**Donray Williams**

**10/5/16**

**Text-Based Adventure Game– Assessment 1 – Year 1**

# **1.0 Requirements Documentation**

# **1.1 Description of problem**

**A. Name:** Text-Based Adventure Game

**B. Problem Statement:** Create a basic text based adventure game within the console using Max’s String Class.

**C. Problem Specification:** You are to create a text based adventure game that can be played in the console. The interactive fiction world should allow the player to navigate an imaginary world by entering commands into the console. The player should be able to navigate between locations by typing commands such as “North”, “South”, “East”, and “West”. Using Max’s String class, it compared entered text against commands to ensure that the user has entered a valid command. Each room should have a description in it that prints on to the console when its entered.

**1.2 Input Information**

A. The user must enter certain commands that are set for that room, to move from room to room type in that direction that you would like to move in (ex. “East” to move East). To take a weapon of choice just type (ex. “Take Mace” or “Take Stick”). To answer the programming joke just type in the correct answer for it (ex. “1” or “True”) would be the same. Typing in “Help” at any time once you are in the game will come up with some instructions.

**1.3 Output Information**

A. The console must display what is inside of each room and where you can go, also what you can or cannot do.

**1.4 User Interface**

A. Not applicable.

# **2.0 System Architecture**

**2.1 Member Functions in the class:**

**PROTOTYPE:** int m\_North();

**DESCRIPTION:** Moves position north from where you currently are.

**PRECONDITION:** You must be allowed to move north from that room.

**POST CONDITION:** Movers position north.

**VISIBILITY:** Public

**PROTOTYPE:** int m\_East();

**DESCRIPTION:** Moves position east from where you currently are.

**PRECONDITION:** You must be allowed to move east from that room.

**POST CONDITION:** Movers position east.

**VISIBILITY:** Public

**PROTOTYPE:** int m\_South();

**DESCRIPTION:** Moves position south from where you currently are.

**PRECONDITION:** You must be allowed to move south from that room.

**POST CONDITION:** Movers position south.

**VISIBILITY:** Public

**PROTOTYPE:** int m\_West();

**DESCRIPTION:** Moves position west from where you currently are.

**PRECONDITION:** You must be allowed to move west from that room.

**POST CONDITION:** Movers position west.

**VISIBILITY:** Public

**PROTOTYPE:** void PrintInfo();

**DESCRIPTION:** Prints info about the room so the user can read.

**PRECONDITION:** No applicable.

**POST CONDITION:** No applicable.

**VISIBILITY:** Public

**PROTOTYPE:** int Response(MyString, Room[][5]);

**DESCRIPTION:** Reads what the user inputs and checks where they are in the array to check if the input is allowed and what it does if it is.

**PRECONDITION:** Must be an allowed command.

**POST CONDITION:** Executes what the word does if entered correctly.

**VISIBILITY:** Public

**PROTOTYPE:** void attack (Enemy &)

**DESCRIPTION:** Creates an attack that is set between two classes passing by reference so the damage and health can be changed.

**PRECONDITION:** Has to be able to attack another class that has hp.

**POST CONDITION:** Does random damage based on the damage that is set.

**VISIBILITY:** Public

**Member Variables:**

**Class** Enemy:

**int** *m\_hp* – Integer that stores the health of the class.

**int** *m\_damage –* Integer that stores the damage of the class.

**Class** Hero : Enemy:

**char \*** *m\_name –* Integer that stores the users name and can be called anytime to state their name.

**bool** *m\_weaponM –* Bool to state if they have this weapon or not.

**bool** *m\_weaponS -* Bool to state if they have this weapon or not.

**int** *m\_heroX –* Integer that stores what X the user is at and can be changed to move position.

**int** *m\_heroY -* Integer that stores what Y the user is at and can be changed to move position.

**Class** Room:

**bool** *m\_EDoor –* Bool stating if it has East door or not.

**bool** *m\_WDoor –* Bool stating if it has West door or not.

**bool** *m\_NDoor –* Bool stating if it has North door or not.

**bool** *m\_SDoor –* Bool stating if it has South door or not.

**bool** *m\_Enemyin –* Bool stating if it has an enemy to fight in it or not.

**bool** *m\_Jokein –* Bool stating if it has a death Joke in it or not.

**bool** *m\_Jokein2 –* Bool stating if it has a different death Joke in it or not.

**bool** *m\_Weaponin –* Bool stating if it has weapons in it to take or not.

**bool** *m\_Death –* Bool stating if you die (return 0) as soon as you enter or not.

**bool** *m\_Boss –* Bool stating if it has Boss in the room or not.

**const char \*** *m\_text –* A const char pointer that prints what I type into to describe each room.

CPP Files:

