printf("\a");
221         gameState = 1;
222         destroy_win(local_win);
223         //Create the player
224         playerx = 10 * SPRITE_WIDTH;
225         playery = 4 * SPRITE_HEIGHT;
226         Player = DefineObject(13,-1,player_ani,create_win(SPRITE_HEIGHT,SPRITE_WIDTH,playery,playerx),0,0,playery,playerx,0,0);
227         //Draw the Map
228         return currentLevel.ImportMap(roomX,roomY,"playerhousebedroom1");
229         break;
230

```

```
231     case 2://Overwrite save 3
232         save = 3;
233         printf("\a");
234         gameState = 1;
235         destroy_win(local_win);
236         //Create the player
237         playerx = 10 * SPRITE_WIDTH;
238         playery = 4 * SPRITE_HEIGHT;
239         Player = DefineObject(13,-1,player_ani,create_win(SPRITE_HEIGHT,SPRITE_WIDTH,playery,playerx),0,0,playery,playerx,0,0);
240         //Draw the Map
241         return currentLevel.ImportMap(roomX,roomY,"playerhousebedroom1");
242         break;
243
244     case 3://Back
245         printf("\a");
246         whichMenu = 0;
247         selection = 0;
248         break;
249
250     }
251     break;
252 case 2: //Continue Saved Game MENU
253     switch(sel)
254     {
255     case 0://Continue Save 1
256
257         break;
258
259     case 1://Continue Save 2
260         printf("\a");
261         break;
262
263     case 2://Continue Save 3
264         printf("\a");
265         break;
266
267     case 3://Back
268         printf("\a");
269         whichMenu = 0;
270         selection = 1;
271         break;
272
273     }
274     break;
275
276 }
```



```
277         return 1;
278     }
279
280     int DrawBackground(char* file, int doGet)
281     {
282         std::string word;
283         std::ifstream fileName;
284         fileName.open(file); //Open the File
285         fileName >> word;
286
287         if (fileName.is_open())
288         {
289             while (fileName.good())
290             {
291                 for (int i = 0; i < 66; i ++)//66 is the # of rows
292                 {
293                     for (int j = 0; j < 199; j ++)//199 is the # of columns
294                     {
295                         if (word[j] != 'G')
296                             mvwaddch(stdscr,i,j,word[j]);
297                     }
298                     fileName >> word;
299                 }
300                 fileName.close(); //Close the file
301             }
302         }
303         else
304         {
305             printf("The File the computer is trying to load doesn't exist\n"); //Apologize
306             printf("Try re-installing the game\n");
307             return 0;
308         }
309
310         wnoutrefresh(stdscr);
311         doupdate();
312
313         if (doGet)
314         {
315             getch();
316             gameState = 0;
317         }
318         return 1;
319     }
320
321     void DrawMessage(int y, int x, int w, int h, char* message)
322     {
```

```
323     int row, col;
324     getmaxyx(stdscr, row, col); //Get the max rows and columns
325     int _x = x, _y = y;
326
327     if (x == -1)
328         _x = (col/2)-(w/2);
329     if (y == -1)
330         _y = (row/3)-(h/2);
331
332     WINDOW *shadow_win;
333     WINDOW *local_win;
334
335     shadow_win = create_win(h+2,w+2,_y-1,_x-1);
336     local_win = create_win(h,w,_y,_x);
337
338     printf("\a");
339
340     for (int i=0; i<h; i++)
341     {
342         for (int j=0; j<(w-4); j++)
343         {
344             if (i*(w-4)+j < strlen(message))
345                 mvwaddch(local_win,i+1,j+2,message[(i*(w-4))+j]);
346         }
347     }
348
349     wborder(local_win, '|', '|', '-', '-', '*', '*', '*', '*');
350
351     wrefresh(shadow_win);
352     wrefresh(local_win);
353
354     while (!EnterWasPressed())
355         local_win = local_win;
356
357     destroy_win(local_win);
358 }
359
360 int DrawInventory(char* title, int item, int price)
361 {
362     keypad(stdscr, TRUE);
363
364     WINDOW *shadow_win;
365     WINDOW *local_win;
366
367     shadow_win = create_win(42,122,1,1);
368     local_win = create_win(40,120,2,2);
```

```
369
370     printf("\a");
371
372     mvwprintw(local_win,3,4,title);
373
374     mvwprintw(local_win,6,4,"Press <ESC> to exit the game");
375     mvwprintw(local_win,8,4,"Press <F1> to save your game");
376     mvwprintw(local_win,10,4,"Press <F2> to return to the Main Menu");
377
378
379     mvwprintw(local_win,14,4,"Helpful Information:"); // Help information
380     mvwprintw(local_win,16,4,"Arrow Keys to move, and Enter to interact with objects"); // Help information
381     mvwprintw(local_win,18,4,"[] : Walk into blocks to push them."); // Help information
382     mvwprintw(local_win,19,4,"// : Walk on ice to slide."); // Help information
383     if (item != -1)
384         mvwprintw(local_win,12,4,"Press <Enter> to buy a %s for %d gold",global_item[item],price);
385
386     mvwprintw(local_win,3,60,"Gold: %d", money);
387     mvwprintw(local_win,3,80,"Save %d", save);
388     mvwprintw(local_win,3,100,"Floor %d of %d",progress,totalfloors);
389
390     wborder(local_win, '|', '|', '-', '-', '*', '*', '*', '*');
391     nonl();
392
393     int ch = getch();
394     wrefresh(shadow_win);
395     wrefresh(local_win);
396
397     while (!InventoryWasPressed())
398     {
399         //Test for which key has been pressed
400         switch(ch)
401         {
402             case KEY_F(1):
403                 //Damian, you need to add the save game here
404                 printf("\a");
405                 break;
406
407             case KEY_F(2):
408                 gameState = 0;
409                 printf("\a");
410                 return 1;
411                 break;
412
413             case 27: //Escape
414                 return 0;
```

```
415                 break;
416
417                 default:
418                     break;
419             }
420
421             ch = getch();
422             wrefresh(shadow_win);
423             wrefresh(local_win);
424         }
425         destroy_win(local_win);
426
427         return 1;
428 }
```

ImportMaps.cpp

```
1 //Includes
2 #include <urses.h>
3 #include <fstream>
4 #include <sstream>
5 #include <stdlib.h>
6 #include <Windows.h>
7 #include <stdio.h>
8 #include <time.h>
9 #include "ImportMaps.h"
10 #include "CreateObject.h"
11 #include "DrawWindows.h"
12 #include "GlobalVars.h"
13 #include "MazeFilter.h"
14
15 //Prototypes
16 std::string int_to_string(int number);
17 int CurrentCell(char xory, int SmallIndex);
18 bool Visited(int x, int y);
19 int SmallIndex(int x, int y);
20
21 //Variables
22 const int MAP_WIDTH = 22;
23 const int MAP_HEIGHT = 11;
24
25 //MazeGeneration
26 bool visited[10][5]; //each cell
27 int currentCell;
28 int visitedCells;
29 int randomNum;
```

```

30 int randomCells[4];
31 int cX, cY, toErase;
32
33 int Level::ImportMap(int x, int y, std::string mapName)
34 {
35     std::string file;
36     int word;
37
38     if (mapName == "NULL") //Concatenation process
39     {
40         std::string result = "Maps\\" + int_to_string(x) + "," + int_to_string(y) + ".txt";
41         //Concatate the file location.
42         file = result;
43         id = 1;
44     }
45     else
46     {
47         std::string result = "Maps\\" + mapName + ".txt";
48         //Concatate the file location
49         file = result;
50         id = 0;
51     }
52
53     std::ifstream fileName;
54     fileName.open(file); //Open the File
55     if (fileName.is_open())
56     {
57         while (fileName.good())
58         {
59             for (int i=0; i < MAP_HEIGHT; i++)
60             {
61                 for (int j=0; j < MAP_WIDTH; j++)
62                 {
63                     fileName >> word;
64                     DefineObjects(word,i,j);
65                 }
66             }
67             fileName.close(); //Close the file
68         }
69     }
70     else
71     {
72         printf("The Map File the computer is trying to load doesn't exist\n"); //Apologize
73         printf("Try re-installing the game\n");
74         getch();
75         return 0; //Close game if can't import file

```

```

76     }
77     return 1;
78 }
79
80 void Level::ToggleLights()
81 {
82     lights = !lights;
83 }
84
85 std::string int_to_string (int number)
86 {
87     std::stringstream ss;
88     ss << number;
89     std::string newString = ss.str();
90     return newString;
91 }
92
93 /*      From Wikipedia - Recursive Backtrack method
94
95     Make the initial cell the current cell and mark it as visited
96     While there are unvisited cells
97     If the current cell has any neighbours which have not been visited
98         Choose randomly one of the unvisited neighbours
99         Push the current cell to the stack
100        Remove the wall between the current cell and the chosen cell
101        Make the chosen cell the current cell and mark it as visited
102    Else if stack is not empty
103        Pop a cell from the stack
104        Make it the current cell
105    Else
106        Pick a random cell, make it the current cell and mark it as visited */
107
108 void Level::GenerateMaze()
109 {
110     for (int i = 0; i < 10; i++)
111         for (int j = 0; j < 5; j++) //everything will be a wall (1)
112             visited[j][i] = false;
113     visitedCells = 1;
114
115     //make a blank maze of all walls
116     for (int i = 0; i < 11; i++)
117         for (int j = 0; j < 22; j++) //everything will be a wall (1)
118             {
119                 if (j != 21)
120                     currentLevel.roomObject[(i*22)+j].SetObjIndex(1);
121                 else

```

