# Table of contents

# C++ files

## MainGameLoop.cpp

```cpp
/* This is the Main Game loop. The code is repeated while loopGame is true. The switch
Statement checks to see the state of the game (if it is in menu, in a fight, or in the overworld)
and calls the corresponding function that is called each frame. */

// Includes
#include <stdio.h>
#include <Windows.h>
#include <curses.h>
#include "DrawWindows.h"
#include "ImportMaps.h"
#include "GlobalVars.h"
#include "RoomController.h"
#include "Interactions.h"

// Prototypes
void ResizeWindow(int _x, int _y);

// Variables
int loopGame = 1, gameState = -1, save = 1;
int roomX = 1, roomY = 1, Step = 0, money = 0;
bool enter = false, inv = false;

char* global_item[4] = { "Bomb", "Lantern", "Hammer", "mk II Lantern" };

Level currentLevel;

using namespace std;

void ResizeWindow(int _x, int _y) //Take the new sizes parameters
{
         resize_term(68,200);//The Lines and Columns needed
         HWND console = GetConsoleWindow();
         RECT r;
         GetWindowRect(console, &r); //stores the console's current dimensions
         MoveWindow(console, r.left, r.top, _x, _y, TRUE);
         ShowWindow( console, SW_MINIMIZE); //Minimize first, incase it is already maximized
         ShowWindow( console, SW_MAXIMIZE); //because maximizing twice messes it up
}

```

```c
int main()
{
	initscr(); //Start Public Domain Curses

	if(has_colors() == FALSE) //Cekck if colors are supported
	{
		printf("Your terminal does not support color\n"); //Apologize
		printf("Closing Game, sorry\n");
		getch();
		loopGame = 0; //Never start the loop or initialize colors, if they arent supported
	}

	loopGame = ImportAsciiArt("Assets\\SpriteSheet.txt");
	noecho();
	curs_set(0); //Invisible cursor
	keypad(stdscr, TRUE); //Additional input
	nonl();

	ResizeWindow(1600,816); //This resizes the window first

	while(loopGame) //Loop while true
	{
		start_color(); //Start color

		init_pair(1, COLOR_BLACK, COLOR_WHITE); //WHITE
		init_pair(2, COLOR_WHITE, COLOR_BLACK); //Hightlighted Text

		switch (gameState)
		{
		case -1:
			loopGame = DrawBackground("Assets\\TowerBackground.txt", 1);
			break;

		case 0: //Main Menu
			loopGame = DrawBackground("Assets\\TowerBackground.txt", 0);
			loopGame = MainMenu();
			break;

		case 1: //Overworld
			loopGame = RoomUpdate();
			break;

		}
	}
	endwin();
	return 0; //End the game
```

```cpp
}

bool EnterWasPressed()
{
	if (enter == false && (GetAsyncKeyState(VK_RETURN) && 0x8000))
	{
		enter = true;
		return true;
	}
	else if (enter == true && (!GetAsyncKeyState(VK_RETURN) && 0x8000))
	{
		enter = false;
		return false;
	}
	return false;
}

bool InventoryWasPressed()
{
	if (inv == false && (GetAsyncKeyState('I') && 0x8000))
	{
		inv = true;
		return true;
	}
	else if (inv == true && (!GetAsyncKeyState('I') && 0x8000))
	{
		inv = false;
		return false;
	}
	return false;
}
```

