Adgp 101
Assessment #1
Jeremy Barzas

I Requirements Documentation
I 1. Description of the Problem

Name: Wumpus World

Problem Statement:
The player must navigate their robot character around the 4x4 grid and get the gold then return the the starting point to win.

Problem Specification:
If the player's robot character moves to a cell that has a wumpus or a pit you lose. if you are on a cell that is next to a pit, gold , or wumpus a warning message of breeze, shimmer, or stench displays to let you know that one or more dangers are near.

I 2. Input Information:

I 2. 1 Input Streams:

Name:N/A
Description:N/A
Format:N/A
Size:N/A
Sample:N/A

I 2. 2 Input Items:

Description: cin is the method i used to get input from the user to move the character.

Type: I used the "char" variable type to accept the input of 'w','a','s','d', as the arrow keys to move.

I 3. Output Information

Output Streams:

Name: N/A
Description: N/A
Format: N/A
Size: N/A

Sample: N/A
Output Items:

Description: cout is the output method I used to display the games information to the screen. In the game the user will first see "You enter a dark cave which direction do you go?" and be prompted to give and input of ‘w’,’a’,’s’,’d’.

Type: char, int, and strings will be printed to the screen.
Range of acceptable values: N/A

I 4. User Interface Information

I 4. 1 Description: The player is show the message "You enter a dark cave which direction do you go?". Then they will be prompted to input a direction with ‘w’,’a’,’s’,’d’. when the player’s robot is on a cell that is next to a wumpus is will display “You smell a stench...”, if the player’s robot is on a cell that is next to a pit it will display “You feel a breeze.”, if the player’s robot is on a cell next to the gold it will display “You see a glimmer”. The player will not be able to see the grid of cell at all and have to navigate based on the warning messages.

Objective:
The player must get to the cell with the gold without moving to a cell with a pit or wumpus and then return to the start location to win.

Danger:
Wumpus - Creature that will destroy your robot if you move on a cell that contains it.
Pit - Hole in the ground that will destroy you robot if you move to a cell that contains it.

II Design Documentation

II 1. System Architecture Description

The objects used in the game are a cell class, a player class, and a position struct. These are located in the header file.

II 2. Information about the Objects

Name: position
Description: this struct gives an x and y variable to be used as points on a grid for other objects to have location.

Struct attributes: two intergers x and y

Class Name: Player
Description: Creates a player that has a position and state of being alive or dead.

Class Attributes:
Name: location
Description:gives the player’s robot a position
Type: struct position

Name: alive
Description: gives the player the ability to be alive or dead.
Type: bool

Name: Player
Description: constructor for the player class
Type: constructor function

Name: locationPtr
Description: creates a pointer to be able to reference the player’s robot’s position.
Type: struct position/pointer

Name: alivePtr
Description: creates a pointer to be able to reference the player’s robot’s state of being alive or dead.
Type: bool/pointer

Class Name : Cell
Description: Creates a cell class to contain location, pit, breeze, gold, glimmer, wumpus, and stench for each grid cell.

Class Attributes:
Name: location
Description: gives the cell a grid position
Type: struct position

Name: pit
Description: makes the cell able to contain a pit
Type: bool

Name: breeze
Description: makes the cell able to contain a breeze
Type: bool

Name: gold
Description: makes the cell able to contain a gold
Type: bool

Name: glimmer
Description: makes the cell able to contain a glimmer
Type: bool

Name: wumpus
Description: makes the cell able to contain a wumpus
Type: bool

Name: stench
Description: makes the cell able to contain a stench
Type: bool

Name: Cell
Description: constructor for the cell class
Type: constructor function

Name: locationPtr
Description: creates a pointer to be able to reference the cell's position.
Type: struct position/pointer

Name: pitPtr
Description: creates a pointer to be able to reference if the cell has a pit.
Type: bool/pointer

Name: breezePtr
Description: creates a pointer to be able to reference if the cell has a breeze.
Type: bool/pointer

Name: goldPtr
Description: creates a pointer to be able to reference if the cell has a gold.
Type: bool/pointer

Name: glimmerPtr
Description: creates a pointer to be able to reference if the cell has a glimmer.
Type: bool/pointer

Name: wumpusPtr
Description: creates a pointer to be able to reference if the cell has a wumpus.