```
122         currentLevel.roomObject[(i*22)+j].SetObjIndex(0);
123     }
124
125     //"Make the initial cell the current cell and mark it as visited"
126     visited[0][0] = true;
127     currentCell = 0;
128     currentLevel.roomObject[23].SetObjIndex(0);
129
130     srand (static_cast<unsigned int>(time(NULL))); //new random number
131
132     //"While there are unvisited cells"
133     while (visitedCells != 50)
134     {
135         cX = CurrentCell('x', currentCell);
136         cY = CurrentCell('y', currentCell);
137
138         //"If the current cell has any neighbours which have not been visited"
139         if (!(Visited(cX,cY-1) == true && Visited(cX,cY+1) == true && Visited(cX+1,cY) == true && Visited(cX-1,cY) == true))
140         {
141             randomNum = 0;
142             //"Choose randomly one of the unvisited neighbours"
143             if (Visited(cX,cY-1) == false) // Above
144             {
145                 randomCells[randomNum] = 0;
146                 randomNum ++;
147             }
148             if (Visited(cX,cY+1) == false) // Below
149             {
150                 randomCells[randomNum] = 1;
151                 randomNum ++;
152             }
153             if (Visited(cX+1,cY) == false) // Right
154             {
155                 randomCells[randomNum] = 2;
156                 randomNum ++;
157             }
158             if (Visited(cX-1,cY) == false) // Left
159             {
160                 randomCells[randomNum] = 3;
161                 randomNum ++;
162             }
163
164             randomNum = rand() % randomNum;
165             //"Push the current cell to the stack"
166             //"Remove the wall between the current cell and the chosen cell"
167
```

```

168         switch (randomCells[randomNum])
169         {
170             case 0:
171                 currentCell = SmallIndex(cX, cY-1);
172                 cY --;
173                 toErase = ((cY*2)+2)*22+(cX*2)+1;
174                 currentLevel.roomObject[toErase].SetObjIndex(0);
175                 break;
176             case 1:
177                 currentCell = SmallIndex(cX, cY+1);
178                 cY ++;
179                 toErase = ((cY*2)*22)+(cX*2)+1;
180                 currentLevel.roomObject[toErase].SetObjIndex(0);
181                 break;
182             case 2:
183                 currentCell = SmallIndex(cX+1, cY);
184                 cX ++;
185                 toErase = ((cY*2)+1)*22+(cX*2);
186                 currentLevel.roomObject[toErase].SetObjIndex(0);
187                 break;
188             case 3:
189                 currentCell = SmallIndex(cX-1, cY);
190                 cX --;
191                 toErase = ((cY*2)+1)*22+(cX*2)+2;
192                 currentLevel.roomObject[toErase].SetObjIndex(0);
193                 break;
194         }
195         // "Make the chosen cell the current cell and mark it as visited"
196         visited[CurrentCell('x', currentCell)][CurrentCell('y', currentCell)] = true;
197
198         toErase = ((cY*2)+1)*22+(cX*2)+1;
199         currentLevel.roomObject[toErase].SetObjIndex(0);
200         visitedCells ++;
201     }
202     // "Else if stack is not empty"
203     else
204     {
205         for(int i = 0; i < 10; i ++)
206         {
207             for(int j = 0; j < 5; j ++)
208             {
209                 if (visited[i][j] == true && !(Visited(i, j-1) == true && Visited(i, j+1) == true && Visited(i+1, j) == true && Visited(i-1, j) == true))
210                 {
211                     currentCell = SmallIndex(i, j);
212                 }
213             }

```



```
214         }
215     }
216 }
217 MazeFilter();
218 }
219
220 int CurrentCell(char xory, int index)
221 {
222     if (xory == 'x')
223     {
224         return index % MAP_WIDTH;
225     }
226     else if (xory == 'y')
227     {
228         return (index - (index % MAP_WIDTH)) / 10;
229     }
230     return 0;
231 }
232
233 bool Visited (int x, int y)
234 {
235     if (x >= 0 && x <= 9 && y >= 0 && y <= 4)
236         return visited[x][y];
237     return true;
238 }
239
240 int SmallIndex(int x, int y)
241 {
242     return (y * MAP_WIDTH) + x;
243 }
244
245 Level::Level()
246 {
247     lights = true;
248 }
```

Interactions.cpp

```
1 //Includes
2 #include "Interactions.h"
3 #include "ImportMaps.h"
4 #include "DrawWindows.h"
5 #include "GlobalVars.h"
6 #include "CreateObject.h"
7 #include <Windows.h>
8 // #include "Inventory.h"
```

```
9
10 //Variables
11 const int SPRITE_WIDTH = 9;
12 const int SPRITE_HEIGHT = 6;
13 bool lantern = false;
14 bool hammer = false;
15 int finalstep;
16 int progress = 0;
17 int totalfloors = 100;
18
19 void Interaction(int obj, int index)
20 {
21
22     switch(obj)
23     {
24         case 301: //Player Bedroom 1
25             printf("\a");
26             Player.SetX(14 * SPRITE_WIDTH);
27             Player.SetY(6 * SPRITE_HEIGHT);
28             Player.SetDirection(2);
29             destroy_win(Player.objectWindow);
30
31             Player.objectWindow = create_win(6,9,Player.GetY(),Player.GetX());
32             Player.DrawSprite(Player,0);
33             currentLevel.ImportMap(0,0,"playerhouseentrance");
34             break;
35
36         case 302: //Player Entrance
37             printf("\a");
38             Player.SetX(12 * SPRITE_WIDTH);
39             Player.SetY(5 * SPRITE_HEIGHT);
40             Player.SetDirection(3);
41             destroy_win(Player.objectWindow);
42
43             Player.objectWindow = create_win(6,9,Player.GetY(),Player.GetX());
44             Player.DrawSprite(Player,0);
45             currentLevel.ImportMap(0,0,"playerhousebedroom1");
46             break;
47
48         case 303: //Player Bedroom 3
49             printf("\a");
50             Player.SetX(18 * SPRITE_WIDTH);
51             Player.SetY(5 * SPRITE_HEIGHT);
52             Player.SetDirection(3);
53             destroy_win(Player.objectWindow);
54
```

```
55         Player.objectWindow = create_win(6,9,Player.GetY(),Player.GetX());
56         Player.DrawSprite(Player,0);
57         currentLevel.ImportMap(0,0,"playerhouseentrance");
58         break;
59
60     case 304: //Player Entrance
61         printf("\a");
62         Player.SetX(8 * SPRITE_WIDTH);
63         Player.SetY(4 * SPRITE_HEIGHT);
64         Player.SetDirection(2);
65         destroy_win(Player.objectWindow);
66
67         Player.objectWindow = create_win(6,9,Player.GetY(),Player.GetX());
68         Player.DrawSprite(Player,0);
69         currentLevel.ImportMap(0,0,"playerhousebedroom3");
70         break;
71
72     case 305: //Player Entrance
73         printf("\a");
74         Player.SetX(7 * SPRITE_WIDTH);
75         Player.SetY(5 * SPRITE_HEIGHT);
76         Player.SetDirection(0);
77         destroy_win(Player.objectWindow);
78
79         Player.objectWindow = create_win(6,9,Player.GetY(),Player.GetX());
80         Player.DrawSprite(Player,0);
81         roomX = 2;
82         roomY = 1;
83         currentLevel.ImportMap(2,1,"NULL");
84         break;
85
86     case 306: //2,1 to Player House
87         printf("\a");
88         Player.SetX(16 * SPRITE_WIDTH);
89         Player.SetY(7 * SPRITE_HEIGHT);
90         Player.SetDirection(1);
91         destroy_win(Player.objectWindow);
92
93         Player.objectWindow = create_win(6,9,Player.GetY(),Player.GetX());
94         Player.DrawSprite(Player,0);
95         currentLevel.ImportMap(0,0,"playerhouseentrance");
96         break;
97
98     case 307: //Player Entrance
99         printf("\a");
100        Player.SetX(10 * SPRITE_WIDTH);
```

```
101         Player.SetY(6 * SPRITE_HEIGHT);
102         Player.SetDirection(1);
103         destroy_win(Player.objectWindow);
104
105         Player.objectWindow = create_win(6, 9, Player.GetY(), Player.GetX());
106         Player.DrawSprite(Player, 0);
107         currentLevel.ImportMap(0, 0, "playerhousebedroom2");
108         break;
109
110     case 308: //Player Bedroom 2
111         printf("\a");
112         Player.SetX(11 * SPRITE_WIDTH);
113         Player.SetY(2 * SPRITE_HEIGHT);
114         Player.SetDirection(0);
115         destroy_win(Player.objectWindow);
116
117         Player.objectWindow = create_win(6, 9, Player.GetY(), Player.GetX());
118         Player.DrawSprite(Player, 0);
119         currentLevel.ImportMap(0, 0, "playerhouseentrance");
120         break;
121
122     case 309: //Cave Entrance
123         if (lantern)
124         {
125             printf("\a");
126             currentLevel.ToggleLights();
127             Player.SetX(8 * SPRITE_WIDTH);
128             Player.SetY(9 * SPRITE_HEIGHT);
129             Player.SetDirection(1);
130             destroy_win(Player.objectWindow);
131
132             Player.objectWindow = create_win(6, 9, Player.GetY(), Player.GetX());
133             Player.DrawSprite(Player, 0);
134             currentLevel.ImportMap(2, 1, "FLOOR 0");
135         }
136         else
137         {
138             Player.SetDirection(3);
139             Player.DrawSprite(Player, 0);
140             DrawMessage(-1, -1, 40, 8, "Nervous Man: Wait! You can't go in there, it's too dark.");
141             DrawMessage(-1, -1, 40, 8, "Nervous Man: Here, take my lantern. I'm too scared to go in anyways.");
142             DrawMessage(-1, -1, 40, 8, "You obtained LANTERN!");
143             lantern = true;
144         }
145         break;
146
```