CreateObject.cpp

```cpp
//Includes
#include "Object.h"
#include "DrawWindows.h"
#include "GlobalVars.h"
#include "Interactions.h"

//Prototypes
void SetAnimations(int a, int b, int c, int d);
void DefineObjects(int object, int y, int x);

//Variables
const int SPRITE_HEIGHT = 6;
const int SPRITE_WIDTH = 9;
int ani[4];

void SetAnimations( int a, int b, int c, int d)
{
	ani[0] = a;
	ani[1] = b;
	ani[2] = c;
	ani[3] = d;
}

void DefineObjects(int object, int y, int x)
{
	int index = (y*22)+x;
	SetAnimations(0,1,2,1);

	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);
	currentLevel.puzzleObject[index].DrawSprite(currentLevel.puzzleObject[index],0);

	switch (object)
	{
	 case -16:
		SetAnimations(0,0,1,1);
		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);
		currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
		break;
	case -15: //Ice
		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);
		currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
		break;
	case -14: //Cobblestone 4
		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);
		currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
		break;
	case -13: //Cobblestone 3
		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);
		currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
		break;
	case -12: //Cobblestone 2
		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);
		currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
		break;
	case -11: //Cobblestone 1
		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);
		currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
		break;
```

```
        case -10: //Bridge Left
                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);
                currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                break;
        case -9:  //Bridge Right
                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);
                currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                break;
        case -8:  //Bridge Middle
                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);
                currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                break;
        case -7:  //Bridge Horizontal Top
                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);
                currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                break;
        case -6:  //Bridge Horizontal Middle
                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);
                currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                break;
        case -5:  //Bridge Horizontal Bottom
                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);
                currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                break;
        case -4:  //Ladder Right
                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);
                currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                break;
        case -3:  //Ladder Left
                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);
                currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                break;
        case -2:  //Ladder Vertical
                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);
                currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                break;
        case -1:  //Blank Space
                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);
                currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                break;
        case 0:           //Blank Space
                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);
                currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                break;
        case 1:           //Chest Unlocked
                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);
```

```
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
          case 2:             //Chest Locked
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
          case 3:             //Chest Opened
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
          case 4:             //Horizontal Table Left
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
          case 5:             //Horizontal Table Middle
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
          case 6:             //Horizontal Table Right
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
          case 7:             //Vertical Table Top
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
          case 8:             //Vertical Table Middle
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
          case 9:             //Vertical Table Bottom
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
          case 10:  //Empty Chair
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
          case 11:  //Woman Seated Facing Left
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
          case 12:  //Woman Seated Facing Right
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
```

```
        case 13:  //Man Seated Facing Up
                  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);
                  currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                  break;
        case 14:  //Man Seated Facing Left
                  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);
                  currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                  break;
        case 15:  //Man Seated Facing Right
                  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);
                  currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                  break;
        case 16:  //Vertical Table on a Brick Wall
                  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);
                  currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                  break;
        case 17:  //Brick Wall
                  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);
                  currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                  break;
        case 18:  //Window on a Brick Wall
                  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);
                  currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                  break;
        case 19:  //Door Closed
                  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);
                  currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                  break;
        case 20:  //Door Open
                  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);
                  currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                  break;
        case 21:  //Cave Door
                  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);
                  currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                  break;
        case 22:  //Fence Going Left
                  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);
                  currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                  break;
        case 23:  //Fence Going Up
                  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);
                  currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                  break;
        case 24:  //Fence Going Right
                  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);
```

```
                currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                break;
        case 25:  //Top of a Shaded Wall
                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);
                currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                break;
        case 26:  //Brick Wall Right
                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);
                currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                break;
        case 27:  //Brick Wall Left
                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);
                currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                break;
        case 28:  //Brick Wall Bottom
                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);
                currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                break;
        case 29:  //Brick top left inside corner
                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);
                currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                break;
        case 30:  //Brick top right inside corner
                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);
                currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                break;
        case 31:  //Brick bottom left inside corner
                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);
                currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                break;
        case 32:  //Brick bottom right inside corner
                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);
                currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                break;
        case 33:  //Brick top left outside corner
                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);
                currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                break;
        case 34:  //Brick top right outside corner
                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);
                currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                break;
        case 35:  //Brick bottom left outside corner
                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);
                currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                break;
        case 36:  //Brick bottom right outside corner
                  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);
                  currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                  break;
        case 37:  //Brick Wall Door Facing Right
                  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);
                  currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                  break;
        case 38:  //Brick Wall Door Facing Left
                  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);
                  currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                  break;
        case 39:  //Brick Wall Door Facing Up
                  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);
                  currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                  break;
        case 40:  //Sign
                  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);
                  currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                  break;
        case 41:  //Bed Top
                  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);
                  currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                  break;
        case 42:  //Bed Bottom
                  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);
                  currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                  break;
        case 43:  //Tree
                  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);
                  currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                  break;
        case 44:  //Bush
                  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);
                  currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                  break;
        case 45:  //Pushing Block
                  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);
                  currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                  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);
                  currentLevel.puzzleObject[index].DrawSprite(currentLevel.puzzleObject[index],0);
                  break;
    case 46:      //Church middle top
                  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);
                  currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                  break;
        case 47:  //Church middle empty
                  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);
                  currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                  break;
        case 48:  //Church middle cross
                  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);
                  currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                  break;
        case 49:  //Church middle door
                  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);
                  currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                  break;
        case 50:  //Church left top
                  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);
                  currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                  break;
        case 51:  //Church left middle
                  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);
                  currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                  break;
        case 52:  //Church right top
                  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);
                  currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                  break;
        case 53:  //Church right middle
                  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);
                  currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                  break;
        case 54:  //Church side bottom
                  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);
                  currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                  break;
        case 55: //Water Pattern 1
                  SetAnimations(0,0,1,1);
                  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);
                  currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                  break;
        /*case 56:          //Water Corner Top Left
                  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);
                  currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                  break;
        case 57:  //Water Corner Top Right
                  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);
                  currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                  break;
        case 58:  //Water Corner Bottom Left
```

```
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
            case 59:  //Water Corner Bottom Right
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
            case 60:  //VOID - CHANGE TO HOLE
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;*/
            case 61:  //Brick Wall Full
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
            case 62:  //Brick Window Full
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
            case 63:  //Brick Door Full
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
            case 64:  //Mod Wall?
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
            case 65:  //Bookshelf
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
            case 66:  //Cliff face
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
            case 67:  //Cliff Left Angle Bottom
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
            case 68:  //Cliff Left Angle Top
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
            case 69:  //Cliff Right Angle Bottom
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
        case 70:  //Cliff Right Angle Top
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
        case 71:  //Cliff Right Side
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
        case 72:  //Barn 1
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
        case 73:  //Barn 2
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
        case 74:  //Barn 3
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
        case 75:  //Barn 4
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
        case 76:  //Barn 5
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
        case 77:  //Barn 6
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
        case 78:  //Barn 7
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
        case 79:  //Barn 8
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
        case 80:  //Barn 9
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
        case 81:  //Barn 10
```

```
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
            case 82:  //Barn 11
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
            case 83:  //Barn 12
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
            case 84:  //Barn 13
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
            case 85:  //Barn 14
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
            case 86:  //Barn 15
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
            case 87:  //Horse 1
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
            case 88:  //Horse 2
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
            case 89:  //Wood Wall top
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
            case 90:  //Wood Wall        left
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
            case 91:  //Wood Wall right
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
            case 92:  //Wood Wall bottom
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
        case 93:  //Wood corner 1
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
        case 94:  //Wood corner 2
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
        case 95:  //Wood corner 3
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
        case 96:  //Wood corner 4
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
        case 97:  //Dog
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
        case 101: //Smashable Block
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
        case 102: //Decorative Boulder
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
        case 103: //Well
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
        case 104: //Campfire
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
        case 105: //Cliff Left Side
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
        case 106: //Cliff Right Side
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
        case 107: //Cliff Back Left Corner
```

```
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
            case 108: //Cliff Back Right Corner
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
            case 109: //Cliff Back Border
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
            case 110: //Cliff Side Height
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
            case 111: //Ladder Up
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
            case 112: //Bridge Right
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
            /*case 113:        //Bridge Middle
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;*/
            case 114: //Indoor Cliff Top Left
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
            case 115: //Indoor Cliff Left
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
            case 116: //Indoor Cliff Bottom Left
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
            case 117: //Indoor Cliff Top Right
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
            case 118: //Indoor Cliff Right
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
        case 119: //Indoor Cliff Bottom Right
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
        case 120: //Indoor Cliff Ladder Right
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
        case 121: //Indoor Cliff Ladder Left
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
        /*case 122:         //Roof Chimney
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;*/
        case 123: //Statue 1
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
        case 124: //Statue 2
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
        case 125: //Statue 3
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
        case 126: //Statue 4
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
        case 127: //Statue 5
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
        case 128: //Statue 6
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
        case 129: //Anvil
                    SetAnimations(0,0,1,1);
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
        case 130: //House
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
        case 131: //House
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
        case 132: //House
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
        case 133: //House
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
        case 134: //House
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
        case 135: //House
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
        case 136: //House
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
        case 137: //House
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
        case 138: //House
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
        case 139: //House
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
        case 140: //House
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
        case 141: //House
                    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);
```

```
                        currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                        break;
                case 142: //Alchemist Logo
                        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);
                        currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                        break;
                case 143: //Survivalist Logo
                        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);
                        currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                        break;
                case 144: //Locksmith Logo
                        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);
                        currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                        break;
                case 145: //Bomb Crafter Logo
                        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);
                        currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                        break;
                case 146: //Odd Corner Brick
                        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);
                        currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                        break;
                case 147: //Odd Corner Brick
                        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);
                        currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                        break;
                case 148: //Odd Corner Brick
                        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);
                        currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                        break;
                case 149: //Odd Corner Brick
                        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);
                        currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                        break;
                case 150: //Odd Corner Brick
                        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);
                        currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                        break;
                case 151: //Odd Corner Brick
                        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);
                        currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                        break;
                case 152: //Odd Corner Brick
                        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);
                        currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                        break;
```

```
        case 153: //Odd Corner Brick
                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);
                currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                break;
        case 154: //Table Corner 1
                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);
                currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                break;
        case 155: //Table Corner 2
                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);
                currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                break;
        case 156: //TEST SPRITE 1
                SetAnimations(0,1,2,3);
                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);
                currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                break;
        case 157: //TEST SPRITE 2
                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);
                currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                break;
        case 158: //TEST SPRITE 3
                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);
                currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                break;
        case 159: //TEST SPRITE 4
                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);
                currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                break;
//Chairs

    case 98:        //Seat facing left
                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);
                currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                break;
    case 99:        //Seat facing right
                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);
                currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                break;
        case 100: //Seat facing up
                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);
                currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                break;

        //Interactable Doors (301-400)

        case 301: //D1
                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);
                currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                break;
        case 302: //D2
                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);
                currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                break;
        case 303: //D3
                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);
                currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                break;
        case 304: //D4
                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);
                currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                break;
        case 305: //D5
                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);
                currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                break;
        case 306: //D6
                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);
                currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                break;
        case 307: //D7
                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);
                currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                break;
        case 308: //D8
                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);
                currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                break;
        case 309: //D9
                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);
                currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                break;
        case 310: //D10
                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);
                currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                break;
        case 311: //Alchemist IN
                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);
                currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                break;
        case 312: //Alchemist OUT
                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);
```

```
			currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
			break;
		case 313: //Survivalist IN
			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);
			currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
			break;
		case 314: //Survivalist OUT
			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);
			currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
			break;
		case 315: //Locksmith IN
			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);
			currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
			break;
		case 316: //Locksmith OUT
			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);
			currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
			break;
		case 317: //Bombshop IN
			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);
			currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
			break;
		case 318: //Bombshop OUT
			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);
			currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
			break;

/*
		//Chests (501-600)

		case 501: //Chest 1: Locked chest in playerhousebedroom1
			mapGrid[(x-2)/SPRT_WIDTH][(y-2)/SPRT_HEIGHT] = 501;
			if (chestsOpened[0] == 1){
				DrawSprite(8,2,x,y,false,false,0);
			}
			else DrawSprite(8,1,x,y,false,false,0);
			break;
		case 502: //Chest 2: Unlocked chest in playerhousebedroom3
			mapGrid[(x-2)/SPRT_WIDTH][(y-2)/SPRT_HEIGHT] = 502;
			if (chestsOpened[1] == 1){
				DrawSprite(8,2,x,y,false,false,0);
			}
			else DrawSprite(8,0,x,y,false,false,0);
			break;
		case 503: //Chest 3: Unlocked chest in 0,-1
			mapGrid[(x-2)/SPRT_WIDTH][(y-2)/SPRT_HEIGHT] = 503;
		if (chestsOpened[2] == 1){
			DrawSprite(8,2,x,y,false,false,0);
		}
		else DrawSprite(8,0,x,y,false,false,0);
		break;
	case 504: //Chest 4: Unlocked chest in 0,-1
		mapGrid[(x-2)/SPRT_WIDTH][(y-2)/SPRT_HEIGHT] = 504;
		if (chestsOpened[3] == 1){
			DrawSprite(8,2,x,y,false,false,0);
		}
		else DrawSprite(8,0,x,y,false,false,0);
		break;
	case 505:
		mapGrid[(x-2)/SPRT_WIDTH][(y-2)/SPRT_HEIGHT] = 505;
		if (chestsOpened[4] == 1){
			DrawSprite(8,2,x,y,false,false,0);
		}
		else DrawSprite(8,0,x,y,false,false,0);
		break;
	case 506:
		mapGrid[(x-2)/SPRT_WIDTH][(y-2)/SPRT_HEIGHT] = 506;
		if (chestsOpened[5] == 1){
			DrawSprite(8,2,x,y,false,false,0);
		}
		else DrawSprite(8,0,x,y,false,false,0);
		break;
*/
	//Signs (601-700)

	case 601: //Sign 1: Your house
		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);
		currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
		break;
	case 602: //Sign 2: Castletown ahead
		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);
		currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
		break;

	//NPCs (701 - 1000)

	case 701: //NPC 1: King/Duke
		currentLevel.roomObject[index] = DefineObject(124,-1,ani,create_win(SPRITE_HEIGHT,SPRITE_WIDTH,y*SPRITE_HEIGHT,x*SPRITE_WIDTH),0,0,y,x,0,object);
		currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
		break;
	case 702: //NPC 2: Blacksmith
		SetAnimations(0,0,1,1);
```

```
                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);
                currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                break;
        case 703:  //NPC 3: Conspicuous Stranger
                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);
                currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                break;
        case 704: //Lantern Man
                if (lantern)
                {
                DefineObjects(105, y, x);
                }
                else
                {
                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);
                currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                }
                break;
        case 706: //Bonfires
                SetAnimations(0,0,1,1);
                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);
                currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                break;
        case 708: //Breakable Wall
                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);
                currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                break;
// Ladders UP
        case 800: //Ladder 0
                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);
                currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                break;
        case 801: //Ladder 1
                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);
                currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                break;
        case 802: //Ladder 2
                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);
                currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                break;
        case 803: //Ladder 3
                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);
                currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                break;
        case 804: //Ladder 4
                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);
```

```
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
          case 805: //Ladder 5
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
          case 806: //Ladder 6
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
          case 807: //Ladder 7
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
          case 808: //Ladder 8
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
          case 809: //Ladder 9
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
          case 810: //Ladder 10
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
          case 811: //Ladder 11
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
          case 812: //Ladder 12
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;

// Ladders DOWN
          case 813: //Ladder top
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
          case -801:          //Ladder 1
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
          case -802:          //Ladder 2
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
          case -803:          //Ladder 3
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
          case -804:          //Ladder 4
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
          case -805:          //Ladder 5
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
          case -806:          //Ladder 6
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
          case -807:          //Ladder 7
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
          case -808:          //Ladder 8
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
          case -809:          //Ladder 9
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
          case -810:          //Ladder 10
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
          case -811:          //Ladder 11
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
          case 999: //Score
                    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);
                    currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    break;
          }
}
```
DrawWindows.cpp

```cpp
//Includes
#include <curses.h>
#include <string.h>
#include <Windows.h>
#include <fstream>
#include <string>
#include "GlobalVars.h"
#include "ImportMaps.h"
#include "Object.h"
#include "Interactions.h"

//Prototypes
WINDOW *create_win(int height, int width, int starty, int startx);
void DrawMenuOptions(WINDOW *local_win);
void destroy_win(WINDOW *local_win);
void PrintInMiddle(WINDOW *window, int y, int width, char* string, int color);
int do_selection(int sel, int men, WINDOW *local_win);
void SetAnimations( int a, int b, int c, int d);
void DrawMessage(int y, int x, int w, int h, char* message);

Object Player;

//Variables
WINDOW *my_win;
const int SPRITE_HEIGHT = 6;
const int SPRITE_WIDTH = 9;
int playerx, playery, ac_playery, ac_playerx;
char* word;

int selection = 0, ch = 0, whichMenu = 0;
int player_ani[4] = {0,1,2,1};

char* string;
//Options for the menus stored in arrays. If a menu is longer than the 4 options, a back
//"more" option should be made available
char* MAIN_MENU[] = { " Start New Game ",
         " Continue Saved Game ",
         " Maze Race ",
         " Exit Game " };
char* START_NEW_GAME[] = { " Overwrite Save 1 ",
         " Overwrite Save 2 ",
         " Overwrite Save 3 ",
         " Back " };
char* CONTINUE[] = { " Continue Save 1 ",
         " Continue Save 2 ",
         " Continue Save 3 ",
        " Back " };

int MainMenu()
{
        int row, col;
        getmaxyx(stdscr,row,col); //Get the max rows and columns

        my_win = create_win(11,40,(row/2)-15,(col/2)-20); //Create a window
        wborder(my_win, '|', '|', '-', '-', '*', '*', '*', '*');

        DrawMenuOptions(my_win); //Draw the options

        ch = getch();

        //Test for which key has been pressed
        switch(ch)
        {
        case KEY_UP:
                selection --;
                break;
        case KEY_DOWN:
                selection ++;
                break;
        case 13: //because KEY_ENTER returns \n
                return do_selection((selection % 4), whichMenu, my_win);
                break;
        case 27: //Escape
                gameState = -1;
                destroy_win(my_win);
        default:
                break;
        }

        return 1;
}

WINDOW *create_win(int height, int width, int starty, int startx)
{ //This creates a window of specified height and width at the y and x coordinates
        WINDOW *local_win;
        local_win = newwin(height, width, starty, startx);
        box(local_win, 0 , 0);
        //custom border for the window
        wborder(local_win, ' ', ' ', ' ',' ',' ',' ',' ',' ');
        wnoutrefresh(local_win);
        return local_win;
}
```

```c

void destroy_win(WINDOW *local_win)
{ //Destroy the window function
	wborder(local_win, ' ', ' ', ' ',' ',' ',' ',' ',' ');
	werase(local_win);
	wnoutrefresh(local_win);
	delwin(local_win);
}

void DrawMenuOptions(WINDOW *local_win)
{
	if (selection == 0) //This is needed because for some reason,
	{//modulos doesnt work with negative numbers that aren't of the multiple you are testing..
		selection = 4;
	}

	for (int i=0; i < 4; i++) //Check which menu to draw, and draw all 4 options
	{
		if ((selection % 4) == i) //Selected
		{
			switch(whichMenu)
			{
			case 0: // Main Menu
				PrintInMiddle(my_win, 2 + (i*2), 40, MAIN_MENU[i], 1);
				break;

			case 1: // NewGame Menu
				PrintInMiddle(my_win, 2 + (i*2), 40, START_NEW_GAME[i], 1);
				break;

			case 2: // ContinueGame Menu
				PrintInMiddle(my_win, 2 + (i*2), 40, CONTINUE[i], 1);
				break;

			}
		}
		else //Not selected
		{
			switch(whichMenu)
			{
			case 0: // Main Menu
				PrintInMiddle(my_win, 2 + (i*2), 40, MAIN_MENU[i], 2);
				break;

			case 1: // NewGame Menu
				PrintInMiddle(my_win, 2 + (i*2), 40, START_NEW_GAME[i], 2);
```

```c
                                        break;

                                case 2: // ContinueGame Menu
                                        PrintInMiddle(my_win, 2 + (i*2), 40, CONTINUE[i], 2);
                                        break;

                                }
                        }
          }
          wrefresh(local_win); //Refresh the window
}

void PrintInMiddle(WINDOW *local_win, int y, int width, char* string, int color)
{// A simple function to print in the given color, in the middle of the given window (stdscr is no window)
          wattron(local_win, COLOR_PAIR(color));

          int middleX = (width/2)-strlen(string)/2;

          mvwprintw(local_win,y,middleX, "%s", string);

          wattroff(local_win, COLOR_PAIR(color));
          wrefresh(local_win); //Refresh the window
}

int do_selection(int sel, int men, WINDOW *local_win) //Do the action needed for the menu
{
          switch(men)
          {
          case 0: //Main Menu MENU
                    switch(sel)
                    {
                    case 0://Start New Game
                              printf("\a");
                              whichMenu = 1;
                              selection = 0;
                              break;

                    case 1://Continue Saved Game
                              printf("\a");
                              whichMenu = 2;
                              selection = 0;
                              break;

                    case 2://MazeRace
                              printf("\a");
                              gameState = 1;
```

```
                            destroy_win(local_win);
                            //Create the player
                            playerx = 1 * SPRITE_WIDTH;
                            playery = 1 * SPRITE_HEIGHT;
                            Player = DefineObject(13,-1,player_ani,create_win(SPRITE_HEIGHT,SPRITE_WIDTH,playery,playerx),0,0,playery,playerx,0,0);
                            //Draw the Map
                            currentLevel.GenerateMaze();
                            break;

                    case 3://Exit Game
                            printf("\a");
                            return 0; //This value is sent all the way back to the MainGameLoop.cpp
                            break; //and means the game loop is to be exited, and the game is quit

                    }
                    break;

            case 1: //Start New Game MENU
                    switch(sel)
                    {
                    case 0://Overwrite save 1
                            save = 1;
                            printf("\a");
                            gameState = 1;
                            destroy_win(local_win);
                            //Create the player
                            playerx = 10 * SPRITE_WIDTH;
                            playery = 4 * SPRITE_HEIGHT;
                            Player = DefineObject(13,-1,player_ani,create_win(SPRITE_HEIGHT,SPRITE_WIDTH,playery,playerx),0,0,playery,playerx,0,0);
                            //Draw the Map
                            return currentLevel.ImportMap(roomX,roomY,"playerhousebedroom1");
                            break;

                    case 1://Overwrite save 2
                            save = 2;
                            printf("\a");
                            gameState = 1;
                            destroy_win(local_win);
                            //Create the player
                            playerx = 10 * SPRITE_WIDTH;
                            playery = 4 * SPRITE_HEIGHT;
                            Player = DefineObject(13,-1,player_ani,create_win(SPRITE_HEIGHT,SPRITE_WIDTH,playery,playerx),0,0,playery,playerx,0,0);
                            //Draw the Map
                            return currentLevel.ImportMap(roomX,roomY,"playerhousebedroom1");
                            break;

                    case 2://Overwrite save 3
                            save = 3;
                            printf("\a");
                            gameState = 1;
                            destroy_win(local_win);
                            //Create the player
                            playerx = 10 * SPRITE_WIDTH;
                            playery = 4 * SPRITE_HEIGHT;
                            Player = DefineObject(13,-1,player_ani,create_win(SPRITE_HEIGHT,SPRITE_WIDTH,playery,playerx),0,0,playery,playerx,0,0);
                            //Draw the Map
                            return currentLevel.ImportMap(roomX,roomY,"playerhousebedroom1");
                            break;

                    case 3://Back
                            printf("\a");
                            whichMenu = 0;
                            selection = 0;
                            break;

                    }
                    break;
         case 2: //Continue Saved Game MENU
                    switch(sel)
                    {
                    case 0://Continue Save 1

                            break;

                    case 1://Continue Save 2
                            printf("\a");
                            break;

                    case 2://Continue Save 3
                            printf("\a");
                            break;

                    case 3://Back
                            printf("\a");
                            whichMenu = 0;
                            selection = 1;
                            break;

                    }
                    break;

         }
```

```
		return 1;
}

int DrawBackground(char* file, int doGet)
{
		std::string word;
		std::ifstream fileName;
		fileName.open(file); //Open the File
		fileName >> word;

		if (fileName.is_open())
		{
				while (fileName.good())
				{
						for (int i = 0; i < 66; i ++)//66 is the # of rows
						{
								for (int j = 0; j < 199; j ++)//199 is the # of columns
								{
										if (word[j] != 'G')
												mvwaddch(stdscr,i,j,word[j]);
								}
								fileName >> word;
						}
						fileName.close(); //Close the file
				}
		}
		else
		{
				printf("The File the computer is trying to load doesn't exist\n"); //Apologize
				printf("Try re-installing the game\n");
				return 0;
		}

		wnoutrefresh(stdscr);
		doupdate();

		if (doGet)
		{
				getch();
				gameState = 0;
		}
		return 1;
}

void DrawMessage(int y, int x, int w, int h, char* message)
{
```

```c
	int row, col;
	getmaxyx(stdscr,row,col); //Get the max rows and columns
	int _x = x, _y = y;

	if (x == -1)
		_x = (col/2)-(w/2);
	if (y == -1)
		_y = (row/3)-(h/2);

	WINDOW *shadow_win;
	WINDOW *local_win;

	shadow_win = create_win(h+2,w+2,_y-1,_x-1);
	local_win = create_win(h,w,_y,_x);

	printf("\a");

	for (int i=0; i<h; i++)
	{
		for (int j=0; j<(w-4); j++)
		{
			if (i*(w-4)+j < strlen(message))
				mvwaddch(local_win,i+1,j+2,message[(i*(w-4))+j]);
		}
	}

	wborder(local_win, '|', '|', '-', '-', '*', '*', '*', '*');

	wrefresh(shadow_win);
	wrefresh(local_win);

	while (!EnterWasPressed())
		local_win = local_win;

	destroy_win(local_win);
}

int DrawInventory(char* title, int item, int price)
{
	keypad(stdscr, TRUE);

	WINDOW *shadow_win;
	WINDOW *local_win;

	shadow_win = create_win(42,122,1,1);
	local_win = create_win(40,120,2,2);
	printf("\a");

	mvwprintw(local_win,3,4,title);

	mvwprintw(local_win,6,4,"Press <ESC> to exit the game");
	mvwprintw(local_win,8,4,"Press <F1> to save your game");
	mvwprintw(local_win,10,4,"Press <F2> to return to the Main Menu");


	mvwprintw(local_win,14,4,"Helpful Information:"); // Help information
	mvwprintw(local_win,16,4,"Arrow Keys to move, and Enter to interact with objects"); // Help information
	mvwprintw(local_win,18,4,"[] : Walk into blocks to push them."); // Help information
	mvwprintw(local_win,19,4,"// : Walk on ice to slide."); // Help information
	if (item != -1)
		mvwprintw(local_win,12,4,"Press <Enter> to buy a %s for %d gold",global_item[item],price);

	mvwprintw(local_win,3,60,"Gold: %d", money);
	mvwprintw(local_win,3,80,"Save %d", save);
	mvwprintw(local_win,3,100,"Floor %d of %d",progress,totalfloors);

	wborder(local_win, '|', '|', '-', '-', '*', '*', '*', '*');
	nonl();

	int ch = getch();
	wrefresh(shadow_win);
	wrefresh(local_win);

	while (!InventoryWasPressed())
	{
		//Test for which key has been pressed
		switch(ch)
		{
		case KEY_F(1):
			//Damian, you need to add the save game here
			printf("\a");
			break;

		case KEY_F(2):
			gameState = 0;
			printf("\a");
			return 1;
			break;

		case 27: //Escape
			return 0;
```

```
                                break;

                        default:
                                break;
                        }

                        ch = getch();
                        wrefresh(shadow_win);
                        wrefresh(local_win);
                }
                destroy_win(local_win);

                return 1;
}
```