#include<iostream>

#include"Room.h"

#include<time.h>

#include<Windows.h>

Room::Room(bool EastDoor, bool WestDoor, bool NorthDoor, bool SouthDoor, bool Enemyin, bool Jokein, bool Jokein2, bool Weaponin, bool DeathRoom, bool BossRoom, const char \* text)

{

m\_EDoor = EastDoor;

m\_WDoor = WestDoor;

m\_NDoor = NorthDoors;

m\_SDoor = SouthDoor;

m\_Enemyin = Enemyin;

m\_Jokein = Jokein;

m\_Jokein2 = Jokein2;

m\_Weaponin = Weaponin;

m\_Death = DeathRoom;

m\_Boss = BossRoom;

m\_text = text;

}

void Room::PrintInfo()

{

for (int i = 0; m\_text[i] != '\0'; ++i)

{

std::cout << m\_text[i];

Sleep(12);

}

std::cout << std::endl;

};

#include <iostream>

#include<time.h>

#include"Enemy.h"

#include <Windows.h>

#include "Room.h"

Enemy::Enemy(int h, int d)

{

m\_hp = h;

m\_damage = d;

}

Hero::Hero(char \* name, bool weaponM, bool weaponS, int heroX, int heroY)

{

m\_name = name;

m\_weaponM = weaponM;

m\_weaponS = weaponS;

m\_heroX = heroX;

m\_heroY = heroY;

}

int Hero::m\_North()

{

m\_heroY--;

return m\_heroY;

}

int Hero::m\_East()

{

m\_heroX++;

return m\_heroX;

}

int Hero::m\_South()

{

m\_heroY++;

return m\_heroY;

}

int Hero::m\_West()

{

m\_heroX--;

return m\_heroX;

}

void Enemy::attack(Enemy & defender)

{

this->m\_hp -= rand() % defender.m\_damage + 5;

defender.m\_hp -= rand() % this->m\_damage + 5;

/\*

Enemy Boss = Enemy(500, 4);

Enemy User = Enemy(250, 170);

\*/

}

int Hero::Response(MyString A, Room Rooms[][5])

{

if (A.subString("north") == true && Rooms[m\_heroY][m\_heroX].m\_NDoor == true)

{

m\_North();

}

else if (A.subString("east") == true && Rooms[m\_heroY][m\_heroX].m\_EDoor == true)

{

m\_East();

}

else if (A.subString("south") == true && Rooms[m\_heroY][m\_heroX].m\_SDoor == true)

{

m\_South();

}

else if (A.subString("west") == true && Rooms[m\_heroY][m\_heroX].m\_WDoor == true)

{

m\_West();

}

else if (A.subString("help") == true)

{

std::cout << "---------------------------------HELP IS HERE ------------------------------------";

std::cout << "To Move in any direction just type it \n(EXAMPLE: To move north, type 'north') and so forth.. \n \n";

std::cout << "If the room says the doors have locked behind you, they are locked typing \ndirectons will not move you. So do as the game master says. \n \n";

std::cout << "There is no map, create one yourself. \n \n";

std::cout << "Yes, this game is hard however it does not require any skill so you are in luck, loser. \n \n";

std::cout << "---------------------------------HELP IS HERE ------------------------------------ \n \n";

}

else if (A.subString("take") == true)

{

if (A.subString("mace") && Rooms[m\_heroY][m\_heroX].m\_Weaponin == true)

{

std::cout << "Of course... You picked the mace, what a surprise..." << std::endl;

Sleep(2900);

m\_weaponM = true;

Rooms[m\_heroY][m\_heroX].m\_Weaponin = false;

}

else if (A.subString("stick") && Rooms[m\_heroY][m\_heroX].m\_Weaponin == true)

{

std::cout << "GREAT CHOICE! YOU GOT DE FREAKIN STICK FROM DE GODS!!";

Sleep(2900);

m\_weaponS = true;

Rooms[m\_heroY][m\_heroX].m\_Weaponin = false;

}

}

else if (A.subString("attack") && Rooms[m\_heroY][m\_heroX].m\_Enemyin == true)

{

if (m\_weaponM == false && m\_weaponS == false)

{

std::cout << "Yo, how you gona kill this thing with your bare hands... YOU DEAD " << m\_name << std::endl;

Sleep(2500);

return 0;

}

/\*else if (A.subString("north") || A.subString("east") || A.subString("south") || A.subString("west") && m\_weaponM == true || m\_weaponS == true)

{

std::cout << "\*The doors are locked...\* \n \n";

}\*/

else if (m\_weaponM == true)

{

std::cout << "You fought a long battle.. however a Mace can't kill this thing... YOU DEAD." << std::endl;