```
147 case 310: //Cave Exit
148     printf("\a");
149     currentLevel.ToggleLights();
150     Player.SetX(10 * SPRITE_WIDTH);
151     Player.SetY(3 * SPRITE_HEIGHT);
152     Player.SetDirection(0);
153     destroy_win(Player.objectWindow);
154
155     Player.objectWindow = create_win(6,9,Player.GetY(),Player.GetX());
156     Player.DrawSprite(Player,0);
157     currentLevel.ImportMap(1,1,"NULL");
158     break;
159
160 case 311: //Alchemist IN
161     DrawMessage(-1,-1,40,8,"The Alchemist seems to have spilled something corrosive on the door knob...");
162     //printf("\a");
163     //Player.SetX(9 * SPRITE_WIDTH);
164     //Player.SetY(6 * SPRITE_HEIGHT);
165     //Player.SetDirection(1);
166     //destroy_win(Player.objectWindow);
167
168     //Player.objectWindow = create_win(6,9,Player.GetY(),Player.GetX());
169     //Player.DrawSprite(Player,0);
170     //currentLevel.ImportMap(1,1,"Alchemist");
171     break;
172
173 case 312: //Alchemist OUT
174     printf("\a");
175     Player.SetX(18 * SPRITE_WIDTH);
176     Player.SetY(3 * SPRITE_HEIGHT);
177     Player.SetDirection(0);
178     destroy_win(Player.objectWindow);
179
180     Player.objectWindow = create_win(6,9,Player.GetY(),Player.GetX());
181     Player.DrawSprite(Player,0);
182     currentLevel.ImportMap(2,1,"NULL");
183     break;
184
185 case 313: //Survivalist IN
186     DrawMessage(-1,-1,40,8,"There's a sign on the door saying 'Out adventuring'...");
187     //printf("\a");
188     //Player.SetX(9 * SPRITE_WIDTH);
189     //Player.SetY(6 * SPRITE_HEIGHT);
190     //Player.SetDirection(1);
191     //destroy_win(Player.objectWindow);
192
```

```
193         //Player.objectWindow = create_win(6,9,Player.GetY(),Player.GetX());
194         //Player.DrawSprite(Player,0);
195         //currentLevel.ImportMap(1,1,"Survivalist");
196         break;
197
198     case 314: //Survivalist OUT
199         //printf("\a");
200         //Player.SetX(15 * SPRITE_WIDTH);
201         //Player.SetY(8 * SPRITE_HEIGHT);
202         //Player.SetDirection(0);
203         //destroy_win(Player.objectWindow);
204
205         //Player.objectWindow = create_win(6,9,Player.GetY(),Player.GetX());
206         //Player.DrawSprite(Player,0);
207         //currentLevel.ImportMap(2,1,"NULL");
208         break;
209
210     case 315: //Locksmith IN
211         DrawMessage(-1,-1,40,8,"The Locksmith seems to have locked himself in...");
212         //printf("\a");
213         //Player.SetX(9 * SPRITE_WIDTH);
214         //Player.SetY(6 * SPRITE_HEIGHT);
215         //Player.SetDirection(1);
216         //destroy_win(Player.objectWindow);
217
218         //Player.objectWindow = create_win(6,9,Player.GetY(),Player.GetX());
219         //Player.DrawSprite(Player,0);
220         //currentLevel.ImportMap(1,1,"Locksmith");
221         break;
222
223     case 316: //Locksmith OUT
224         printf("\a");
225         Player.SetX(12 * SPRITE_WIDTH);
226         Player.SetY(3 * SPRITE_HEIGHT);
227         Player.SetDirection(0);
228         destroy_win(Player.objectWindow);
229
230         Player.objectWindow = create_win(6,9,Player.GetY(),Player.GetX());
231         Player.DrawSprite(Player,0);
232         currentLevel.ImportMap(2,1,"NULL");
233         break;
234
235     case 317: //Bombcrafter IN
236         DrawMessage(-1,-1,40,8,"Nobody seem's to be in...");
237         //printf("\a");
238         Player.SetX(9 * SPRITE_WIDTH);
```

```
239         Player.SetY(6 * SPRITE_HEIGHT);
240         Player.SetDirection(1);
241         destroy_win(Player.objectWindow);
242         Player.objectWindow = create_win(6,9,Player.GetY(),Player.GetX());
243         Player.DrawSprite(Player,0);
244         currentLevel.ImportMap(1,1,"Bombcrafter");
245         break;*/
246         //Inventory();
247         break;
248
249     case 318: //Bombcrafter OUT
250         printf("\a");
251         Player.SetX(3 * SPRITE_WIDTH);
252         Player.SetY(3 * SPRITE_HEIGHT);
253         Player.SetDirection(0);
254         destroy_win(Player.objectWindow);
255
256         Player.objectWindow = create_win(6,9,Player.GetY(),Player.GetX());
257         Player.DrawSprite(Player,0);
258         currentLevel.ImportMap(2,1,"NULL");
259         break;
260
261     //Ladders UP
262     case 800: //Floor 0
263         printf("\a");
264         Player.SetX(1 * SPRITE_WIDTH);
265         Player.SetY(1 * SPRITE_HEIGHT);
266         Player.SetDirection(0);
267         destroy_win(Player.objectWindow);
268
269         Player.objectWindow = create_win(6,9,Player.GetY(),Player.GetX());
270         Player.DrawSprite(Player,0);
271         currentLevel.ImportMap(0,0,"FLOOR 1");
272         break;
273
274     case 801: //Floor 1
275         progress = 1;
276         printf("\a");
277         Player.SetX(9 * SPRITE_WIDTH);
278         Player.SetY(5 * SPRITE_HEIGHT);
279         Player.SetDirection(0);
280         destroy_win(Player.objectWindow);
281
282         Player.objectWindow = create_win(6,9,Player.GetY(),Player.GetX());
283         Player.DrawSprite(Player,0);
284         currentLevel.ImportMap(0,0,"FLOOR 2");
```

```
285         break;
286
287     case 802: //Floor 2
288         progress = 2;
289         printf("\a");
290         Player.SetX(1 * SPRITE_WIDTH);
291         Player.SetY(1 * SPRITE_HEIGHT);
292         Player.SetDirection(0);
293         destroy_win(Player.objectWindow);
294
295         Player.objectWindow = create_win(6, 9, Player.GetY(), Player.GetX());
296         Player.DrawSprite(Player, 0);
297         currentLevel.ImportMap(0, 0, "FLOOR 3");
298         break;
299
300     case 803: //Floor 3
301         progress = 3;
302         printf("\a");
303         Player.SetX(17 * SPRITE_WIDTH);
304         Player.SetY(1 * SPRITE_HEIGHT);
305         Player.SetDirection(0);
306         destroy_win(Player.objectWindow);
307
308         Player.objectWindow = create_win(6, 9, Player.GetY(), Player.GetX());
309         Player.DrawSprite(Player, 0);
310         currentLevel.ImportMap(0, 0, "FLOOR 4");
311         break;
312
313     case 804: //Floor 4
314         progress = 4;
315         printf("\a");
316         Player.SetX(2 * SPRITE_WIDTH);
317         Player.SetY(9 * SPRITE_HEIGHT);
318         Player.SetDirection(0);
319         destroy_win(Player.objectWindow);
320
321         Player.objectWindow = create_win(6, 9, Player.GetY(), Player.GetX());
322         Player.DrawSprite(Player, 0);
323         currentLevel.ImportMap(0, 0, "FLOOR 5");
324         break;
325
326     case 805: //Floor
327         progress = 5;
328         printf("\a");
329         Player.SetX(18 * SPRITE_WIDTH);
330         Player.SetY(3 * SPRITE_HEIGHT);
```



```
331         Player.SetDirection(0);
332         destroy_win(Player.objectWindow);
333
334         Player.objectWindow = create_win(6,9,Player.GetY(),Player.GetX());
335         Player.DrawSprite(Player,0);
336         currentLevel.ImportMap(0,0,"FLOOR 7");
337         break;
338
339     case 806: //Floor 6
340         progress = 6;
341         printf("\a");
342         Player.SetX(18 * SPRITE_WIDTH);
343         Player.SetY(3 * SPRITE_HEIGHT);
344         Player.SetDirection(0);
345         destroy_win(Player.objectWindow);
346         Player.objectWindow = create_win(6,9,Player.GetY(),Player.GetX());
347         Player.DrawSprite(Player,0);
348         currentLevel.ImportMap(0,0,"FLOOR 7");
349         break;
350
351     case 807: //Floor 7
352         progress = 7;
353         printf("\a");
354         Player.SetX(3 * SPRITE_WIDTH);
355         Player.SetY(9 * SPRITE_HEIGHT);
356         Player.SetDirection(0);
357         destroy_win(Player.objectWindow);
358         Player.objectWindow = create_win(6,9,Player.GetY(),Player.GetX());
359         Player.DrawSprite(Player,0);
360         currentLevel.ImportMap(0,0,"FLOOR 8");
361         break;
362
363     case 808: //Floor 8
364         progress = 8;
365         printf("\a");
366         Player.SetX(1 * SPRITE_WIDTH);
367         Player.SetY(6 * SPRITE_HEIGHT);
368         Player.SetDirection(0);
369         destroy_win(Player.objectWindow);
370         Player.objectWindow = create_win(6,9,Player.GetY(),Player.GetX());
371         Player.DrawSprite(Player,0);
372         currentLevel.ImportMap(0,0,"FLOOR 9");
373         break;
374
375     case 809: //Floor 9
376         progress = 9;
```