## ImportMaps.cpp

```
//Includes
#include <curses.h>
#include <fstream>
#include <sstream>
#include <stdlib.h>
#include <Windows.h>
#include <stdio.h>
#include <time.h>
#include "ImportMaps.h"
#include "CreateObject.h"
#include "DrawWindows.h"
#include "GlobalVars.h"
#include "MazeFilter.h"

//Prototypes
std::string int_to_string(int number);
int CurrentCell(char xory, int SmallIndex);
bool Visited(int x, int y);
int SmallIndex(int x, int y);

//Variables
const int MAP_WIDTH = 22;
const int MAP_HEIGHT = 11;

//MazeGeneration
bool visited[10][5]; //each cell
int currentCell;
int visitedCells;
int randomNum;
```

```cpp
int randomCells[4];
int cX, cY, toErase;

int Level::ImportMap(int x, int y, std::string mapName)
{
	std::string file;
	int word;

	if (mapName == "NULL") //Concatenation process
	{
		std::string result = "Maps\\" + int_to_string(x) + "," + int_to_string(y) + ".txt";
		//Concatate the file location.
		file = result;
		id = 1;
	}
	else
	{
		std::string result = "Maps\\" + mapName + ".txt";
		//Concatate the file location
		file = result;
		id = 0;
	}

	std::ifstream fileName;
	fileName.open(file); //Open the File
	if (fileName.is_open())
	{
		while (fileName.good())
		{
			for (int i=0; i < MAP_HEIGHT; i++)
			{
				for (int j=0; j < MAP_WIDTH; j++)
				{
					fileName >> word;
					DefineObjects(word,i,j);
				}
			}
			fileName.close(); //Close the file
		}
	}
	else
	{
		printf("The Map File the computer is trying to load doesn't exist\n"); //Apologize
		printf("Try re-installing the game\n");
		getch();
		return 0; //Close game if can't import file
```

```
        }
        return 1;
}

void Level::ToggleLights()
{
        lights = !lights;
}

std::string int_to_string (int number)
{
        std::stringstream ss;
        ss << number;
        std::string newString = ss.str();
        return newString;
}

 /*     From Wikipedia - Recursive Backtrack method

        Make the initial cell the current cell and mark it as visited
   While there are unvisited cells
       If the current cell has any neighbours which have not been visited
           Choose randomly one of the unvisited neighbours
           Push the current cell to the stack
           Remove the wall between the current cell and the chosen cell
           Make the chosen cell the current cell and mark it as visited
       Else if stack is not empty
           Pop a cell from the stack
           Make it the current cell
       Else
           Pick a random cell, make it the current cell and mark it as visited */

void Level::GenerateMaze()
{
        for (int i = 0; i < 10; i++)
                for (int j = 0; j < 5; j++)  //everything will be a wall (1)
                        visited[j][i] = false;
        visitedCells = 1;

        //make a blank maze of all walls
        for (int i = 0; i < 11; i++)
                for (int j = 0; j < 22; j++) //everything will be a wall (1)
                {
                        if (j != 21)
                                currentLevel.roomObject[(i*22)+j].SetObjIndex(1);
                        else
                currentLevel.roomObject[(i*22)+j].SetObjIndex(0);
        }

    //"Make the initial cell the current cell and mark it as visited"
    visited[0][0] = true;
    currentCell = 0;
    currentLevel.roomObject[23].SetObjIndex(0);

    srand (static_cast<unsigned int>(time(NULL))); //new random number

    //"While there are unvisited cells"
    while (visitedCells != 50)
    {
        cX = CurrentCell('x', currentCell);
        cY = CurrentCell('y', currentCell);

        //"If the current cell has any neighbours which have not been visited"
        if (!(Visited(cX,cY-1) == true && Visited(cX,cY+1) == true && Visited(cX+1,cY) == true && Visited(cX-1,cY) == true))
        {
            randomNum = 0;
            //"Choose randomly one of the unvisited neighbours"
            if (Visited(cX,cY-1) == false) // Above
            {
                randomCells[randomNum] = 0;
                randomNum ++;
            }
            if (Visited(cX,cY+1) == false) // Below
            {
                randomCells[randomNum] = 1;
                randomNum ++;
            }
            if (Visited(cX+1,cY) == false) // Right
            {
                randomCells[randomNum] = 2;
                randomNum ++;
            }
            if (Visited(cX-1,cY) == false) // Left
            {
                randomCells[randomNum] = 3;
                randomNum ++;
            }

            randomNum = rand() % randomNum;
            //"Push the current cell to the stack"
            //"Remove the wall between the current cell and the chosen cell"

				switch (randomCells[randomNum])
				{
				case 0:
					currentCell = SmallIndex(cX,cY-1);
					cY --;
					toErase = (((cY*2)+2)*22)+(cX*2)+1;
					currentLevel.roomObject[toErase].SetObjIndex(0);
					break;
				case 1:
					currentCell = SmallIndex(cX,cY+1);
					cY ++;
					toErase = ((cY*2)*22)+(cX*2)+1;
					currentLevel.roomObject[toErase].SetObjIndex(0);
					break;
				case 2:
					currentCell = SmallIndex(cX+1,cY);
					cX ++;
					toErase = (((cY*2)+1)*22)+(cX*2);
					currentLevel.roomObject[toErase].SetObjIndex(0);
					break;
				case 3:
					currentCell = SmallIndex(cX-1,cY);
					cX --;
					toErase = (((cY*2)+1)*22)+(cX*2)+2;
					currentLevel.roomObject[toErase].SetObjIndex(0);
					break;
				}
				//"Make the chosen cell the current cell and mark it as visited"
				visited[CurrentCell('x', currentCell)][CurrentCell('y', currentCell)] = true;

				toErase = (((cY*2)+1)*22)+((cX*2))+1;
				currentLevel.roomObject[toErase].SetObjIndex(0);
				visitedCells ++;
			}
			//"Else if stack is not empty"
			else
			{
				for(int i = 0; i < 10; i ++)
				{
					for(int j = 0; j < 5; j ++)
					{
						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))
						{
							currentCell = SmallIndex(i,j);
						}
					}
```

```
                                }
                        }
                }
                MazeFilter();
}

int CurrentCell(char xory, int index)
{
                if (xory == 'x')
                {
                        return index % MAP_WIDTH;
                }
                else if (xory == 'y')
                {
                        return (index - (index % MAP_WIDTH)) / 10;
                }
                return 0;
}

bool Visited (int x, int y)
{
                if (x >= 0 && x <= 9 && y >= 0 && y <= 4)
                        return visited[x][y];
                return true;
}

int SmallIndex(int x, int y)
{
                return (y * MAP_WIDTH) + x;
}

Level::Level()
{
                lights = true;
}
```