Type: bool/pointer
Name: stenchPtr
Description: creates a pointer to be able to reference if the cell has a stench.
Type: bool/pointer

III Implementation Documentation

III 1. Source code

Header File:
Name: Classes.h
Code:

```
#pragma once
#include <iostream>

using namespace std;

struct position
{
	int x;
	int y;
};

class Cell
{
private:
	position location;
	bool pit;
	bool breeze;
	bool gold;
	bool glimmer;
	bool wumpus;
	bool stench;

public:
	Cell(position, bool, bool, bool, bool ,bool, bool);
	position* locationPtr = &location;
	bool* pitPtr = &pit;
	bool* breezePtr = &breeze;
	bool* goldPtr = &gold;
	bool* glimmerPtr = &glimmer;
	bool* wumpusPtr = &wumpus;
```

```
        bool* stenchPtr = &stench;
};

class Player
{
        private:
                position location;
                bool alive;

        public:
                Player(position, bool);
                position* locationPtr = &location;
                bool* alivePtr = &alive;
};
```

Source File:
Name: Source.cpp
Code:

```
#include "Classes.h"

Cell::Cell(position l, bool h, bool b, bool g, bool glim, bool w, bool s)
{
        location = l;
        pit = h;
        breeze = b;
        gold = g;
        glimmer = glim;
        wumpus = w;
        stench = s;
}

Player::Player(position l, bool a)
{
        location = l;
        alive = a;
}
```

Source File:
Name: Main.cpp
Code:

```
#include "Classes.h"
```

```
void createGrid(int rows, int cols, Cell g[])
{
	for (int i = 0; i < rows; i++)
	{
		for (int j = 0; j < cols; j++)
		{
			cout << "x: " << i << " y: " << j;
			cout << endl;
		}
	}
}

int main()
{
	char input;

	Player robot = Player({ 0,0 }, true);

	Cell grid[16] =
	{
	//||position||pit||breeze||gold||glimmer||wumpus||stench
	Cell({ 0,0 }, false, false, false, false, false, false),
	Cell({ 0,1 }, false, true, false, false, false, false),
	Cell({ 0,2 }, false, false, false, false, false, false),
	Cell({ 0,3 }, false, false, false, false, false, true),
	Cell({ 1,0 }, false, true, false, false, false, false),
	Cell({ 1,1 }, true, false, false, false, false, false),
	Cell({ 1,2 }, false, true, false, false, false, true),
	Cell({ 1,3 }, false, false, false, false, true, false),
	Cell({ 2,0 }, false, true, false, false, false, false),
	Cell({ 2,1 }, false, true, false, false, false, false),
	Cell({ 2,2 }, false, true, false, false, false, false),
	Cell({ 2,3 }, false, false, false, true, false, true),
	Cell({ 3,0 }, true, false, false, false, false, false),
	Cell({ 3,1 }, false, true, false, false, false, false),
	Cell({ 3,2 }, true, false, false, false, false, false),
	Cell({ 3,3 }, false, false, true, false, false, false),
	};

	createGrid(4, 4, grid);

	do
	{
		cout << "Use 'w','a','s','d' as the arrow keys to move." << endl;
		cout << "You enter a dark cave which direction do you go?" << endl;

		cin >> input;

		if (!(input == 'w' || input == 'a' || input == 's' || input == 'd'))
		{
			cout << "That is not a valid input..." << endl;
			cout << "Use 'w','a','s','d' as the arrow keys to move." << endl;
		}

		else if(input == 'w')
		{
			*robot.locationPtr +
		}

		else if (input == 'a')
		{

		}

		else if (input == 's')
		{

		}

		else if (input == 'd')
		{

		}

	} while (*robot.alivePtr == true);


	system("pause");
}
```