```
377     printf("\a");
378     Player.SetX(19 * SPRITE_WIDTH);
379     Player.SetY(5 * SPRITE_HEIGHT);
380     Player.SetDirection(0);
381     destroy_win(Player.objectWindow);
382     Player.objectWindow = create_win(6, 9, Player.GetY(), Player.GetX());
383     Player.DrawSprite(Player, 0);
384     currentLevel.ImportMap(0, 0, "FLOOR 10");
385     break;
386
387 case 810: //Floor 10
388     progress = 10;
389     printf("\a");
390     Player.SetX(2 * SPRITE_WIDTH);
391     Player.SetY(8 * SPRITE_HEIGHT);
392     Player.SetDirection(0);
393     destroy_win(Player.objectWindow);
394
395     Player.objectWindow = create_win(6, 9, Player.GetY(), Player.GetX());
396     Player.DrawSprite(Player, 0);
397     currentLevel.ImportMap(0, 0, "FLOOR 11");
398     break;
399
400 case 811: //Floor 11
401     progress = 11;
402     printf("\a");
403     Player.SetX(1 * SPRITE_WIDTH);
404     Player.SetY(9 * SPRITE_HEIGHT);
405     Player.SetDirection(0);
406     destroy_win(Player.objectWindow);
407
408     Player.objectWindow = create_win(6, 9, Player.GetY(), Player.GetX());
409     Player.DrawSprite(Player, 0);
410     finalstep = Step;
411     currentLevel.ImportMap(0, 0, "TEMP 5");
412     break;
413 case 812: //Floor 12
414     progress = 12;
415     printf("\a");
416     Player.SetX(18 * SPRITE_WIDTH);
417     Player.SetY(9 * SPRITE_HEIGHT);
418     Player.SetDirection(0);
419     destroy_win(Player.objectWindow);
420
421     Player.objectWindow = create_win(6, 9, Player.GetY(), Player.GetX());
422     Player.DrawSprite(Player, 0);
```

```
423         finalstep = Step;
424         currentLevel.ImportMap(0,0,"TEMP 7");
425         break;
426 //Ladders DOWN
427     case -800: //Floor 0
428         printf("\a");
429         Player.SetX(1 * SPRITE_WIDTH);
430         Player.SetY(1 * SPRITE_HEIGHT);
431         Player.SetDirection(0);
432         destroy_win(Player.objectWindow);
433
434         Player.objectWindow = create_win(6,9,Player.GetY(),Player.GetX());
435         Player.DrawSprite(Player,0);
436         currentLevel.ImportMap(0,0,"FLOOR 0");
437         break;
438
439     case 813: //Top
440         printf("\a");
441         Player.SetX(10 * SPRITE_WIDTH);
442         Player.SetY(7 * SPRITE_HEIGHT);
443         Player.SetDirection(1);
444         destroy_win(Player.objectWindow);
445
446         Player.objectWindow = create_win(6,9,Player.GetY(),Player.GetX());
447         Player.DrawSprite(Player,0);
448         currentLevel.ImportMap(0,9,"0,10");
449         break;
450
451     //case -802: //Floor 2
452     //     printf("\a");
453     //     Player.SetX(1 * SPRITE_WIDTH);
454     //     Player.SetY(1 * SPRITE_HEIGHT);
455     //     Player.SetDirection(0);
456     //     destroy_win(Player.objectWindow);
457
458     //     Player.objectWindow = create_win(6,9,Player.GetY(),Player.GetX());
459     //     Player.DrawSprite(Player,0);
460     //     currentLevel.ImportMap(0,0,"FLOOR 2");
461     //     break;
462
463     //case -803: //TEMP 3
464     //     printf("\a");
465     //     Player.SetX(19 * SPRITE_WIDTH);
466     //     Player.SetY(3 * SPRITE_HEIGHT);
467     //     Player.SetDirection(0);
468     //     destroy_win(Player.objectWindow);
```

```
469
470 //      Player.objectWindow = create_win(6,9,Player.GetY(),Player.GetX());
471 //      Player.DrawSprite(Player,0);
472 //      currentLevel.ImportMap(0,0,"TEMP 3");
473 //      break;
474
475 //case -804: //TEMP 4
476 //      printf("\a");
477 //      Player.SetX(1 * SPRITE_WIDTH);
478 //      Player.SetY(9 * SPRITE_HEIGHT);
479 //      Player.SetDirection(0);
480 //      destroy_win(Player.objectWindow);
481
482 //      Player.objectWindow = create_win(6,9,Player.GetY(),Player.GetX());
483 //      Player.DrawSprite(Player,0);
484 //      currentLevel.ImportMap(0,0,"TEMP 4");
485 //      break;
486
487 //case -805: //TEMP 5
488 //      printf("\a");
489 //      Player.SetX(14 * SPRITE_WIDTH);
490 //      Player.SetY(3 * SPRITE_HEIGHT);
491 //      Player.SetDirection(0);
492 //      destroy_win(Player.objectWindow);
493
494 //      Player.objectWindow = create_win(6,9,Player.GetY(),Player.GetX());
495 //      Player.DrawSprite(Player,0);
496 //      currentLevel.ImportMap(0,0,"TEMP 5");
497 //      break;
498
499 //case -806: //TEMP 6
500 //      printf("\a");
501 //      Player.SetX(19 * SPRITE_WIDTH);
502 //      Player.SetY(8 * SPRITE_HEIGHT);
503 //      Player.SetDirection(0);
504 //      destroy_win(Player.objectWindow);
505
506 //      Player.objectWindow = create_win(6,9,Player.GetY(),Player.GetX());
507 //      Player.DrawSprite(Player,0);
508 //      currentLevel.ImportMap(0,0,"TEMP 6");
509 //      break;
510 // NPC's
511 case 702: //Blacksmith
512     if (hammer)
513         DrawMessage(-1,-1,40,8,"Blacksmith: That hammer will break down most cracked walls.");
514     else
```

```

515     {
516         DrawMessage(-1,-1,40,8,"Blacksmith: You there! You don't    seem to have a hammer.");
517         DrawMessage(-1,-1,40,8,"Blacksmith: There's no way you'll    make it through the tower without    one.");
518         DrawMessage(-1,-1,40,8,"Blacksmith: I'm not using this one    anymore, maybe you can put it to    good use.");
519         DrawMessage(-1,-1,40,8,"You obtained HAMMER!");
520         hammer = true;
521     }
522     break;
523 case 704: //Lantern Man
524     if (lantern)
525         DrawMessage(-1,-1,40,8,"Nervous Man: Good Luck!");
526     else
527     {
528         DrawMessage(-1,-1,40,8,"Nervous Man: Wait! You can't go in    there, it's too dark.");
529         DrawMessage(-1,-1,40,8,"Nervous Man: Here, take my lantern. I'm too scared to go in anyways.");
530         DrawMessage(-1,-1,40,8,"You obtained LANTERN!");
531         lantern = true;
532     }
533     break;
534 case 708: //Breakable Wall
535     if (hammer)
536     {
537         DrawMessage(-1,-1,40,8,"You broke the wall down with your    hammer!");
538         currentLevel.roomObject[index].SetObjIndex(0);
539         currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
540     }
541     else
542         DrawMessage(-1,-1,40,8,"This wall is cracked, you could    probably knock it down if you had    the correct tool. Maybe you should    look around town.");
543     break;
544 case 701: //King
545     DrawMessage(-1,-1,40,8,"Duke: Congratulations on making it    though the Castellum!");
546     DrawMessage(-1,-1,40,8,"Duke: I hereby present you with yourcertificate of Adventurity, you may now venture beyond the town walls.");
547     DrawMessage(-1,-1,40,8,"Duke: Go forth Adventurer! The worldawaits!");
548     DrawMessage(-1,-1,40,8,"Congratulations! You complete in ?? steps.");
549     DrawMessage(-1,-1,40,8,"Press 'I' then 'Esc' to exit.");
550     break;
551 }
552 }
553
554 void Switch(int &x, int &y, Object &player_object)
555 {
556     if (player_object.GetX()/SPRITE_WIDTH == 21 && (GetAsyncKeyState(VK_RIGHT) && 0x8000))//Right
557     {
558         roomX += 1;
559         Player.SetX(0);
560         destroy_win(Player.objectWindow);

```

```

561         Player.objectWindow = create_win(6,9,Player.GetY(),Player.GetX());
562         Player.DrawSprite(Player,0);
563         currentLevel.ImportMap(roomX,roomY,"NULL");
564     }
565
566     if (player_object.GetX()/SPRITE_WIDTH == 0 && (GetAsyncKeyState(VK_LEFT) && 0x8000))//Left
567     {
568         roomX -= 1;
569         Player.SetX(21 * SPRITE_WIDTH);
570         destroy_win(Player.objectWindow);
571         Player.objectWindow = create_win(6,9,Player.GetY(),Player.GetX());
572         Player.DrawSprite(Player,0);
573         currentLevel.ImportMap(roomX,roomY,"NULL");
574     }
575
576     if (player_object.GetY()/SPRITE_HEIGHT == 10 && (GetAsyncKeyState(VK_DOWN) && 0x8000))//Down
577     {
578         roomY += 1;
579         Player.SetY(0);
580         destroy_win(Player.objectWindow);
581         Player.objectWindow = create_win(6,9,Player.GetY(),Player.GetX());
582         Player.DrawSprite(Player,0);
583         currentLevel.ImportMap(roomX,roomY,"NULL");
584     }
585
586     if (player_object.GetY()/SPRITE_HEIGHT == 0 && (GetAsyncKeyState(VK_UP) && 0x8000))//Up
587     {
588         roomY -= 1;
589         Player.SetY(10 * SPRITE_HEIGHT);
590         destroy_win(Player.objectWindow);
591         Player.objectWindow = create_win(6,9,Player.GetY(),Player.GetX());
592         Player.DrawSprite(Player,0);
593         currentLevel.ImportMap(roomX,roomY,"NULL");
594     }
595 }

```

MazeFilter.cpp

```

1 #include "MazeFilter.h"
2 #include "CreateObject.h"
3 #include "Object.h"
4 #include "GlobalVars.h"
5 #include "DrawWindows.h"
6
7 int Index(int x, int y);//Prototype for "Index" function
8 int BlockVal(int x, int y);//Prototype for "BlockVal" function

```