## Interactions.cpp

```
//Includes
#include "Interactions.h"
#include "ImportMaps.h"
#include "DrawWindows.h"
#include "GlobalVars.h"
#include "CreateObject.h"
#include <Windows.h>
//#include "Inventory.h"
```

```

//Variables
const int SPRITE_WIDTH = 9;
const int SPRITE_HEIGHT = 6;
bool lantern = false;
bool hammer = false;
int finalstep;
int progress = 0;
int totalfloors = 100;

void Interaction(int obj, int index)
{

	switch(obj)
	{
	case 301: //Player Bedroom 1
		printf("\a");
		Player.SetX(14 * SPRITE_WIDTH);
		Player.SetY(6 * SPRITE_HEIGHT);
		Player.SetDirection(2);
		destroy_win(Player.objectWindow);

		Player.objectWindow = create_win(6,9,Player.GetY(),Player.GetX());
		Player.DrawSprite(Player,0);
		currentLevel.ImportMap(0,0,"playerhouseentrance");
		break;

	case 302: //Player Entrance
		printf("\a");
		Player.SetX(12 * SPRITE_WIDTH);
		Player.SetY(5 * SPRITE_HEIGHT);
		Player.SetDirection(3);
		destroy_win(Player.objectWindow);

		Player.objectWindow = create_win(6,9,Player.GetY(),Player.GetX());
		Player.DrawSprite(Player,0);
		currentLevel.ImportMap(0,0,"playerhousebedroom1");
		break;

	case 303: //Player Bedroom 3
		printf("\a");
		Player.SetX(18 * SPRITE_WIDTH);
		Player.SetY(5 * SPRITE_HEIGHT);
		Player.SetDirection(3);
		destroy_win(Player.objectWindow);

```

```
                    Player.objectWindow = create_win(6,9,Player.GetY(),Player.GetX());
                    Player.DrawSprite(Player,0);
                    currentLevel.ImportMap(0,0,"playerhouseentrance");
                    break;

          case 304: //Player Entrance
                    printf("\a");
                    Player.SetX(8 * SPRITE_WIDTH);
                    Player.SetY(4 * SPRITE_HEIGHT);
                    Player.SetDirection(2);
                    destroy_win(Player.objectWindow);

                    Player.objectWindow = create_win(6,9,Player.GetY(),Player.GetX());
                    Player.DrawSprite(Player,0);
                    currentLevel.ImportMap(0,0,"playerhousebedroom3");
                    break;

          case 305: //Player Entrance
                    printf("\a");
                    Player.SetX(7 * SPRITE_WIDTH);
                    Player.SetY(5 * SPRITE_HEIGHT);
                    Player.SetDirection(0);
                    destroy_win(Player.objectWindow);

                    Player.objectWindow = create_win(6,9,Player.GetY(),Player.GetX());
                    Player.DrawSprite(Player,0);
                    roomX = 2;
                    roomY = 1;
                    currentLevel.ImportMap(2,1,"NULL");
                    break;

          case 306: //2,1 to Player House
                    printf("\a");
                    Player.SetX(16 * SPRITE_WIDTH);
                    Player.SetY(7 * SPRITE_HEIGHT);
                    Player.SetDirection(1);
                    destroy_win(Player.objectWindow);

                    Player.objectWindow = create_win(6,9,Player.GetY(),Player.GetX());
                    Player.DrawSprite(Player,0);
                    currentLevel.ImportMap(0,0,"playerhouseentrance");
                    break;

          case 307: //Player Entrance
                    printf("\a");
                    Player.SetX(10 * SPRITE_WIDTH);
```

```
                Player.SetY(6 * SPRITE_HEIGHT);
                Player.SetDirection(1);
                destroy_win(Player.objectWindow);

                Player.objectWindow = create_win(6,9,Player.GetY(),Player.GetX());
                Player.DrawSprite(Player,0);
                currentLevel.ImportMap(0,0,"playerhousebedroom2");
                break;

        case 308: //Player Bedroom 2
                printf("\a");
                Player.SetX(11 * SPRITE_WIDTH);
                Player.SetY(2 * SPRITE_HEIGHT);
                Player.SetDirection(0);
                destroy_win(Player.objectWindow);

                Player.objectWindow = create_win(6,9,Player.GetY(),Player.GetX());
                Player.DrawSprite(Player,0);
                currentLevel.ImportMap(0,0,"playerhouseentrance");
                break;

        case 309: //Cave Entrance
                if (lantern)
                {
                printf("\a");
                currentLevel.ToggleLights();
                Player.SetX(8 * SPRITE_WIDTH);
                Player.SetY(9 * SPRITE_HEIGHT);
                Player.SetDirection(1);
                destroy_win(Player.objectWindow);

                Player.objectWindow = create_win(6,9,Player.GetY(),Player.GetX());
                Player.DrawSprite(Player,0);
                currentLevel.ImportMap(2,1,"FLOOR 0");
                }
                else
                {
                Player.SetDirection(3);
                Player.DrawSprite(Player,0);
                DrawMessage(-1,-1,40,8,"Nervous Man: Wait! You can't go in  there, it's too dark.");
                DrawMessage(-1,-1,40,8,"Nervous Man: Here, take my lantern. I'm too scared to go in anyways.");
                DrawMessage(-1,-1,40,8,"You obtained LANTERN!");
                lantern = true;
                }
                break;

      case 310: //Cave Exit
                printf("\a");
                currentLevel.ToggleLights();
                Player.SetX(10 * SPRITE_WIDTH);
                Player.SetY(3 * SPRITE_HEIGHT);
                Player.SetDirection(0);
                destroy_win(Player.objectWindow);

                Player.objectWindow = create_win(6,9,Player.GetY(),Player.GetX());
                Player.DrawSprite(Player,0);
                currentLevel.ImportMap(1,1,"NULL");
                break;

      case 311: //Alchemist IN
                DrawMessage(-1,-1,40,8,"The Alchemist seems to have spilled something corrosive on the door knob...");
                //printf("\a");
                //Player.SetX(9 * SPRITE_WIDTH);
                //Player.SetY(6 * SPRITE_HEIGHT);
                //Player.SetDirection(1);
                //destroy_win(Player.objectWindow);

                //Player.objectWindow = create_win(6,9,Player.GetY(),Player.GetX());
                //Player.DrawSprite(Player,0);
                //currentLevel.ImportMap(1,1,"Alchemist");
                break;

      case 312: //Alchemist OUT
                printf("\a");
                Player.SetX(18 * SPRITE_WIDTH);
                Player.SetY(3 * SPRITE_HEIGHT);
                Player.SetDirection(0);
                destroy_win(Player.objectWindow);

                Player.objectWindow = create_win(6,9,Player.GetY(),Player.GetX());
                Player.DrawSprite(Player,0);
                currentLevel.ImportMap(2,1,"NULL");
                break;

      case 313: //Survivalist IN
                DrawMessage(-1,-1,40,8,"There's a sign on the door saying   'Out adventuring'...");
                //printf("\a");
                //Player.SetX(9 * SPRITE_WIDTH);
                //Player.SetY(6 * SPRITE_HEIGHT);
                //Player.SetDirection(1);
                //destroy_win(Player.objectWindow);