Sleep(2900);

return 0;

}

else if (m\_weaponS == true)

{

Enemy SmallEnemy = Enemy(100, 10);

Enemy User = Enemy(250, 100);

std::cout << "Cockatrice HP -> [" << SmallEnemy.m\_hp << "]\n \n";

std::cout << "User HP -> [" << User.m\_hp << "]\n \n";

while (SmallEnemy.m\_hp > 0)

{

SmallEnemy.attack(User);

User.attack(SmallEnemy);

std::cout << "Cockatrice Remaining HP ->" << SmallEnemy.m\_hp << "\n \n";

std::cout << "----------------------";

Sleep(550);

std::cout << "User Remaining HP ->" << User.m\_hp << "\n \n";

}

std::cout << "K.O!! REKT!!! OMG THE STICK DESTROYS ANYTHING IT TOUCHES!!" << std::endl;

Sleep(2900);

system("cls");

//std::cout << "Four doors are here go anywhere you please. \n~Which path shall you take?" << std::endl;

Rooms[m\_heroY][m\_heroX].m\_NDoor = true;

Rooms[m\_heroY][m\_heroX].m\_EDoor = true;

Rooms[m\_heroY][m\_heroX].m\_SDoor = true;

Rooms[m\_heroY][m\_heroX].m\_WDoor = true;

Rooms[m\_heroY][m\_heroX].m\_Enemyin = false;

}

}

else if (Rooms[m\_heroY][m\_heroX].m\_Jokein == true)

{

if (A.subString("1") || A.subString("true"))

{

std::cout << "Correct!" << std::endl;

Sleep(2500);

system("cls");

Rooms[m\_heroY][m\_heroX].m\_NDoor = true;

Rooms[m\_heroY][m\_heroX].m\_EDoor = true;

Rooms[m\_heroY][m\_heroX].m\_SDoor = true;

Rooms[m\_heroY][m\_heroX].m\_WDoor = true;

Rooms[m\_heroY][m\_heroX].m\_Jokein = false;

}

else if (A.subString("north") || A.subString("east") || A.subString("south") || A.subString("west"))

{

std::cout << "\*The doors are locked...\* \n \n";

}

else

{

char m\_text[255] = ("Incorrect \n");

for (int i = 0; m\_text[i] != '\0'; ++i)

{

std::cout << m\_text[i];

Sleep(420);

}

Sleep(2500);

return 0;

}

}

else if (Rooms[m\_heroY][m\_heroX].m\_Jokein2 == true)

{

if (A.subString("none") || A.subString("0"))

{

std::cout << "Correct!" << std::endl;

Sleep(2500);

system("cls");

std::cout << "Three doors are here North, East, South, go anywhere you please. \n~Which path shall you take " << m\_name << "?" << std::endl;

Rooms[m\_heroY][m\_heroX].m\_NDoor = true;

Rooms[m\_heroY][m\_heroX].m\_EDoor = true;

Rooms[m\_heroY][m\_heroX].m\_SDoor = true;

Rooms[m\_heroY][m\_heroX].m\_WDoor = true;

Rooms[m\_heroY][m\_heroX].m\_Jokein2 = false;

}

else if (A.subString("north") || A.subString("east") || A.subString("south") || A.subString("west"))

{

std::cout << "\*The doors are locked...\* \n \n";

}

else

{

char m\_text[255] = ("Incorrect \n");

for (int i = 0; m\_text[i] != '\0'; ++i)

{

std::cout << m\_text[i];

Sleep(420);

}

Sleep(2500);

return 0;

}

}

if (Rooms[m\_heroY][m\_heroX].m\_Death == true)

{

std::cout << "RIP " << m\_name << " has fallen. " << std::endl;

Sleep(2900);

return 0;

}

if (Rooms[m\_heroY][m\_heroX].m\_Weaponin == true)

{

std::cout << m\_name << ", There seems to be two weapons of some sort in this room \nA Mace and a Stick ? \nIf you would like one, type Take and then your weapon of choice, if not ignore. \n(EXAMPLE: Take Mace)";

}

if (Rooms[m\_heroY][m\_heroX].m\_Enemyin == true)

{

std::cout << "OMG, a Cockatrice is snarling and staring at you!\n \n\*The doors lock behind you..You must fight\* \n \nType attack... \n";

Rooms[m\_heroY][m\_heroX].m\_NDoor = false;

Rooms[m\_heroY][m\_heroX].m\_EDoor = false;

Rooms[m\_heroY][m\_heroX].m\_SDoor = false;

Rooms[m\_heroY][m\_heroX].m\_WDoor = false;

}

if (Rooms[m\_heroY][m\_heroX].m\_Jokein == true)