```

9 int SetBlockObj(int x, int y); //Prototype for "SetBlock" function
10 int SetBlockSprite(int x, int y); //Prototype for "SetBlock" function
11
12 const int MAP_HEIGHT = 11;
13 const int MAP_WIDTH = 22;
14 const int SPRITE_HEIGHT = 6;
15 const int SPRITE_WIDTH = 9;
16
17 void MazeFilter()
18 {
19     for (int i = 0; i < MAP_HEIGHT; i++)
20     {
21         for (int j = 0; j < MAP_WIDTH; j++) //This ultimately checks all map spaces
22         {
23             if (currentLevel.roomObject[Index(j,i)].GetObjIndex() != 0) //If a block is not empty, assign it to what it should be.
24             {
25                 int ani[4] = {0,0,0,0};
26                 currentLevel.roomObject[Index(j,i)] = DefineObject(SetBlockSprite(j,i)-1,-
1,ani,create_win(SPRITE_HEIGHT,SPRITE_WIDTH,i*SPRITE_HEIGHT,j*SPRITE_WIDTH),0,0,i,j,0,SetBlockObj(j,i));
27             }
28             else if (currentLevel.roomObject[Index(j,i)].GetObjIndex() == 0)
29             {
30                 int ani[4] = {0,0,0,0};
31                 currentLevel.roomObject[Index(j,i)] = DefineObject(200,-1,ani,create_win(SPRITE_HEIGHT,SPRITE_WIDTH,i*SPRITE_HEIGHT,j*SPRITE_WIDTH),0,0,i,j,0,0);
32             }
33         }
34     }
35 }
36
37 int Index(int x, int y)
38 {
39     return (y * MAP_WIDTH) + x;
40 }
41
42 int BlockVal (int x, int y) //Prevents calling a window that doesn't exist
43 {
44     if (x >= 0 && x < 21 && y >= 0 && y < 11)
45     {
46         return (currentLevel.roomObject[Index(x,y)].GetObjIndex());
47     }
48     else return (0);
49 }
50
51 int SetBlockObj(int x, int y) //Returns the value for the 'object' parameter
52 {
53     if (BlockVal (x+1,y) != 0 && BlockVal (x,y+1) != 0 && BlockVal (x-1,y) != 0 && BlockVal (x,y-1) != 0) // -|- Left, right, up, and down

```

```

54         return (82);
55     else if (BlockVal (x+1,y) != 0 && BlockVal (x,y+1) != 0 && BlockVal (x-1,y) != 0) // -,-      Left, right, and down
56         return (83);
57     else if (BlockVal (x+1,y) != 0 && BlockVal (x-1,y) != 0 && BlockVal (x,y-1) != 0) // -'-      Left, right, and up
58         return (84);
59     else if (BlockVal (x,y+1) != 0 && BlockVal (x-1,y) != 0 && BlockVal (x,y-1) != 0) // -|      Left, up, and down
60         return (85);
61     else if (BlockVal (x+1,y) != 0 && BlockVal (x,y+1) != 0 && BlockVal (x,y-1) != 0) // |-      Right, up, and down
62         return (86);
63     else if (BlockVal (x+1,y) != 0 && BlockVal (x,y+1) != 0) // , -      Right and down
64         return (89);
65     else if (BlockVal (x-1,y) != 0 && BlockVal (x,y+1) != 0) // -,      Left and down
66         return (90);
67     else if (BlockVal (x+1,y) != 0 && BlockVal (x,y-1) != 0) // '-      Right and up
68         return (91);
69     else if (BlockVal (x-1,y) != 0 && BlockVal (x,y-1) != 0) // -'      Left and up
70         return (92);
71     else if (BlockVal (x+1,y) != 0 && BlockVal (x-1,y) != 0) // - -      Left and right
72         return (79);
73     else if (BlockVal (x,y+1) != 0 && BlockVal (x,y-1) != 0) // |      Up and down
74         return (73);
75     else if (BlockVal (x,y-1) != 0) // '      Up
76         return (76);
77     else if (BlockVal (x,y+1) != 0) // ,      Down
78         return (72);
79     else if (BlockVal (x+1,y) != 0) // -      Right
80         return (81);
81     else if (BlockVal (x-1,y) != 0) // -      Left
82         return (80);
83     else
84         return (93); //      None
85 }
86
87 int SetBlockSprite(int x, int y) //Returns the value for the 'sprite' parameter
88 {
89     if (BlockVal (x+1,y) != 0 && BlockVal (x,y+1) != 0 && BlockVal (x-1,y) != 0 && BlockVal (x,y-1) != 0) // -|-      Left, right, up, and down
90         return (165);
91     else if (BlockVal (x+1,y) != 0 && BlockVal (x,y+1) != 0 && BlockVal (x-1,y) != 0) // -,-      Left, right, and down
92         return (166);
93     else if (BlockVal (x+1,y) != 0 && BlockVal (x-1,y) != 0 && BlockVal (x,y-1) != 0) // -'-      Left, right, and up
94         return (167);
95     else if (BlockVal (x,y+1) != 0 && BlockVal (x-1,y) != 0 && BlockVal (x,y-1) != 0) // -|      Left, up, and down
96         return (168);
97     else if (BlockVal (x+1,y) != 0 && BlockVal (x,y+1) != 0 && BlockVal (x,y-1) != 0) // |-      Right, up, and down
98         return (169);
99     else if (BlockVal (x+1,y) != 0 && BlockVal (x,y+1) != 0) // , -      Right and down

```



```
100         return (173);
101     else if (BlockVal (x-1,y) != 0 && BlockVal (x,y+1) != 0) // -,      Left and down
102         return (174);
103     else if (BlockVal (x+1,y) != 0 && BlockVal (x,y-1) != 0) // '-     Right and up
104         return (175);
105     else if (BlockVal (x-1,y) != 0 && BlockVal (x,y-1) != 0) // -'      Left and up
106         return (176);
107     else if (BlockVal (x+1,y) != 0 && BlockVal (x-1,y) != 0) // - -     Left and right
108         return (162);
109     else if (BlockVal (x,y+1) != 0 && BlockVal (x,y-1) != 0) // |       Up and down
110         return (156);
111     else if (BlockVal (x,y-1) != 0) //      '           Up
112         return (159);
113     else if (BlockVal (x,y+1) != 0) //      ,           Down
114         return (155);
115     else if (BlockVal (x+1,y) != 0) //      -           Right
116         return (164);
117     else if (BlockVal (x-1,y) != 0) //      -           Left
118         return (163);
119     else
120         return (177); //      None
121 }
```

Object.cpp

```
1  //Includes
2  #include "Object.h"
3  #include "GlobalVars.h"
4  #include "DrawWindows.h"
5  #include "RoomController.h"
6  #include <curses.h>
7  #include <math.h>
8  #include <Windows.h>
9  #pragma comment (lib , "winmm.lib")
10
11 //Prototypes
12 Object DefineObject(int spr, int img, int ani[4], WINDOW *local_win, int v, int h, int y, int x, int dir, int object);
13 int distance(int x1, int y1, int x2, int y2);
14
15 //Variables
16 const int SPRITE_HEIGHT = 6;
17 const int SPRITE_WIDTH = 9;
18 const int MAP_HEIGHT = 11;
19 const int MAP_WIDTH = 22;
20
21 void Object::SetSprite(int spr, int img, int ani[4], WINDOW *local_win)
```

```

22 {           //Set the sprites
23     spriteIndex = spr;
24     imageIndex = img;
25     for (int i = 0; i < 4; i++)
26         animations[i] = ani[i];
27     objectWindow = local_win;
28 }
29
30 void Object::SetSpeed(int v, int h)
31 {           //Set the speed
32     vspeed = v;
33     hspeed = h;
34 }
35
36 void Object::SetDirection(int dir, int y, int x)
37 {           //Set position and orientation
38     direction = dir;
39     position[0] = y;
40     position[1] = x;
41 }
42
43 Object DefineObject(int spr, int img, int ani[4], WINDOW *local_win, int v, int h, int y, int x, int dir, int local_objIndex)
44 {           //Constructor
45     Object local_object;
46     int _ani[4];
47     for (int i = 0; i < 4; i++)
48         _ani[i] = ani[i];
49     local_object.SetSprite(spr, img, _ani, local_win);
50     local_object.SetSpeed(v, h);
51     local_object.SetDirection(dir, y, x);
52     local_object.SetObjIndex(local_objIndex);
53     return local_object;
54 }
55
56 Object boulderXY(int x, int y)
57 {
58     for (int i = 0; i < MAP_HEIGHT; i++)
59     {
60         for (int j = 0; j < MAP_WIDTH; j++)
61         {
62             if (currentLevel.puzzleObject[(i*MAP_WIDTH)+j].GetX() == x && currentLevel.puzzleObject[(i*MAP_WIDTH)+j].GetY() == y)
63                 return currentLevel.puzzleObject[(i*MAP_WIDTH)+j];
64         }
65     }
66     return currentLevel.puzzleObject[0];
67 }

```