```

```
                //Player.objectWindow = create_win(6,9,Player.GetY(),Player.GetX());
                //Player.DrawSprite(Player,0);
                //currentLevel.ImportMap(1,1,"Survivalist");
                break;

        case 314: //Survivalist OUT
                //printf("\a");
                //Player.SetX(15 * SPRITE_WIDTH);
                //Player.SetY(8 * SPRITE_HEIGHT);
                //Player.SetDirection(0);
                //destroy_win(Player.objectWindow);

                //Player.objectWindow = create_win(6,9,Player.GetY(),Player.GetX());
                //Player.DrawSprite(Player,0);
                //currentLevel.ImportMap(2,1,"NULL");
                break;

        case 315: //Locksmith IN
                DrawMessage(-1,-1,40,8,"The Locksmith seems to have locked  himself in...");
                //printf("\a");
                //Player.SetX(9 * SPRITE_WIDTH);
                //Player.SetY(6 * SPRITE_HEIGHT);
                //Player.SetDirection(1);
                //destroy_win(Player.objectWindow);

                //Player.objectWindow = create_win(6,9,Player.GetY(),Player.GetX());
                //Player.DrawSprite(Player,0);
                //currentLevel.ImportMap(1,1,"Locksmith");
                break;

        case 316: //Locksmith OUT
                printf("\a");
                Player.SetX(12 * SPRITE_WIDTH);
                Player.SetY(3 * SPRITE_HEIGHT);
                Player.SetDirection(0);
                destroy_win(Player.objectWindow);

                Player.objectWindow = create_win(6,9,Player.GetY(),Player.GetX());
                Player.DrawSprite(Player,0);
                currentLevel.ImportMap(2,1,"NULL");
                break;

        case 317: //Bombcrafter IN
                DrawMessage(-1,-1,40,8,"Nobody seem's to be in...");
                /*printf("\a");
                Player.SetX(9 * SPRITE_WIDTH);
                Player.SetY(6 * SPRITE_HEIGHT);
                Player.SetDirection(1);
                destroy_win(Player.objectWindow);
                Player.objectWindow = create_win(6,9,Player.GetY(),Player.GetX());
                Player.DrawSprite(Player,0);
                currentLevel.ImportMap(1,1,"Bombcrafter");
                break;*/
                //Inventory();
                break;

        case 318: //Bombcrafter OUT
                printf("\a");
                Player.SetX(3 * SPRITE_WIDTH);
                Player.SetY(3 * SPRITE_HEIGHT);
                Player.SetDirection(0);
                destroy_win(Player.objectWindow);

                Player.objectWindow = create_win(6,9,Player.GetY(),Player.GetX());
                Player.DrawSprite(Player,0);
                currentLevel.ImportMap(2,1,"NULL");
                break;

        //Ladders UP
        case 800: //Floor 0
                printf("\a");
                Player.SetX(1 * SPRITE_WIDTH);
                Player.SetY(1 * SPRITE_HEIGHT);
                Player.SetDirection(0);
                destroy_win(Player.objectWindow);

                Player.objectWindow = create_win(6,9,Player.GetY(),Player.GetX());
                Player.DrawSprite(Player,0);
                currentLevel.ImportMap(0,0,"FLOOR 1");
                break;

        case 801: //Floor 1
                progress = 1;
                printf("\a");
                Player.SetX(9 * SPRITE_WIDTH);
                Player.SetY(5 * SPRITE_HEIGHT);
                Player.SetDirection(0);
                destroy_win(Player.objectWindow);

                Player.objectWindow = create_win(6,9,Player.GetY(),Player.GetX());
                Player.DrawSprite(Player,0);
                currentLevel.ImportMap(0,0,"FLOOR 2");
```

```
                    break;

        case 802: //Floor 2
                    progress = 2;
                    printf("\a");
                    Player.SetX(1 * SPRITE_WIDTH);
                    Player.SetY(1 * SPRITE_HEIGHT);
                    Player.SetDirection(0);
                    destroy_win(Player.objectWindow);

                    Player.objectWindow = create_win(6,9,Player.GetY(),Player.GetX());
                    Player.DrawSprite(Player,0);
                    currentLevel.ImportMap(0,0,"FLOOR 3");
                    break;

        case 803: //Floor 3
                    progress = 3;
                    printf("\a");
                    Player.SetX(17 * SPRITE_WIDTH);
                    Player.SetY(1 * SPRITE_HEIGHT);
                    Player.SetDirection(0);
                    destroy_win(Player.objectWindow);

                    Player.objectWindow = create_win(6,9,Player.GetY(),Player.GetX());
                    Player.DrawSprite(Player,0);
                    currentLevel.ImportMap(0,0,"FLOOR 4");
                    break;

        case 804: //Floor 4
                    progress = 4;
                    printf("\a");
                    Player.SetX(2 * SPRITE_WIDTH);
                    Player.SetY(9 * SPRITE_HEIGHT);
                    Player.SetDirection(0);
                    destroy_win(Player.objectWindow);

                    Player.objectWindow = create_win(6,9,Player.GetY(),Player.GetX());
                    Player.DrawSprite(Player,0);
                    currentLevel.ImportMap(0,0,"FLOOR 5");
                    break;

        case 805: //Floor
                    progress = 5;
                    printf("\a");
                    Player.SetX(18 * SPRITE_WIDTH);
                    Player.SetY(3 * SPRITE_HEIGHT);
```

```
                Player.SetDirection(0);
                destroy_win(Player.objectWindow);

                Player.objectWindow = create_win(6,9,Player.GetY(),Player.GetX());
                Player.DrawSprite(Player,0);
                currentLevel.ImportMap(0,0,"FLOOR 7");
                break;

        case 806: //Floor 6
                progress = 6;
                printf("\a");
                Player.SetX(18 * SPRITE_WIDTH);
                Player.SetY(3 * SPRITE_HEIGHT);
                Player.SetDirection(0);
                destroy_win(Player.objectWindow);
                Player.objectWindow = create_win(6,9,Player.GetY(),Player.GetX());
                Player.DrawSprite(Player,0);
                currentLevel.ImportMap(0,0,"FLOOR 7");
                break;

        case 807: //Floor 7
                progress = 7;
                printf("\a");
                Player.SetX(3 * SPRITE_WIDTH);
                Player.SetY(9 * SPRITE_HEIGHT);
                Player.SetDirection(0);
                destroy_win(Player.objectWindow);
                Player.objectWindow = create_win(6,9,Player.GetY(),Player.GetX());
                Player.DrawSprite(Player,0);
                currentLevel.ImportMap(0,0,"FLOOR 8");
                break;

        case 808: //Floor 8
                progress = 8;
                printf("\a");
                Player.SetX(1 * SPRITE_WIDTH);
                Player.SetY(6 * SPRITE_HEIGHT);
                Player.SetDirection(0);
                destroy_win(Player.objectWindow);
                Player.objectWindow = create_win(6,9,Player.GetY(),Player.GetX());
                Player.DrawSprite(Player,0);
                currentLevel.ImportMap(0,0,"FLOOR 9");
                break;

        case 809: //Floor 9
                progress = 9;
                printf("\a");
                Player.SetX(19 * SPRITE_WIDTH);
                Player.SetY(5 * SPRITE_HEIGHT);
                Player.SetDirection(0);
                destroy_win(Player.objectWindow);
                Player.objectWindow = create_win(6,9,Player.GetY(),Player.GetX());
                Player.DrawSprite(Player,0);
                currentLevel.ImportMap(0,0,"FLOOR 10");
                break;

        case 810: //Floor 10
                progress = 10;
                printf("\a");
                Player.SetX(2 * SPRITE_WIDTH);
                Player.SetY(8 * SPRITE_HEIGHT);
                Player.SetDirection(0);
                destroy_win(Player.objectWindow);

                Player.objectWindow = create_win(6,9,Player.GetY(),Player.GetX());
                Player.DrawSprite(Player,0);
                currentLevel.ImportMap(0,0,"FLOOR 11");
                break;

        case 811: //Floor 11
                progress = 11;
                printf("\a");
                Player.SetX(1 * SPRITE_WIDTH);
                Player.SetY(9 * SPRITE_HEIGHT);
                Player.SetDirection(0);
                destroy_win(Player.objectWindow);

                Player.objectWindow = create_win(6,9,Player.GetY(),Player.GetX());
                Player.DrawSprite(Player,0);
                finalstep = Step;
                currentLevel.ImportMap(0,0,"TEMP 5");
                break;
       case 812: //Floor 12
                progress = 12;
                printf("\a");
                Player.SetX(18 * SPRITE_WIDTH);
                Player.SetY(9 * SPRITE_HEIGHT);
                Player.SetDirection(0);
                destroy_win(Player.objectWindow);

                Player.objectWindow = create_win(6,9,Player.GetY(),Player.GetX());
                Player.DrawSprite(Player,0);
```

```
                  finalstep = Step;
                  currentLevel.ImportMap(0,0,"TEMP 7");
                  break;
//Ladders DOWN
         case -800: //Floor 0
                  printf("\a");
                  Player.SetX(1 * SPRITE_WIDTH);
                  Player.SetY(1 * SPRITE_HEIGHT);
                  Player.SetDirection(0);
                  destroy_win(Player.objectWindow);

                  Player.objectWindow = create_win(6,9,Player.GetY(),Player.GetX());
                  Player.DrawSprite(Player,0);
                  currentLevel.ImportMap(0,0,"FLOOR 0");
                  break;

         case 813: //Top
                  printf("\a");
                  Player.SetX(10 * SPRITE_WIDTH);
                  Player.SetY(7 * SPRITE_HEIGHT);
                  Player.SetDirection(1);
                  destroy_win(Player.objectWindow);

                  Player.objectWindow = create_win(6,9,Player.GetY(),Player.GetX());
                  Player.DrawSprite(Player,0);
                  currentLevel.ImportMap(0,9,"0,10");
                  break;

         //case -802: //Floor 2
         //       printf("\a");
         //       Player.SetX(1 * SPRITE_WIDTH);
         //       Player.SetY(1 * SPRITE_HEIGHT);
         //       Player.SetDirection(0);
         //       destroy_win(Player.objectWindow);

         //       Player.objectWindow = create_win(6,9,Player.GetY(),Player.GetX());
         //       Player.DrawSprite(Player,0);
         //       currentLevel.ImportMap(0,0,"FLOOR 2");
         //       break;

         //case -803: //TEMP 3
         //       printf("\a");
         //       Player.SetX(19 * SPRITE_WIDTH);
         //       Player.SetY(3 * SPRITE_HEIGHT);
         //       Player.SetDirection(0);
         //       destroy_win(Player.objectWindow);
```

```

        //        Player.objectWindow = create_win(6,9,Player.GetY(),Player.GetX());
        //        Player.DrawSprite(Player,0);
        //        currentLevel.ImportMap(0,0,"TEMP 3");
        //        break;

        //case -804: //TEMP 4
        //        printf("\a");
        //        Player.SetX(1 * SPRITE_WIDTH);
        //        Player.SetY(9 * SPRITE_HEIGHT);
        //        Player.SetDirection(0);
        //        destroy_win(Player.objectWindow);

        //        Player.objectWindow = create_win(6,9,Player.GetY(),Player.GetX());
        //        Player.DrawSprite(Player,0);
        //        currentLevel.ImportMap(0,0,"TEMP 4");
        //        break;

        //case -805: //TEMP 5
        //        printf("\a");
        //        Player.SetX(14 * SPRITE_WIDTH);
        //        Player.SetY(3 * SPRITE_HEIGHT);
        //        Player.SetDirection(0);
        //        destroy_win(Player.objectWindow);

        //        Player.objectWindow = create_win(6,9,Player.GetY(),Player.GetX());
        //        Player.DrawSprite(Player,0);
        //        currentLevel.ImportMap(0,0,"TEMP 5");
        //        break;

        //case -806: //TEMP 6
        //        printf("\a");
        //        Player.SetX(19 * SPRITE_WIDTH);
        //        Player.SetY(8 * SPRITE_HEIGHT);
        //        Player.SetDirection(0);
        //        destroy_win(Player.objectWindow);

        //        Player.objectWindow = create_win(6,9,Player.GetY(),Player.GetX());
        //        Player.DrawSprite(Player,0);
        //        currentLevel.ImportMap(0,0,"TEMP 6");
        //        break;
        // NPC's
        case 702: //Blacksmith
                if (hammer)
                        DrawMessage(-1,-1,40,8,"Blacksmith: That hammer will break  down most cracked walls.");
                else
```

```
                    {
                    DrawMessage(-1,-1,40,8,"Blacksmith: You there! You don't    seem to have a hammer.");
                    DrawMessage(-1,-1,40,8,"Blacksmith: There's no way you'll   make it through the tower without   one.");
                    DrawMessage(-1,-1,40,8,"Blacksmith: I'm not using this one  anymore, maybe you can put it to    good use.");
                    DrawMessage(-1,-1,40,8,"You obtained HAMMER!");
                    hammer = true;
                    }
                    break;
          case 704: //Lantern Man
                    if (lantern)
                              DrawMessage(-1,-1,40,8,"Nervous Man: Good Luck!");
                    else
                    {
                    DrawMessage(-1,-1,40,8,"Nervous Man: Wait! You can't go in  there, it's too dark.");
                    DrawMessage(-1,-1,40,8,"Nervous Man: Here, take my lantern. I'm too scared to go in anyways.");
                    DrawMessage(-1,-1,40,8,"You obtained LANTERN!");
                    lantern = true;
                    }
                    break;
          case 708: //Breakable Wall
                    if (hammer)
                    {
                              DrawMessage(-1,-1,40,8,"You broke the wall down with your   hammer!");
                              currentLevel.roomObject[index].SetObjIndex(0);
                              currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],0);
                    }
                    else
                              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.");
                    break;
          case 701: //King
                    DrawMessage(-1,-1,40,8,"Duke: Congratulations on making it  though the Castellum!");
                    DrawMessage(-1,-1,40,8,"Duke: I hereby present you with yourcertificate of Adventurity, you may now venture beyond the town walls.");
                    DrawMessage(-1,-1,40,8,"Duke: Go forth Adventurer! The worldawaits!");
                    DrawMessage(-1,-1,40,8,"Congratulations! You complete in ?? steps.");
                    DrawMessage(-1,-1,40,8,"Press 'I' then 'Esc' to exit.");
                    break;
          }
}

void Switch(int &x, int &y, Object &player_object)
{
          if (player_object.GetX()/SPRITE_WIDTH == 21 && (GetAsyncKeyState(VK_RIGHT) && 0x8000))//Right
          {
                    roomX += 1;
                    Player.SetX(0);
                    destroy_win(Player.objectWindow);
                    Player.objectWindow = create_win(6,9,Player.GetY(),Player.GetX());
                    Player.DrawSprite(Player,0);
                    currentLevel.ImportMap(roomX,roomY,"NULL");
          }

          if (player_object.GetX()/SPRITE_WIDTH == 0 && (GetAsyncKeyState(VK_LEFT) && 0x8000))//Left
          {
                    roomX -= 1;
                    Player.SetX(21 * SPRITE_WIDTH);
                    destroy_win(Player.objectWindow);
                    Player.objectWindow = create_win(6,9,Player.GetY(),Player.GetX());
                    Player.DrawSprite(Player,0);
                    currentLevel.ImportMap(roomX,roomY,"NULL");
          }

          if (player_object.GetY()/SPRITE_HEIGHT == 10 && (GetAsyncKeyState(VK_DOWN) && 0x8000))//Down
          {
                    roomY += 1;
                    Player.SetY(0);
                    destroy_win(Player.objectWindow);
                    Player.objectWindow = create_win(6,9,Player.GetY(),Player.GetX());
                    Player.DrawSprite(Player,0);
                    currentLevel.ImportMap(roomX,roomY,"NULL");
          }

          if (player_object.GetY()/SPRITE_HEIGHT == 0 && (GetAsyncKeyState(VK_UP) && 0x8000))//Up
          {
                    roomY -= 1;
                    Player.SetY(10 * SPRITE_HEIGHT);
                    destroy_win(Player.objectWindow);
                    Player.objectWindow = create_win(6,9,Player.GetY(),Player.GetX());
                    Player.DrawSprite(Player,0);
                    currentLevel.ImportMap(roomX,roomY,"NULL");
          }
}
```