{

std::cout << "Oh my " << m\_name << ", you have entered a death room with programming jokes to continue your journey you must answer correctly. \n \n----------------------------------- \nQ: 0 is false and 1 is true, right? \n----------------------------------- \n \n" ;

Rooms[m\_heroY][m\_heroX].m\_NDoor = false;

Rooms[m\_heroY][m\_heroX].m\_EDoor = false;

Rooms[m\_heroY][m\_heroX].m\_SDoor = false;

Rooms[m\_heroY][m\_heroX].m\_WDoor = false;

}

if (Rooms[m\_heroY][m\_heroX].m\_Jokein2 == true)

{

std::cout << "Oh my " << m\_name << ", you have entered a death room with programming jokes to continue your journey you must answer correctly. \n \n----------------------------------- \nQ: How many programmers does it take to change a light bulb? \n \n----------------------------------- \n" << std::endl;

Rooms[m\_heroY][m\_heroX].m\_NDoor = false;

Rooms[m\_heroY][m\_heroX].m\_EDoor = false;

Rooms[m\_heroY][m\_heroX].m\_SDoor = false;

Rooms[m\_heroY][m\_heroX].m\_WDoor = false;

}

if (Rooms[m\_heroY][m\_heroX].m\_Boss == true)

{

if (m\_weaponM == false && m\_weaponS == false)

{

std::cout << "Yo, how you gona kill this thing with your bare hands... YOU DEAD " << m\_name << std::endl;

Sleep(2500);

return 0;

}

else if (m\_weaponM == true)

{

std::cout << " /You encounter the great big huge giant boss called great big huge giant boss\\ \n\*The doors lock behind you..You must fight but wait your God stick, it turned into a long sword... Must be a trap of some sort, walking through that boss door must trigger it. You must fight on your own.\* \n \nType attack to start this epic fight good luck " << m\_name << "... \n \n";

Sleep(2900);

if (A.subString("attack"))

{

std::cout << "You fought a long battle.. however a Mace can't kill this thing... YOU DEAD. \n \nTip Stick > Mace" << std::endl;

Sleep(2900);

return 0;

}

}

else if (m\_weaponS == true)

{

std::cout << " /You encounter the great big huge giant boss called great big huge giant boss\\ \n\*The doors lock behind you..You must fight but wait your God stick, it turned into a long sword... Must be a trap of some sort, walking through that boss door must trigger it. You must fight on your own.\* \n \nType attack to start this epic fight good luck " << m\_name << "... \n \n";

Rooms[m\_heroY][m\_heroX].m\_NDoor = false;

Rooms[m\_heroY][m\_heroX].m\_EDoor = false;

Rooms[m\_heroY][m\_heroX].m\_SDoor = false;

Rooms[m\_heroY][m\_heroX].m\_WDoor = false;

if (A.subString("attack"))

{

Enemy Boss = Enemy(500, 10);

Enemy User = Enemy(250, 100);

std::cout << "Boss HP -> [" << Boss.m\_hp << "]\n \n";

std::cout << "User HP -> [" << User.m\_hp << "]\n \n";

while (Boss.m\_hp > 0)

{

Boss.attack(User);

User.attack(Boss);

std::cout << "Boss Remaining HP ->" << Boss.m\_hp << "\n";

std::cout << "----------------------";

Sleep(550);

std::cout << "User Remaining HP ->" << User.m\_hp << "\n \n";

}

{

Sleep(3500);

system("cls");

std::cout << "Victory!! \n \nThe Exit the Boss was blocking is now slowly opening, revealing dense trees in a rainforest... It seems like you are free. \n \n";

Sleep(2900);

return 0;

}

}

}

}

return -1;

}

**-Read Me -**

You can access this file via this link: https://github.com/wdonray/Text-Base-Adventure-

Click the button that says “Clone or download” button listed on the top right in this page, then click “Download ZIP” this will allow you to access each file individually.

The String-Class folder contains the sources and headers for my game.

The TBAG zipped folder contains my .exe if you would like to just play the game.

If you would like to access the files without downloading the ZIP, instructions are below.

An executable to start the program and test it is also enclosed in the repository.

To access this executable click on the file named “TBAG.zip” and download that file. Its next to History at top right of screen.

An Assessment documentation is enclosed in the repository.

To access Assessment documentation which is on Word, click on the file named “Assessment 1 TBAG.docx” and also download that file from there.

Once you run the program follow the instructions that are on the screen.

Settings I used for my console:

**Screen Buffer Size**

Width: 82

Height: 300

**Windows Size**

Width: 82

Height: 37

**Window Position**

Let System position window

**Font**

Lucida Console size 18

**Colors**

Blue: 128