```

68
69 Object boulderXYS(int x, int y, int v, int h)
70 {
71     for (int i = 0; i < MAP_HEIGHT; i++)
72     {
73         for (int j = 0; j < MAP_WIDTH; j++)
74         {
75             if (currentLevel.puzzleObject[(i*MAP_WIDTH)+j].GetX() == x && currentLevel.puzzleObject[(i*MAP_WIDTH)+j].GetY() == y)
76             {
77                 if (v == 0 && h == 0)
78                 {
79
80                     currentLevel.puzzleObject[(i*MAP_WIDTH)+j].SetDirection(0,currentLevel.puzzleObject[(i*MAP_WIDTH)+j].GetY()/SPRITE_HEIGHT,currentLevel.puzzleObject[(i*MAP_WIDTH)+j].GetX()/SPRITE_WIDTH);
81                     currentLevel.puzzleObject[(i*MAP_WIDTH)+j].SetSpeed(0,0);
82                 }
83                 else
84                 {
85                     currentLevel.puzzleObject[(i*MAP_WIDTH)+j].SetDirection(0,currentLevel.puzzleObject[(i*MAP_WIDTH)+j].GetY()*SPRITE_HEIGHT,currentLevel.puzzleObject[(i*MAP_WIDTH)+j].GetX()*SPRITE_WIDTH);
86                     currentLevel.puzzleObject[(i*MAP_WIDTH)+j].SetSpeed(v,h);
87                 }
88             }
89         }
90     }
91     return currentLevel.puzzleObject[0];
92 }
93
94 int Object::DrawSprite(Object local_object, int animationIndex)
95 {
96     start_color();
97     init_pair(1, COLOR_WHITE, COLOR_BLACK);
98     init_pair(2, COLOR_BLACK, COLOR_BLACK);
99     init_pair(3, COLOR_YELLOW, COLOR_BLACK);
100
101     int x1, x2, y1, y2;
102
103     wborder(local_object.objectWindow, ' ', ' ', ' ', ' ', ' ', ' ', ' ', ' ', ' ', ' ');
104
105     int k = local_object.spriteIndex + local_object.animations[(animationIndex + local_object.imageIndex)%4];
106     if (local_object.GetImageIndex() == -1) // Don't animate
107         k = local_object.spriteIndex;
108
109     if (local_object.GetImageIndex() == -2) // Don't draw
110         return 0;
111
112     k += local_object.direction * 3;

```

```

112
113     x2 = Player.position[1]/SPRITE_WIDTH;
114     y2 = Player.position[0]/SPRITE_HEIGHT;
115
116     if (!currentLevel.AreLightsOn())
117     {
118         x1 = local_object.position[1];
119         y1 = local_object.position[0];
120
121         if (distance(x1, y1, x2, y2) > 31)
122             return 0;
123         if (distance(x1, y1, x2, y2) == 30 || distance(x1, y1, x2, y2) == 31 || distance(x1, y1, x2, y2) == 28 )
124         {
125             for (int i=0; i < SPRITE_HEIGHT; i++)
126             {
127                 for (int j=0; j < SPRITE_WIDTH; j++)
128                 {
129                     mvwaddch(local_object.objectWindow,i,j,ASCII_ART[k][i][j] | COLOR_PAIR(2));
130                 }
131             }
132             wnoutrefresh(local_object.objectWindow);
133             return 1;
134         }
135     }
136
137     for (int i=0; i < SPRITE_HEIGHT; i++)
138     {
139         for (int j=0; j < SPRITE_WIDTH; j++)
140         {
141             if (local_object.GetObjIndex() == 1000)
142             {
143                 attron(COLOR_PAIR(3));
144                 mvwprintw(local_object.objectWindow,1,1,"Score\n%d",Step);
145                 attroff(COLOR_PAIR(3));
146                 wnoutrefresh(local_object.objectWindow);
147             }
148             else
149                 mvwaddch(local_object.objectWindow,i,j,ASCII_ART[k][i][j] | COLOR_PAIR(1));
150         }
151     }
152
153     wnoutrefresh(local_object.objectWindow);
154     return 1;
155 }
156
157 int Object::DrawLayerSprite(Object local_object, int animationIndex)

```

```

158 {
159     wborder(local_object.objectWindow, ' ', ' ', ' ', ' ', ' ', ' ', ' ', ' ', ' ');
160
161     int index, x, y, k = local_object.spriteIndex + local_object.animations[(animationIndex + local_object.imageIndex)%4];
162
163     if (local_object.GetImageIndex() == -1) // Don't animate
164         k = local_object.spriteIndex;
165     if (local_object.GetImageIndex() == -2) // Don't Draw
166         return 0;
167
168     k += local_object.direction * 3;
169
170     for (int i=0; i < SPRITE_HEIGHT; i++)
171     {
172         for (int j=0; j < SPRITE_WIDTH; j++)
173         {
174             x = local_object.position[1] + j;
175             y = local_object.position[0] + i;
176
177             index = ((y-(y%SPRITE_HEIGHT))/SPRITE_HEIGHT);
178             index = index * MAP_WIDTH;
179             index += ((x-(x%SPRITE_WIDTH))/SPRITE_WIDTH);
180
181             clear();
182
183             if (ASCII_ART[k][i][j] != 'g')
184                 mvwaddch(local_object.objectWindow,i,j,ASCII_ART[k][i][j]);
185             else
186                 mvwaddch(local_object.objectWindow,i,j,mvwinch(currentLevel.roomObject[index].objectWindow, y%SPRITE_HEIGHT, x%SPRITE_WIDTH));
187         }
188     }
189     wnoutrefresh(local_object.objectWindow);
190     return 1;
191 }
192
193 int distance(int x1, int y1, int x2, int y2)
194 {
195     int distance = 0;
196     double formula = ((x1-x2)*(x1-x2))+((y1-y2)*(y1-y2));
197
198     formula = sqrt(formula);
199
200     distance = int(floor(formula * 10));
201     return distance;
202 }

```

.h PlayerMovement.cpp

```
1  //Includes
2  #include "Object.h"
3  #include "DrawWindows.h"
4  #include "GlobalVars.h"
5  #include "Interactions.h"
6  #include <Windows.h>
7
8  //Prototypes
9  int PlayerMove(Object &boulder);
10 void MoveToSnap(Object &local_object);
11 bool CheckFree(int dir, Object &local_object, int d);
12 void Move(int dir, Object &local_object);
13 void Door(Object &boulder);
14 int BoulderMove(Object &boulder);
15
16 //Variables
17 const int MAP_WIDTH = 22;
18 const int MAP_HEIGHT = 11;
19 const int SPRITE_WIDTH = 9;
20 const int SPRITE_HEIGHT = 6;
21
22 int PlayerMove(Object &player_object)
23 {
24     Switch(roomX, roomY, player_object);
25     if (EnterWasPressed())
26         Door(player_object);
27
28     if (player_object.GetY()%SPRITE_HEIGHT == 0 && player_object.GetX()%SPRITE_WIDTH == 0)
29     {
30         player_object.SetImageIndex(-1);
31         Move(0, player_object); //All Directions
32     }
33     else if (player_object.GetY()%SPRITE_HEIGHT == 0)
34         Move(1, player_object); //Horizontal Only
35     else if (player_object.GetX()%SPRITE_WIDTH == 0)
36         Move(2, player_object); //Vertical Only
37
38     MoveToSnap(player_object);
39     return 0;
40 }
41
42 void MoveToSnap(Object &local_object)
```

```

43 {
44     if (local_object.GetHSpeed() != 0 || local_object.GetVSpeed() != 0)
45     {
46         destroy_win(local_object.objectWindow);
47         local_object.SetY(local_object.GetY()+local_object.GetVSpeed());
48         local_object.SetX(local_object.GetX()+local_object.GetHSpeed());
49         local_object.SetDirection(local_object.GetDirection(),local_object.GetY(),local_object.GetX());
50         local_object.objectWindow = create_win(6,9,local_object.GetY(),local_object.GetX());
51     }
52 }
53
54 bool CheckFree(int dir, Object &local_object, int d)
55 {
56     int i = local_object.GetY()/SPRITE_HEIGHT;
57     int j = local_object.GetX()/SPRITE_WIDTH;
58     int index, indexF;
59
60     switch (dir)
61     {
62
63     case 0: // Down
64         i += 1;
65         index = (i*MAP_WIDTH)+j;
66         i += 1;
67         indexF = (i*MAP_WIDTH)+j;
68
69         if (boulderXY(j,i-1).GetObjIndex() == 45 && (currentLevel.roomObject[indexF].GetObjIndex() <= 0 || currentLevel.roomObject[indexF].GetObjIndex() == 55) &&
boulderXY(j,i).GetObjIndex() != 45 && d == 0)
70         {
71             boulderXYS(j,i-1,1,0);
72             return false;
73         }
74         else if (boulderXY(j,i-1).GetObjIndex() == 45 && (currentLevel.roomObject[indexF].GetObjIndex() > 0 || boulderXY(j,i).GetObjIndex() == 45) && d == 0)
75             return false;
76         else if (currentLevel.roomObject[index].GetObjIndex() <= 0)
77             return true;
78         break;
79
80     case 1: // Up
81         i -= 1;
82         index = (i*MAP_WIDTH)+j;
83         i -= 1;
84         indexF = (i*MAP_WIDTH)+j;
85
86         if (boulderXY(j,i+1).GetObjIndex() == 45 && (currentLevel.roomObject[indexF].GetObjIndex() <= 0 || currentLevel.roomObject[indexF].GetObjIndex() == 55) &&

```