## MazeFilter.cpp

```
#include "MazeFilter.h"
#include "CreateObject.h"
#include "Object.h"
#include "GlobalVars.h"
#include "DrawWindows.h"

int Index(int x, int y);//Prototype for "Index" function
int BlockVal(int x, int y);//Prototype for "BlockVal" function
int SetBlockObj(int x, int y);//Prototype for "SetBlock" function
int SetBlockSprite(int x, int y);//Prototype for "SetBlock" function

const int MAP_HEIGHT = 11;
const int MAP_WIDTH = 22;
const int SPRITE_HEIGHT = 6;
const int SPRITE_WIDTH = 9;

void MazeFilter()
{
        for (int i = 0; i < MAP_HEIGHT; i++)
        {
                for (int j = 0; j < MAP_WIDTH; j++) //This ultimately checks all map spaces
                {
                        if (currentLevel.roomObject[Index(j,i)].GetObjIndex() != 0)//If a block is not empty, assign it to what it should be.
                        {
                                int ani[4] = {0,0,0,0};
                                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));
                        }
                        else if (currentLevel.roomObject[Index(j,i)].GetObjIndex() == 0)
                        {
                                int ani[4] = {0,0,0,0};
                                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);
                        }
                }
        }
}

int Index(int x, int y)
{
        return (y * MAP_WIDTH) + x;
}

int BlockVal (int x, int y)//Prevents calling a window that doesn't exist
{
        if (x >= 0 && x < 21 && y >= 0 && y < 11)
        {
                return (currentLevel.roomObject[Index(x,y)].GetObjIndex());
        }
        else return (0);
}

int SetBlockObj(int x, int y)//Returns the value for the 'object' parameter
{
        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
```

```
                return (82);
        else if (BlockVal (x+1,y) != 0 && BlockVal (x,y+1) != 0 && BlockVal (x-1,y) != 0)// -,-    Left, right, and down
                return (83);
        else if (BlockVal (x+1,y) != 0 && BlockVal (x-1,y) != 0 && BlockVal (x,y-1) != 0)// -'-    Left, right, and up
                return (84);
        else if (BlockVal (x,y+1) != 0 && BlockVal (x-1,y) != 0 && BlockVal (x,y-1) != 0)// -|     Left, up, and down
                return (85);
        else if (BlockVal (x+1,y) != 0 && BlockVal (x,y+1) != 0 && BlockVal (x,y-1) != 0)//  |-    Right, up, and down
                return (86);
        else if (BlockVal (x+1,y) != 0 && BlockVal (x,y+1) != 0)//  ,-    Right and down
                return (89);
        else if (BlockVal (x-1,y) != 0 && BlockVal (x,y+1) != 0)// -,     Left and down
                return (90);
        else if (BlockVal (x+1,y) != 0 && BlockVal (x,y-1) != 0)//  '-    Right and up
                return (91);
        else if (BlockVal (x-1,y) != 0 && BlockVal (x,y-1) != 0)// -'     Left and up
                return (92);
        else if (BlockVal (x+1,y) != 0 && BlockVal (x-1,y) != 0)// - -    Left and right
                return (79);
        else if (BlockVal (x,y+1) != 0 && BlockVal (x,y-1) != 0)//  |     Up and down
                return (73);
        else if (BlockVal (x,y-1) != 0)//   '     Up
                return (76);
        else if (BlockVal (x,y+1) != 0)//   ,     Down
                return (72);
        else if (BlockVal (x+1,y) != 0)//    -    Right
                return (81);
        else if (BlockVal (x-1,y) != 0)//  -      Left
                return (80);
        else
                return (93);//         None
}

int SetBlockSprite(int x, int y)//Returns the value for the 'sprite' parameter
{
        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
                return (165);
        else if (BlockVal (x+1,y) != 0 && BlockVal (x,y+1) != 0 && BlockVal (x-1,y) != 0)// -,-    Left, right, and down
                return (166);
        else if (BlockVal (x+1,y) != 0 && BlockVal (x-1,y) != 0 && BlockVal (x,y-1) != 0)// -'-    Left, right, and up
                return (167);
        else if (BlockVal (x,y+1) != 0 && BlockVal (x-1,y) != 0 && BlockVal (x,y-1) != 0)// -|     Left, up, and down
                return (168);
        else if (BlockVal (x+1,y) != 0 && BlockVal (x,y+1) != 0 && BlockVal (x,y-1) != 0)//  |-    Right, up, and down
                return (169);
        else if (BlockVal (x+1,y) != 0 && BlockVal (x,y+1) != 0)//  ,-    Right and down
```

```
                return (173);
        else if (BlockVal (x-1,y) != 0 && BlockVal (x,y+1) != 0)// -,     Left and down
                return (174);
        else if (BlockVal (x+1,y) != 0 && BlockVal (x,y-1) != 0)//  '-    Right and up
                return (175);
        else if (BlockVal (x-1,y) != 0 && BlockVal (x,y-1) != 0)// -'     Left and up
                return (176);
        else if (BlockVal (x+1,y) != 0 && BlockVal (x-1,y) != 0)// - -    Left and right
                return (162);
        else if (BlockVal (x,y+1) != 0 && BlockVal (x,y-1) != 0)//  |     Up and down
                return (156);
        else if (BlockVal (x,y-1) != 0)//   '     Up
                return (159);
        else if (BlockVal (x,y+1) != 0)//   ,     Down
                return (155);
        else if (BlockVal (x+1,y) != 0)//    -    Right
                return (164);
        else if (BlockVal (x-1,y) != 0)//  -      Left
                return (163);
        else
                return (177);//         None
}
```

## Object.cpp

```
//Includes
#include "Object.h"
#include "GlobalVars.h"
#include "DrawWindows.h"
#include "RoomController.h"
#include <curses.h>
#include <math.h>
#include <Windows.h>
#pragma comment (lib , "winmm.lib")

//Prototypes
Object DefineObject(int spr, int img, int ani[4], WINDOW *local_win, int v, int h, int y, int x, int dir, int object);
int distance(int x1, int y1, int x2, int y2);

//Variables
const int SPRITE_HEIGHT = 6;
const int SPRITE_WIDTH = 9;
const int MAP_HEIGHT = 11;
const int MAP_WIDTH = 22;

void Object::SetSprite(int spr, int img, int ani[4], WINDOW *local_win)
```

```
{         //Set the sprites
          spriteIndex = spr;
          imageIndex = img;
          for (int i = 0; i < 4; i++)
                    animations[i] = ani[i];
          objectWindow = local_win;
}

void Object::SetSpeed(int v, int h)
{         //Set the speed
          vspeed = v;
          hspeed = h;
}

void Object::SetDirection(int dir, int y, int x)
{         //Set position and orientation
          direction = dir;
          position[0] = y;
          position[1] = x;
}

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)
{         //Constructor
          Object local_object;
          int _ani[4];
          for (int i = 0; i < 4; i++)
                    _ani[i] = ani[i];
          local_object.SetSprite(spr, img, _ani, local_win);
          local_object.SetSpeed(v, h);
          local_object.SetDirection(dir, y, x);
          local_object.SetObjIndex(local_objIndex);
          return local_object;
}

Object boulderXY(int x, int y)
{
          for (int i = 0; i < MAP_HEIGHT; i ++)
          {
                    for (int j = 0; j < MAP_WIDTH; j ++)
                    {
                              if (currentLevel.puzzleObject[(i*MAP_WIDTH)+j].GetX() == x && currentLevel.puzzleObject[(i*MAP_WIDTH)+j].GetY() == y)
                                        return currentLevel.puzzleObject[(i*MAP_WIDTH)+j];
                    }
          }
          return currentLevel.puzzleObject[0];
}
```

```cpp

Object boulderXYS(int x, int y, int v, int h)
{
        for (int i = 0; i < MAP_HEIGHT; i ++)
        {
                for (int j = 0; j < MAP_WIDTH; j ++)
                {
                        if (currentLevel.puzzleObject[(i*MAP_WIDTH)+j].GetX() == x && currentLevel.puzzleObject[(i*MAP_WIDTH)+j].GetY() == y)
                        {
                                if (v == 0 && h == 0)
                                {

        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);
                                        currentLevel.puzzleObject[(i*MAP_WIDTH)+j].SetSpeed(0,0);
                                }
                                else
                                {

        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);
                                        currentLevel.puzzleObject[(i*MAP_WIDTH)+j].SetSpeed(v,h);
                                }
                        }
                }
        }
        return currentLevel.puzzleObject[0];
}

int Object::DrawSprite(Object local_object, int animationIndex)
{
        start_color();
        init_pair(1, COLOR_WHITE, COLOR_BLACK);
        init_pair(2, COLOR_BLACK, COLOR_BLACK);
        init_pair(3, COLOR_YELLOW, COLOR_BLACK);

        int x1, x2, y1, y2;

        wborder(local_object.objectWindow, ' ', ' ', ' ',' ',' ',' ',' ',' ');

        int k = local_object.spriteIndex + local_object.animations[(animationIndex + local_object.imageIndex)%4];
        if (local_object.GetImageIndex() == -1) // Don't animate
                k = local_object.spriteIndex;

        if (local_object.GetImageIndex() == -2) // Don't draw
                return 0;

        k += local_object.direction * 3;

        x2 = Player.position[1]/SPRITE_WIDTH;
        y2 = Player.position[0]/SPRITE_HEIGHT;

        if (!currentLevel.AreLightsOn())
        {
                x1 = local_object.position[1];
                y1 = local_object.position[0];

                if (distance(x1, y1, x2, y2) > 31)
                        return 0;
                if (distance(x1, y1, x2, y2) == 30 || distance(x1, y1, x2, y2) == 31 || distance(x1, y1, x2, y2) == 28 )
                {
                        for (int i=0; i < SPRITE_HEIGHT; i++)
                        {
                                for (int j=0; j < SPRITE_WIDTH; j++)
                                {
                                        mvwaddch(local_object.objectWindow,i,j,ASCII_ART[k][i][j] | COLOR_PAIR(2));
                                }
                        }
                        wnoutrefresh(local_object.objectWindow);
                        return 1;
                }
        }

        for (int i=0; i < SPRITE_HEIGHT; i++)
        {
                for (int j=0; j < SPRITE_WIDTH; j++)
                {
                        if (local_object.GetObjIndex() == 1000)
                        {
                                attron(COLOR_PAIR(3));
                                mvwprintw(local_object.objectWindow,1,1,"Score\n%d",Step);
                                attroff(COLOR_PAIR(3));
                                wnoutrefresh(local_object.objectWindow);
                        }
                        else
                                mvwaddch(local_object.objectWindow,i,j,ASCII_ART[k][i][j] | COLOR_PAIR(1));
                }
        }

        wnoutrefresh(local_object.objectWindow);
        return 1;
}

int Object::DrawLayerSprite(Object local_object, int animationIndex)
```

```
{
	wborder(local_object.objectWindow, ' ', ' ', ' ',' ',' ',' ',' ',' ');

	int index, x, y, k = local_object.spriteIndex + local_object.animations[(animationIndex + local_object.imageIndex)%4];

	if (local_object.GetImageIndex() == -1) // Don't animate
		k = local_object.spriteIndex;
	if (local_object.GetImageIndex() == -2) // Don't Draw
		return 0;

	k += local_object.direction * 3;

	for (int i=0; i < SPRITE_HEIGHT; i++)
	{
		for (int j=0; j < SPRITE_WIDTH; j++)
		{
			x = local_object.position[1] + j;
			y = local_object.position[0] + i;

			index = ((y-(y%SPRITE_HEIGHT))/SPRITE_HEIGHT);
			index = index * MAP_WIDTH;
			index += ((x-(x%SPRITE_WIDTH))/SPRITE_WIDTH);

			clear();

			if (ASCII_ART[k][i][j] != 'g')
				mvwaddch(local_object.objectWindow,i,j,ASCII_ART[k][i][j]);
			else
				mvwaddch(local_object.objectWindow,i,j,mvwinch(currentLevel.roomObject[index].objectWindow, y%SPRITE_HEIGHT, x%SPRITE_WIDTH));
		}
	}
	wnoutrefresh(local_object.objectWindow);
	return 1;
}

int distance(int x1, int y1, int x2, int y2)
{
	int distance = 0;
	double formula = ((x1-x2)*(x1-x2))+((y1-y2)*(y1-y2));

	formula = sqrt(formula);

	distance = int(floor(formula * 10));
	return distance;
}
```