```

    boulderXY(j,i).GetObjIndex() != 45 && d == 0)
87         {
88             boulderXYS(j,i+1,-1,0);
89             return false;
90         }
91     else if (boulderXY(j,i+1).GetObjIndex() == 45 && (currentLevel.roomObject[indexF].GetObjIndex() > 0 || boulderXY(j,i).GetObjIndex() == 45) && d == 0)
92         return false;
93     else if (currentLevel.roomObject[index].GetObjIndex() <= 0)
94         return true;
95     break;
96
97     case 2: // Right
98         j += 1;
99         index = (i*MAP_WIDTH)+j;
100        j += 1;
101        indexF = (i*MAP_WIDTH)+j;
102
103        if (boulderXY(j-1,i).GetObjIndex() == 45 && (currentLevel.roomObject[indexF].GetObjIndex() <= 0 || currentLevel.roomObject[indexF].GetObjIndex() == 55) &&
boulderXY(j,i).GetObjIndex() != 45 && d == 0)
104            {
105                boulderXYS(j-1,i,0,1);
106                return false;
107            }
108        else if (boulderXY(j-1,i).GetObjIndex() == 45 && (currentLevel.roomObject[indexF].GetObjIndex() > 0 || boulderXY(j,i).GetObjIndex() == 45) && d == 0)
109            return false;
110        else if (currentLevel.roomObject[index].GetObjIndex() <= 0)
111            return true;
112        break;
113
114    case 3: // Left
115        j -= 1;
116        index = (i*MAP_WIDTH)+j;
117        j -= 1;
118        indexF = (i*MAP_WIDTH)+j;
119
120        if (boulderXY(j+1,i).GetObjIndex() == 45 && (currentLevel.roomObject[indexF].GetObjIndex() <= 0 || currentLevel.roomObject[indexF].GetObjIndex() == 55) &&
boulderXY(j,i).GetObjIndex() != 45 && d == 0)
121            {
122                boulderXYS(j+1,i,0,-1);
123                return false;
124            }
125        else if (boulderXY(j+1,i).GetObjIndex() == 45 && (currentLevel.roomObject[indexF].GetObjIndex() > 0 || boulderXY(j,i).GetObjIndex() == 45) && d == 0)
126            return false;
127        else if (currentLevel.roomObject[index].GetObjIndex() <= 0)
128            return true;

```



```
129         break;
130
131     }
132
133     return false;
134 }
135
136 void Move(int dir, Object &local_object)
137 {
138     if (currentLevel.roomObject[((local_object.GetY()/SPRITE_HEIGHT) * MAP_WIDTH) + (local_object.GetX()/SPRITE_WIDTH)].GetObjIndex() == -15)
139         if (local_object.GetY()%SPRITE_HEIGHT == 0 && local_object.GetX()%SPRITE_WIDTH == 0)
140         {
141             if (local_object.GetVSpeed() == 1 && !CheckFree(0,local_object, dir))
142             {
143                 local_object.SetSpeed(0,0);
144                 if (currentLevel.roomObject[0].GetObjIndex() == 999)
145                     Step ++;
146             }
147             if (local_object.GetVSpeed() == -1 && !CheckFree(1,local_object, dir))
148             {
149                 local_object.SetSpeed(0,0);
150                 if (currentLevel.roomObject[0].GetObjIndex() == 999)
151                     Step ++;
152             }
153             if (local_object.GetHSpeed() == 1 && !CheckFree(2,local_object, dir))
154             {
155                 local_object.SetSpeed(0,0);
156                 if (currentLevel.roomObject[0].GetObjIndex() == 999)
157                     Step ++;
158             }
159             if (local_object.GetHSpeed() == -1 && !CheckFree(3,local_object, dir))
160             {
161                 local_object.SetSpeed(0,0);
162                 if (currentLevel.roomObject[0].GetObjIndex() == 999)
163                     Step ++;
164             }
165         }
166
167     if (dir == 0 && currentLevel.roomObject[((local_object.GetY()/SPRITE_HEIGHT) * MAP_WIDTH) + (local_object.GetX()/SPRITE_WIDTH)].GetObjIndex() != -15)
168     {
169         if (local_object.GetHSpeed() != 0 || local_object.GetVSpeed() != 0)
170             if (currentLevel.roomObject[0].GetObjIndex() == 999)
171                 Step ++;
172         local_object.SetSpeed(0,0);
173     }
```

```

174
175     if (dir == 0 || dir == 1)
176     {
177         if (GetAsyncKeyState(VK_RIGHT) && 0x8000)
178         {
179             local_object.SetDirection(2);
180             if (CheckFree(2, local_object, dir))
181             {
182                 if (currentLevel.roomObject[((local_object.GetY()/SPRITE_HEIGHT) * MAP_WIDTH) + (local_object.GetX()/SPRITE_WIDTH)].GetObjIndex() != -15 ||
183                     (currentLevel.roomObject[((local_object.GetY()/SPRITE_HEIGHT) * MAP_WIDTH) + (local_object.GetX()/SPRITE_WIDTH)].GetObjIndex() == -15 &&
184                         local_object.GetHSpeed() == 0 && local_object.GetVSpeed() == 0))
185                     local_object.SetHSpeed(1);
186                 local_object.SetDirection(2);
187                 local_object.SetImageIndex(0);
188             }
189         }
190
191         if (GetAsyncKeyState(VK_LEFT) && 0x8000)
192         {
193             local_object.SetDirection(3);
194             if (CheckFree(3, local_object, dir))
195             {
196                 if (currentLevel.roomObject[((local_object.GetY()/SPRITE_HEIGHT) * MAP_WIDTH) + (local_object.GetX()/SPRITE_WIDTH)].GetObjIndex() != -15 ||
197                     (currentLevel.roomObject[((local_object.GetY()/SPRITE_HEIGHT) * MAP_WIDTH) + (local_object.GetX()/SPRITE_WIDTH)].GetObjIndex() == -15 &&
198                         local_object.GetHSpeed() == 0 && local_object.GetVSpeed() == 0))
199                     local_object.SetHSpeed(-1);
200                 local_object.SetDirection(3);
201                 local_object.SetImageIndex(0);
202             }
203         }
204     }
205
206     if (dir == 0 || dir == 2)
207     {
208         if (GetAsyncKeyState(VK_DOWN) && 0x8000 && local_object.GetHSpeed() == 0)
209         {
210             local_object.SetDirection(0);
211             if (CheckFree(0, local_object, dir))
212             {
213                 if (currentLevel.roomObject[((local_object.GetY()/SPRITE_HEIGHT) * MAP_WIDTH) + (local_object.GetX()/SPRITE_WIDTH)].GetObjIndex() != -15 ||
214                     (currentLevel.roomObject[((local_object.GetY()/SPRITE_HEIGHT) * MAP_WIDTH) + (local_object.GetX()/SPRITE_WIDTH)].GetObjIndex() == -15 &&
215                         local_object.GetHSpeed() == 0 && local_object.GetVSpeed() == 0))
216                     local_object.SetVSpeed(1);
217                 local_object.SetDirection(0);
218                 local_object.SetImageIndex(0);

```

```

219         }
220     }
221
222     if (GetAsyncKeyState(VK_UP) && 0x8000 && local_object.GetHSpeed() == 0)
223     {
224         local_object.SetDirection(1);
225         if (CheckFree(1, local_object, dir))
226         {
227             if (currentLevel.roomObject[((local_object.GetY()/SPRITE_HEIGHT) * MAP_WIDTH) + (local_object.GetX()/SPRITE_WIDTH)].GetObjIndex() != -15 ||
228                 (currentLevel.roomObject[((local_object.GetY()/SPRITE_HEIGHT) * MAP_WIDTH) + (local_object.GetX()/SPRITE_WIDTH)].GetObjIndex() == -15 &&
229                     local_object.GetHSpeed() == 0 && local_object.GetVSpeed() == 0))
230                 local_object.SetVSpeed(-1);
231             local_object.SetDirection(1);
232             local_object.SetImageIndex(0);
233         }
234     }
235 }
236 }
237
238 void Door(Object &player_object)
239 {
240     int i,j, index;
241
242     i = player_object.GetX()/SPRITE_WIDTH;
243     j = player_object.GetY()/SPRITE_HEIGHT;
244
245     switch (player_object.GetDirection())
246     {
247     case 0: //Down
248         j += 1;
249         break;
250     case 1: //Up
251         j -= 1;
252         break;
253     case 2: //Right
254         i += 1;
255         break;
256     case 3: //Left
257         i -= 1;
258         break;
259     }
260
261     index = (j*MAP_WIDTH) + i;
262
263     Interaction(currentLevel.roomObject[index].GetObjIndex(), index);

```

```

264 }
265
266 int BoulderMove(Object &boulder)
267 {
268     switch (boulder.GetVSpeed())
269     {
270     case -1:
271         boulder.SetDirection(1);
272         break;
273     case 0:
274         switch (boulder.GetHSpeed())
275         {
276         case -1:
277             boulder.SetDirection(3);
278             break;
279         case 0:
280             break;
281         case 1:
282             boulder.SetDirection(2);
283             break;
284         }
285         break;
286     case 1:
287         boulder.SetDirection(0);
288         break;
289     }
290
291     if (boulder.GetHSpeed() != 0 || boulder.GetVSpeed() != 0)
292     {
293         MoveToSnap(boulder);
294         if (boulder.GetX()%SPRITE_WIDTH == 0 && boulder.GetY()%SPRITE_HEIGHT == 0)
295         {
296             printf("\a");
297             if (currentLevel.roomObject[((boulder.GetY()/SPRITE_HEIGHT)*MAP_WIDTH)+(boulder.GetX()/SPRITE_WIDTH)].GetObjIndex() == 55)
298             {
299                 int ani[4] = { 0, 0, 1, 1};
300                 currentLevel.roomObject[((boulder.GetY()/SPRITE_HEIGHT)*MAP_WIDTH)+(boulder.GetX()/SPRITE_WIDTH)] =
DefineObject(83,1,ani,create_win(SPRITE_HEIGHT,SPRITE_WIDTH,boulder.GetY(),boulder.GetX()),0,0,boulder.GetY()/SPRITE_HEIGHT,boulder.GetX()/ 9,0,-16);
301
302                 currentLevel.roomObject[((boulder.GetY()/SPRITE_HEIGHT)*MAP_WIDTH)+(boulder.GetX()/SPRITE_WIDTH)].DrawSprite(currentLevel.roomObject[((boulder.GetY()/SPRITE_HEIGHT)*MAP_WIDTH)+(boulder.Get
X()/SPRITE_WIDTH)],0);
303
304                 boulder.SetObjIndex(0);
305                 boulder.SetImageIndex(-2);
306             }
307             else if ((currentLevel.roomObject[((boulder.GetY()/SPRITE_HEIGHT)*MAP_WIDTH)+(boulder.GetX()/SPRITE_WIDTH)].GetObjIndex() == -15 &&

```

```

    !CheckFree(boulder.GetDirection(),boulder,0)) ||
307         currentLevel.roomObject[ ((boulder.GetY()/SPRITE_HEIGHT)*MAP_WIDTH)+(boulder.GetX()/SPRITE_WIDTH)].GetObjIndex() != -15)
308     {
309         boulder.SetSpeed(0,0);
310         boulder.SetX(boulder.GetX()/SPRITE_WIDTH);
311         boulder.SetY(boulder.GetY()/SPRITE_HEIGHT);
312     }
313 }
314 }
315     boulder.SetDirection(0);
316     boulder.SetImageIndex(-1);
317     return 0;
318 }
1
```

RoomController.cpp

```

1 //Includes
2 #include "Object.h"
3 #include "DrawWindows.h"
4 #include "PlayerMovement.h"
5 #include "GlobalVars.h"
6 #include <curses.h>
7 #include <string>
8 #include <Windows.h>
9 #include <fstream>
10
11 //Prototypes
12 void BoulderFunctions(int index);
13
14 //Variables
15 const int MAP_HEIGHT = 11;
16 const int MAP_WIDTH = 22;
17 const int SPRITE_HEIGHT = 6;
18 const int SPRITE_WIDTH = 9;
19 int updatedSprite = 0, k, l;
20 char ASCII_ART[400][6][9];
21
22 int RoomUpdate()//Code that updates all of the objects
23 {
24     Sleep(30);
25     int index;
26
27     PlayerMove(Player);
28
29     for (int i=0; i < MAP_HEIGHT; i++)
```

```

30     {
31         //Nested for loops to check all 242 objects
32         for (int j=0; j < MAP_WIDTH; j++)
33         {
34             index = (i*MAP_WIDTH)+j;
35             currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],updatedSprite); //Draw each of their sprites animated
36         }
37     }
38     if (Player.GetHSpeed() != 0 || Player.GetVSpeed() != 0)
39         Player.DrawLayerSprite(Player,updatedSprite); //Draw the player sprite animated
40     else
41         Player.DrawLayerSprite(Player,0); //Draw the player sprite animated
42
43     for (int i=0; i < MAP_HEIGHT; i++)
44     {
45         //Nested for loops to check all 242 objects
46         for (int j=0; j < MAP_WIDTH; j++)
47         {
48             index = (i*MAP_WIDTH)+j;
49             if (currentLevel.puzzleObject[index].GetObjIndex() == 45)
50             {
51                 BoulderMove(currentLevel.puzzleObject[index]);
52                 currentLevel.puzzleObject[index].DrawLayerSprite(currentLevel.puzzleObject[index],0); //Draw each of their sprites animated
53             }
54         }
55     }
56     if (currentLevel.roomObject[0].GetObjIndex() == 999)
57     {
58         for (int i = 0; i < SPRITE_HEIGHT; i++)
59             for (int j = 0; j < SPRITE_WIDTH; j++)
60                 mvwaddch(currentLevel.roomObject[0].objectWindow,i,j,' ');
61         attron(COLOR_PAIR(3));
62         mvwprintw(currentLevel.roomObject[0].objectWindow,2,2,"Score");
63         mvwprintw(currentLevel.roomObject[0].objectWindow,3,2,"%d",Step);
64         attroff(COLOR_PAIR(3));
65         wborder(currentLevel.roomObject[0].objectWindow, '|', '|', '-', '-', '*', '*', '*', '*');
66
67         wnoutrefresh(currentLevel.roomObject[0].objectWindow);
68     }
69
70     doupdate();
71
72     if (updatedSprite == 3)
73         updatedSprite = -1;
74     updatedSprite++;
75

```

```

76     if (InventoryWasPressed())
77     {
78         return DrawInventory("Inventory", -1, 1);
79     }
80     return 1;
81 }
82
83 int ImportAsciiArt(std::string file)
84 {
85     std::string word;
86     std::ifstream fileName;
87     fileName.open(file);
88     fileName >> word;
89
90     if (fileName.is_open())
91     {
92         while (fileName.good())
93         {
94             for (int i=0; i< 400; i++)
95             {
96                 for (int j=0; j < 6; j++)
97                 {
98                     for (int k=0; k < 9; k++)
99                     {
100                         if (word[k] == 'G')
101                             ASCII_ART[i][j][k] = ' '; //Blank if it is a G
102                         else
103                             ASCII_ART[i][j][k] = word[k]; //Ascii_art[Sprite][Y][X]
104                     }
105                     fileName >> word;
106                 }
107             }
108         }
109         fileName.close();
110     }
111     else
112     {
113         printf("The Ascii Art the computer is trying to load doesn't exist\n"); //Apologize
114         printf("Try re-installing the game\n");
115         getch();
116         return 0;
117     }
118     return 1;
119 }

```

Headers

CreateObject.h

```
1 #ifndef CREATE_OBJECT
2 #define CREATE_OBJECT
3     void DefineObjects(int object, int y, int x);
4 #endif
```

DrawWindows.h

```
1 #include "Object.h"
2
3 #ifndef MENU
4 #define MENU
5     int MainMenu();
6     int DrawBackground(char* file, int doGet);
7     WINDOW *create_win(int height, int width, int starty, int startx);
8     void destroy_win(WINDOW *local_win);
9     void DrawMessage(int y, int x, int w, int h, char* message);
10    int DrawInventory(char* title, int item, int price);
11    extern Object Player;
12 #endif
```

GlobalVars.h

```
1 #include "ImportMaps.h"
2 //These variables can be accessed throughout all cpp files
3 #ifndef GLOBAL_VARS
4 #define GLOBAL_VARS
5     extern Level currentLevel;
6     extern int gameState;
7     extern int roomX;
8     extern int roomY;
9     extern int Step;
10    extern char* global_item[4];
11    extern int money;
12    extern int save;
13
14    bool EnterWasPressed();
15    bool InventoryWasPressed();
16 #endif
```

ImportMaps.h


```
1 #include <string>
2 #include <Windows.h>
3 #include "Object.h"
4
5 #ifndef IMPORT_MAPS
6 #define IMPORT_MAPS
7
8 class Level
9 {
10 private:
11     bool lights;
12     int id;
13
14 public:
15     Level(); //Default Constructor
16
17     Object roomObject[242];
18     Object puzzleObject[242];
19
20     bool AreLightsOn() { return lights; }
21     int GetLevelID() { return id; }
22     int ImportMap(int x, int y, std::string mapName);
23     void ToggleLights();
24     void GenerateMaze();
25 };
26
27 #endif
```

Interactions.h

```
1 #include "Object.h"
2
3 #ifndef INTERACTIONS
4 #define INTERACTIONS
5
6     void Interaction(int obj, int index);
7     void Switch(int &x, int &y, Object &player_object);
8     extern bool lantern;
9     extern int progress;
10    extern int totalfloors;
11 #endif
```

MazeFilter.h

```
1 #ifndef MAZE_FILTER
2 #define MAZE_FILTER
3     void MazeFilter();
```

4 #endif

Object.h

```
1  #include <curses.h>
2  #include <Windows.h>
3
4  #ifndef OBJECT
5  #define OBJECT
6
7  class Object
8  {
9  private:
10     int spriteIndex;
11         int objIndex;
12     int imageIndex;
13     int hspeed;
14     int vspeed;
15     int direction;
16         //          for direction    1
17         //                                     3    2
18         //                                     0
19     int position[2]; //x y coordinates
20 public:
21     WINDOW *objectWindow;
22     int animations[4];
23
24     void SetSpriteIndex(int s){ spriteIndex = s; }
25     void SetImageIndex(int i){ imageIndex = i; }
26     void SetDirection(int d){ direction = d; }
27     void SetObjIndex(int o){ objIndex = o; }
28     void SetHSpeed(int h){ hspeed = h; }
29     void SetVSpeed(int v){ vspeed = v; }
30     void SetX(int x){ position[1] = x; }
31     void SetY(int y){ position[0] = y; }
32
33     int GetSpriteIndex(){ return spriteIndex; }
34     int GetImageIndex(){ return imageIndex; }
35     int GetDirection(){ return direction; }
36     int GetObjIndex(){ return objIndex; }
37     int GetVSpeed(){ return vspeed; }
38     int GetHSpeed(){ return hspeed; }
39     int GetX(){ return position[1]; }
40     int GetY(){ return position[0]; }
41
42     //Functions for the Object Class
```

```
43     void SetSprite(int spr, int img, int ani[4], WINDOW *local_win);
44     void SetSpeed(int v, int h);
45     void SetDirection(int dir, int y, int x);
46     int DrawSprite(Object local_object, int animationIndex);
47     int DrawLayerSprite(Object local_object, int animationIndex);
48 };
49
50     int distance(int x1, int y1, int x2, int y2);
51     Object boulderXY(int x, int y);
52     Object boulderXYS(int x, int y, int v, int h);
53
54     Object DefineObject(int spr, int img, int ani[4], WINDOW *local_win, int v, int h, int y, int x, int dir, int object);
55
56 #endif
```

PlayerMovement.h

```
1 #include "Object.h"
2
3 #ifndef PLAYER_MOVEMENT
4 #define PLAYER_MOVEMENT
5     int PlayerMove(Object &player_object);
6     int BoulderMove(Object &boulder);
7 #endif
```

RoomController.h

```
1 #include <string>
2
3 #ifndef ROOM_CONTROLLER
4 #define ROOM_CONTROLLER
5     int ImportAsciiArt(std::string file);
6     int RoomUpdate();
7
8     extern char ASCII_ART[400][6][9];
9 #endif
```