.h PlayerMovement.cpp

```cpp
//Includes
#include "Object.h"
#include "DrawWindows.h"
#include "GlobalVars.h"
#include "Interactions.h"
#include <Windows.h>

//Prototypes
int PlayerMove(Object &boulder);
void MoveToSnap(Object &local_object);
bool CheckFree(int dir, Object &local_object, int d);
void Move(int dir, Object &local_object);
void Door(Object &boulder);
int BoulderMove(Object &boulder);

//Variables
const int MAP_WIDTH = 22;
const int MAP_HEIGHT = 11;
const int SPRITE_WIDTH = 9;
const int SPRITE_HEIGHT = 6;

int PlayerMove(Object &player_object)
{
	Switch(roomX, roomY, player_object);
	if (EnterWasPressed())
		Door(player_object);

	if (player_object.GetY()%SPRITE_HEIGHT == 0 && player_object.GetX()%SPRITE_WIDTH == 0)
	{
		player_object.SetImageIndex(-1);
		Move(0, player_object); //All Directions
	}
	else if (player_object.GetY()%SPRITE_HEIGHT == 0)
		Move(1, player_object); //Horizontal Only
	else if (player_object.GetX()%SPRITE_WIDTH == 0)
		Move(2, player_object); //Vertical Only

	MoveToSnap(player_object);
	return 0;
}

void MoveToSnap(Object &local_object)
```

```
{
	if (local_object.GetHSpeed() != 0 || local_object.GetVSpeed() != 0)
	{
		destroy_win(local_object.objectWindow);
		local_object.SetY(local_object.GetY()+local_object.GetVSpeed());
		local_object.SetX(local_object.GetX()+local_object.GetHSpeed());
		local_object.SetDirection(local_object.GetDirection(),local_object.GetY(),local_object.GetX());
		local_object.objectWindow = create_win(6,9,local_object.GetY(),local_object.GetX());
	}
}

bool CheckFree(int dir, Object &local_object, int d)
{
	int i = local_object.GetY()/SPRITE_HEIGHT;
	int j = local_object.GetX()/SPRITE_WIDTH;
	int index, indexF;

	switch (dir)
	{

	case 0: // Down
		i += 1;
		index = (i*MAP_WIDTH)+j;
		i += 1;
		indexF = (i*MAP_WIDTH)+j;

		if (boulderXY(j,i-1).GetObjIndex() == 45 && (currentLevel.roomObject[indexF].GetObjIndex() <= 0 || currentLevel.roomObject[indexF].GetObjIndex() == 55) && 
boulderXY(j,i).GetObjIndex() != 45 && d == 0)
		{
			boulderXYS(j,i-1,1,0);
			return false;
		}
		else if (boulderXY(j,i-1).GetObjIndex() == 45 && (currentLevel.roomObject[indexF].GetObjIndex() > 0 || boulderXY(j,i).GetObjIndex() == 45) && d == 0)
			return false;
		else if (currentLevel.roomObject[index].GetObjIndex() <= 0)
			return true;
		break;

	case 1: // Up
		i -= 1;
		index = (i*MAP_WIDTH)+j;
		i -= 1;
		indexF = (i*MAP_WIDTH)+j;

		if (boulderXY(j,i+1).GetObjIndex() == 45 && (currentLevel.roomObject[indexF].GetObjIndex() <= 0 || currentLevel.roomObject[indexF].GetObjIndex() == 55) && 
boulderXY(j,i).GetObjIndex() != 45 && d == 0)
			{
				boulderXYS(j,i+1,-1,0);
				return false;
			}
			else if (boulderXY(j,i+1).GetObjIndex() == 45 && (currentLevel.roomObject[indexF].GetObjIndex() > 0 || boulderXY(j,i).GetObjIndex() == 45) && d == 0)
				return false;
			else if (currentLevel.roomObject[index].GetObjIndex() <= 0)
				return true;
			break;

		case 2: // Right
			j += 1;
			index = (i*MAP_WIDTH)+j;
			j += 1;
			indexF = (i*MAP_WIDTH)+j;

			if (boulderXY(j-1,i).GetObjIndex() == 45 && (currentLevel.roomObject[indexF].GetObjIndex() <= 0 || currentLevel.roomObject[indexF].GetObjIndex() == 55) &&
boulderXY(j,i).GetObjIndex() != 45 && d == 0)
			{
				boulderXYS(j-1,i,0,1);
				return false;
			}
			else if (boulderXY(j-1,i).GetObjIndex() == 45 && (currentLevel.roomObject[indexF].GetObjIndex() > 0 || boulderXY(j,i).GetObjIndex() == 45) && d == 0)
				return false;
			else if (currentLevel.roomObject[index].GetObjIndex() <= 0)
				return true;
			break;

		case 3: // Left
			j -= 1;
			index = (i*MAP_WIDTH)+j;
			j -= 1;
			indexF = (i*MAP_WIDTH)+j;

			if (boulderXY(j+1,i).GetObjIndex() == 45 && (currentLevel.roomObject[indexF].GetObjIndex() <= 0 || currentLevel.roomObject[indexF].GetObjIndex() == 55) &&
boulderXY(j,i).GetObjIndex() != 45 && d == 0)
			{
				boulderXYS(j+1,i,0,-1);
				return false;
			}
			else if (boulderXY(j+1,i).GetObjIndex() == 45 && (currentLevel.roomObject[indexF].GetObjIndex() > 0 || boulderXY(j,i).GetObjIndex() == 45) && d == 0)
				return false;
			else if (currentLevel.roomObject[index].GetObjIndex() <= 0)
				return true;
```

```
                    break;

          }

          return false;
}

void Move(int dir, Object &local_object)
{
          if (currentLevel.roomObject[((local_object.GetY()/SPRITE_HEIGHT) * MAP_WIDTH) + (local_object.GetX()/SPRITE_WIDTH)].GetObjIndex() == -15)
                    if (local_object.GetY()%SPRITE_HEIGHT == 0 && local_object.GetX()%SPRITE_WIDTH == 0)
                    {
                              if (local_object.GetVSpeed() == 1 && !CheckFree(0,local_object, dir))
                              {
                                        local_object.SetSpeed(0,0);
                                        if (currentLevel.roomObject[0].GetObjIndex() == 999)
                                                  Step ++;
                              }
                              if (local_object.GetVSpeed() == -1 && !CheckFree(1,local_object, dir))
                              {
                                        local_object.SetSpeed(0,0);
                                        if (currentLevel.roomObject[0].GetObjIndex() == 999)
                                                  Step ++;
                              }
                              if (local_object.GetHSpeed() == 1 && !CheckFree(2,local_object, dir))
                              {
                                        local_object.SetSpeed(0,0);
                                        if (currentLevel.roomObject[0].GetObjIndex() == 999)
                                                  Step ++;
                              }
                              if (local_object.GetHSpeed() == -1 && !CheckFree(3,local_object, dir))
                              {
                                        local_object.SetSpeed(0,0);
                                        if (currentLevel.roomObject[0].GetObjIndex() == 999)
                                                  Step ++;
                              }
                    }

          if (dir == 0 && currentLevel.roomObject[((local_object.GetY()/SPRITE_HEIGHT) * MAP_WIDTH) + (local_object.GetX()/SPRITE_WIDTH)].GetObjIndex() != -15)
          {
                    if (local_object.GetHSpeed() != 0 || local_object.GetVSpeed() != 0)
                              if (currentLevel.roomObject[0].GetObjIndex() == 999)
                                        Step ++;
                    local_object.SetSpeed(0,0);
          }
```

```
	if (dir == 0 || dir == 1)
	{
		if (GetAsyncKeyState(VK_RIGHT) && 0x8000)
		{
			local_object.SetDirection(2);
			if (CheckFree(2, local_object, dir))
			{
				if (currentLevel.roomObject[((local_object.GetY()/SPRITE_HEIGHT) * MAP_WIDTH) + (local_object.GetX()/SPRITE_WIDTH)].GetObjIndex() != -15 ||
					(currentLevel.roomObject[((local_object.GetY()/SPRITE_HEIGHT) * MAP_WIDTH) + (local_object.GetX()/SPRITE_WIDTH)].GetObjIndex() == -15 &&
					local_object.GetHSpeed() == 0 && local_object.GetVSpeed() == 0))
					local_object.SetHSpeed(1);
				local_object.SetDirection(2);
				local_object.SetImageIndex(0);
			}
		}

		if (GetAsyncKeyState(VK_LEFT) && 0x8000)
		{
			local_object.SetDirection(3);
			if (CheckFree(3, local_object, dir))
			{
				if (currentLevel.roomObject[((local_object.GetY()/SPRITE_HEIGHT) * MAP_WIDTH) + (local_object.GetX()/SPRITE_WIDTH)].GetObjIndex() != -15 ||
					(currentLevel.roomObject[((local_object.GetY()/SPRITE_HEIGHT) * MAP_WIDTH) + (local_object.GetX()/SPRITE_WIDTH)].GetObjIndex() == -15 &&
					local_object.GetHSpeed() == 0 && local_object.GetVSpeed() == 0))
					local_object.SetHSpeed(-1);
				local_object.SetDirection(3);
				local_object.SetImageIndex(0);
			}
		}
	}

	if (dir == 0 || dir == 2)
	{
		if (GetAsyncKeyState(VK_DOWN) && 0x8000 && local_object.GetHSpeed() == 0)
		{
			local_object.SetDirection(0);
			if (CheckFree(0, local_object, dir))
			{
				if (currentLevel.roomObject[((local_object.GetY()/SPRITE_HEIGHT) * MAP_WIDTH) + (local_object.GetX()/SPRITE_WIDTH)].GetObjIndex() != -15 ||
					(currentLevel.roomObject[((local_object.GetY()/SPRITE_HEIGHT) * MAP_WIDTH) + (local_object.GetX()/SPRITE_WIDTH)].GetObjIndex() == -15 &&
					local_object.GetHSpeed() == 0 && local_object.GetVSpeed() == 0))
					local_object.SetVSpeed(1);
				local_object.SetDirection(0);
				local_object.SetImageIndex(0);
```

```
                        }
                }

                if (GetAsyncKeyState(VK_UP) && 0x8000 && local_object.GetHSpeed() == 0)
                {
                        local_object.SetDirection(1);
                        if (CheckFree(1, local_object, dir))
                        {
                                if (currentLevel.roomObject[((local_object.GetY()/SPRITE_HEIGHT) * MAP_WIDTH) + (local_object.GetX()/SPRITE_WIDTH)].GetObjIndex() != -15 ||
                                        (currentLevel.roomObject[((local_object.GetY()/SPRITE_HEIGHT) * MAP_WIDTH) + (local_object.GetX()/SPRITE_WIDTH)].GetObjIndex() == -15 &&
                                        local_object.GetHSpeed() == 0 && local_object.GetVSpeed() == 0))
                                        local_object.SetVSpeed(-1);
                                local_object.SetDirection(1);
                                local_object.SetImageIndex(0);
                        }
                }
        }
}

void Door(Object &player_object)
{
        int i,j, index;

        i = player_object.GetX()/SPRITE_WIDTH;
        j = player_object.GetY()/SPRITE_HEIGHT;

        switch (player_object.GetDirection())
        {
        case 0: //Down
                j += 1;
                break;
        case 1: //Up
                j -= 1;
                break;
        case 2: //Right
                i += 1;
                break;
        case 3: //Left
                i -= 1;
                break;
        }

        index = (j*MAP_WIDTH) + i;

        Interaction(currentLevel.roomObject[index].GetObjIndex(), index);
```

```
}

int BoulderMove(Object &boulder)
{
	switch (boulder.GetVSpeed())
	{
	case -1:
		boulder.SetDirection(1);
		break;
	case 0:
		switch (boulder.GetHSpeed())
		{
		case -1:
			boulder.SetDirection(3);
			break;
		case 0:
			break;
		case 1:
			boulder.SetDirection(2);
			break;
		}
		break;
	case 1:
		boulder.SetDirection(0);
		break;
	}

	if (boulder.GetHSpeed() != 0 || boulder.GetVSpeed() != 0)
	{
		MoveToSnap(boulder);
		if (boulder.GetX()%SPRITE_WIDTH == 0 && boulder.GetY()%SPRITE_HEIGHT == 0)
		{
			printf("\a");
			if (currentLevel.roomObject[((boulder.GetY()/SPRITE_HEIGHT)*MAP_WIDTH)+(boulder.GetX()/SPRITE_WIDTH)].GetObjIndex() == 55)
			{
				int ani[4] = { 0, 0, 1, 1};
				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);

	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);
				boulder.SetObjIndex(0);
				boulder.SetImageIndex(-2);
			}
			else if ((currentLevel.roomObject[((boulder.GetY()/SPRITE_HEIGHT)*MAP_WIDTH)+(boulder.GetX()/SPRITE_WIDTH)].GetObjIndex() == -15 &&
!CheckFree(boulder.GetDirection(),boulder,0)) ||
                currentLevel.roomObject[((boulder.GetY()/SPRITE_HEIGHT)*MAP_WIDTH)+(boulder.GetX()/SPRITE_WIDTH)].GetObjIndex() != -15)
            {
                boulder.SetSpeed(0,0);
                boulder.SetX(boulder.GetX()/SPRITE_WIDTH);
                boulder.SetY(boulder.GetY()/SPRITE_HEIGHT);
            }
        }
    }
    boulder.SetDirection(0);
    boulder.SetImageIndex(-1);
    return 0;
}
```

## RoomController.cpp

```
//Includes
#include "Object.h"
#include "DrawWindows.h"
#include "PlayerMovement.h"
#include "GlobalVars.h"
#include <curses.h>
#include <string>
#include <Windows.h>
#include <fstream>

//Prototypes
void BoulderFunctions(int index);

//Variables
const int MAP_HEIGHT = 11;
const int MAP_WIDTH = 22;
const int SPRITE_HEIGHT = 6;
const int SPRITE_WIDTH = 9;
int updatedSprite = 0, k, l;
char ASCII_ART[400][6][9];

int RoomUpdate()//Code that updates all of the objects
{
    Sleep(30);
    int index;

    PlayerMove(Player);

    for (int i=0; i < MAP_HEIGHT; i++)
```

```
	{	//Nested for loops to check all 242 objects
		for (int j=0; j < MAP_WIDTH; j++)
		{
			index = (i*MAP_WIDTH)+j;
			currentLevel.roomObject[index].DrawSprite(currentLevel.roomObject[index],updatedSprite); //Draw each of their sprites animated
		}
	}

	if (Player.GetHSpeed() != 0 || Player.GetVSpeed() != 0)
		Player.DrawLayerSprite(Player,updatedSprite); //Draw the player sprite animated
	else
		Player.DrawLayerSprite(Player,0); //Draw the player sprite animated

	for (int i=0; i < MAP_HEIGHT; i++)
	{	//Nested for loops to check all 242 objects
		for (int j=0; j < MAP_WIDTH; j++)
		{
			index = (i*MAP_WIDTH)+j;
			if (currentLevel.puzzleObject[index].GetObjIndex() == 45)
			{
				BoulderMove(currentLevel.puzzleObject[index]);
				currentLevel.puzzleObject[index].DrawLayerSprite(currentLevel.puzzleObject[index],0); //Draw each of their sprites animated
			}
		}
	}

	if (currentLevel.roomObject[0].GetObjIndex() == 999)
	{
		for (int i = 0; i < SPRITE_HEIGHT; i ++)
			for (int j = 0; j < SPRITE_WIDTH; j ++)
				mvwaddch(currentLevel.roomObject[0].objectWindow,i,j,' ');
		attron(COLOR_PAIR(3));
		mvwprintw(currentLevel.roomObject[0].objectWindow,2,2,"Score");
		mvwprintw(currentLevel.roomObject[0].objectWindow,3,2,"%d",Step);
		attroff(COLOR_PAIR(3));
		wborder(currentLevel.roomObject[0].objectWindow, '|', '|', '-', '-', '*', '*', '*', '*');

		wnoutrefresh(currentLevel.roomObject[0].objectWindow);
	}

	doupdate();

	if (updatedSprite == 3)
		updatedSprite = -1;
	updatedSprite ++;

	if (InventoryWasPressed())
	{
		return DrawInventory("Inventory", -1, 1);
	}
	return 1;
}

int ImportAsciiArt(std::string file)
{
	std::string word;
	std::ifstream fileName;
	fileName.open(file);
	fileName >> word;

	if (fileName.is_open())
	{
		while (fileName.good())
		{
			for (int i=0; i< 400; i++)
			{
				for (int j=0; j < 6; j++)
				{
					for (int k=0; k < 9; k++)
					{
						if (word[k] == 'G')
							ASCII_ART[i][j][k] = ' ';//Blank if it is a G
						else
							ASCII_ART[i][j][k] = word[k];//Ascii_art[Sprite][Y][X]
					}
					fileName >> word;
				}
			}
		}
		fileName.close();
	}
	else
	{
		printf("The Ascii Art the computer is trying to load doesn't exist\n"); //Apologize
		printf("Try re-installing the game\n");
		getch();
		return 0;
	}
	return 1;
}
```

# Headers

## CreateObject.h

```
#ifndef CREATE_OBJECT
#define CREAte_OBJECT
         void DefineObjects(int object, int y, int x);
#endif
```

## DrawWindows.h

```
#include "Object.h"

#ifndef MENU
#define MENU
         int MainMenu();
         int DrawBackground(char* file, int doGet);
         WINDOW *create_win(int height, int width, int starty, int startx);
         void destroy_win(WINDOW *local_win);
         void DrawMessage(int y, int x, int w, int h, char* message);
         int DrawInventory(char* title, int item, int price);
         extern Object Player;
#endif
```

## GlobalVars.h

```
#include "ImportMaps.h"
//These variables can be accessed throughout all cpp files
#ifndef GLOBAL_VARS
#define GLOBAL_VARS
         extern Level currentLevel;
         extern int gameState;
         extern int roomX;
         extern int roomY;
         extern int Step;
         extern char* global_item[4];
         extern int money;
         extern int save;

         bool EnterWasPressed();
         bool InventoryWasPressed();
#endif
```

## ImportMaps.h

```
#include <string>
#include <Windows.h>
#include "Object.h"

#ifndef IMPORT_MAPS
#define IMPORT_MAPS

class Level
{
private:
    bool lights;
          int id;

public:
          Level(); //Default Constructor

          Object roomObject[242];
          Object puzzleObject[242];

          bool AreLightsOn() { return lights; }
          int GetLevelID() { return id; }
          int ImportMap(int x, int y, std::string mapName);
          void ToggleLights();
          void GenerateMaze();
};

#endif
```

## Interactions.h

```
#include "Object.h"

#ifndef INTERACTIONS
#define INTERACTIONS
          void Interaction(int obj, int index);
          void Switch(int &x, int &y, Object &player_object);
          extern bool lantern;
          extern int progress;
          extern int totalfloors;
#endif
```

## MazeFilter.h

```
#ifndef MAZE_FILTER
#define MAZE_FILTER
          void MazeFilter();
```

```
#endif
```

## Object.h

```
#include <curses.h>
#include <Windows.h>

#ifndef OBJECT
#define OBJECT

class Object
{
private:
    int spriteIndex;
          int objIndex;
    int imageIndex;
    int hspeed;
    int vspeed;
          int direction;
          //       for direction   1
          //                                      3   2
          //                                              0
          int position[2];//x y coordinates
public:
          WINDOW *objectWindow;
          int animations[4];

          void SetSpriteIndex(int s){ spriteIndex = s; }
          void SetImageIndex(int i){ imageIndex = i; }
          void SetDirection(int d){ direction = d; }
          void SetObjIndex(int o){ objIndex = o; }
          void SetHSpeed(int h){ hspeed = h; }
          void SetVSpeed(int v){ vspeed = v; }
          void SetX(int x){ position[1] = x; }
          void SetY(int y){ position[0] = y; }

          int GetSpriteIndex(){ return spriteIndex; }
          int GetImageIndex(){ return imageIndex; }
          int GetDirection(){ return direction; }
          int GetObjIndex(){ return objIndex; }
          int GetVSpeed(){ return vspeed; }
          int GetHSpeed(){ return hspeed; }
          int GetX(){ return position[1]; }
          int GetY(){ return position[0]; }

          //Functions for the Object Class
```

```
        void SetSprite(int spr, int img, int ani[4], WINDOW *local_win);
        void SetSpeed(int v, int h);
        void SetDirection(int dir, int y, int x);
        int DrawSprite(Object local_object, int animationIndex);
        int DrawLayerSprite(Object local_object, int animationIndex);
};

        int distance(int x1, int y1, int x2, int y2);
        Object boulderXY(int x, int y);
        Object boulderXYS(int x, int y, int v, int h);

        Object DefineObject(int spr, int img, int ani[4], WINDOW *local_win, int v, int h, int y, int x, int dir, int object);

#endif
```

## PlayerMovement.h

```
#include "Object.h"

#ifndef PLAYER_MOVEMENT
#define PLAYER_MOVEMENT
        int PlayerMove(Object &player_object);
        int BoulderMove(Object &boulder);
#endif
```

## RoomController.h

```
#include <string>

#ifndef ROOM_CONTROLLER
#define ROOM_CONTROLLER
        int ImportAsciiArt(std::string file);
        int RoomUpdate();

        extern char ASCII_ART[400][6][9];
#endif
```