**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**: getLength (No arguments)

**Description**: Returns the instance of m\_length that was set in MyString.

**Precondition**: None

**Postcondition**:

**Visibility**: Public

* **Prototype**: index (int input)

**Description**: Takes in the char m\_Data with paramater input then returns it.

**Precondition**: Must have an index

**Postcondition**:

**Visibility**: Public

* **Prototype**: compare (MyString str)

**Description**: Compares each character that is inside of each string to check if they are equal.

**Precondition**: Must have two strings

**Postcondition**:

**Visibility**: Public

* **Prototype**: append (MyString str)

**Description**: Appends MyString to paramaters that are passed in Main.cpp.

**Precondition**: Must have two strings

**Postcondition**:

**Visibility**: Public

* **Prototype**: prepend (MyString str)

**Description**: Prepends "Front -> " which is passed in my Main.cpp on to the previous appended word.

**Precondition**: Must have two strings

**Postcondition**:

**Visibility**: Public

* **Prototype**: ToUpper (No arguments)

**Description**: Uppercases every character in the prepended word above.

**Precondition**: None

**Postcondition**:

**Visibility**: Public

* **Prototype:** ToLower (No arguments)

**Description:** Lowercases every character in the prepended word above.

**Precondition:** None

**Postcondition:**

**Visibility:** Public

* **Prototype**: findSubString (char \* sub)

**Description**: Using a bool, search for a certain character inside of the string if found return true, if not return false.

**Precondition**: Must be passed a char \*

**Postcondition**:

**Visibility**: Public

* **Prototype**: findSubStringIndex (int input, char \* sub)

**Description**: Using a bool, goes to a pre set index and searches for a character if found return true, if not return false.

**Precondition**: Must be passed an input and char \*

**Postcondition**:

**Visibility**: Public

* **Prototype**: constCStyle (No arguments)

**Description**: Sets m\_Data to now equal a const char \* then return that variable that was set to the const char \*.

**Precondition**: None

**Postcondition**:

**Visibility**: Public

* **Prototype**: Replace (char \* mine, char \* sub)

**Description**: Searches the string for a preset character which is set in parameters in the Main.cpp then swaps in with another character that is also in the same parameter.

**Precondition**: Must be passed two char \*

**Postcondition**:

**Visibility**: Public

**Member Variables:**

**char** *m\_Data [255]* – A character array that stores the string.

**int** *m\_length* – Integer that stores the length of the string.

CPP File:

#include<iostream>

#include"MyString.h"

#include <time.h>

#include "Room.h"

#include "Enemy.h"

#include <Windows.h>

int main()

{

srand(time(NULL));

bool DidYouWin = false;

std::cout << " --------------------------- \n !Welcome to Bright Souls 3! \n --------------------------- \n \nThe map is hidden if you would like a map create it yourself (By Playing a lot) \n \n~RULES: To Move in any direction just type it \n(EXAMPLE: To move north, type 'north') and so forth...\nWARNING: If you are impatient and just spam Enter, game will close! \n \n \*Press 'E' to enter the game or 'Q' to quit\*" << std::endl;

char go;

std::cin >> go;

std::cin.ignore();

while (go != 'Q' && go != 'E')

{

std::cout << "Try again... Make sure the 'E' is capitalized \n";

std::cin >> go;

std::cin.ignore();

}

if (go == 'E')

{

system("cls");

std::cout << "You fool... \*Sighhh\* Now what is your name? \n-->";

}

else if (go == 'Q')

{

return 0;

}

//Setting a username that I can call the variable m\_name anywhere

char m\_name[255];

std::cin.getline(m\_name, 255);

Hero User = Hero(m\_name, 0, 0, 0, 0);

system("cls");

std::cout << "Hello " << m\_name << ", and welcome to your despair." << "\n \n";

Sleep(800);

std::cout << "You were thrown into this death trap for no reason. You must find the exit in \norder to live... Enjoy." << "\n \n";

int CountDown = 5;

std::cout << "Countdown until game starts: ";

//A simple countdown

while (CountDown > 0)

{

Sleep(1100);

std::cout << CountDown << " ";

CountDown--;

}

system("cls");

//Creating each individual rooms using bools to decide what they have and do not have

Room Room01 = Room{ 1,0,0,1,0,0,0,0,0,0, "Only two doors are in this room, you may go East and South. \n~Which path will you take?" };

Room Room02 = Room{ 1,1,0,1,0,0,0,0,0,0, "The room is empty... For now. \n \nThree doors are in this room East, South, and West. \n~Which path will you take?" };

Room Room03 = Room{ 1,1,0,1,0,0,0,0,0,0, "Three doors are in this room East, South, and West. \n~Which path will you take?" };

Room Room04 = Room{ 1,1,0,1,0,0,0,0,0,0, " -- WARNING -- WARNING -- \n \nThree doors are in this room East, South, and West. \n~Which path will you take?" };

Room Room05 = Room{ 1,0,0,1,0,0,0,0,1,0, "" };

Room Room06 = Room{ 1,0,1,1,0,0,0,0,0,0, "Three doors are in this room North, East, South. \n~Which path will you take nerd?" };

Room Room07 = Room{ 1,1,1,1,0,0,0,1,0,0, "\nFour doors are here go anywhere you please. \n~Which path shall you take? \n" };

Room Room08 = Room{ 1,1,1,1,1,0,0,0,0,0, "\n Four doors are here go anywhere you please. \n~Which path shall you take?" };

Room Room09 = Room{ 1,1,1,1,0,1,0,0,0,0, "Four doors are here go anywhere you please. \n~Which path shall you take?" };

Room Room10 = Room{ 0,1,1,1,0,0,0,0,0,0, " -- WARNING -- WARNING -- \n \nThree doors are in this room North, West, and South. \n~Which path will you take?" };

Room Room11 = Room{ 1,0,1,1,0,0,0,0,0,0, "Three doors are in this room North, East, South. \n~Which path will you take?" };

Room Room12 = Room{ 1,1,1,1,0,0,0,0,0,0, "Four doors are here go anywhere you please. \*Giggles\* \n~Which path shall you take?" };

Room Room13 = Room{ 1,1,1,1,0,0,0,0,1,0, "Want to hear a joke? \n" };

Room Room14 = Room{ 1,1,1,1,0,0,0,0,0,0, "Four doors are here go anywhere you please. \*Giggles\* \n~Which path shall you take?" };

Room Room15 = Room{ 0,1,1,1,0,0,0,0,0,0, "Three doors are in this room North, West, and South. \n~Which path will you take?" };

Room Room16 = Room{ 1,0,1,1,0,0,1,0,0,0, " -- WARNING -- WARNING -- " };

Room Room17 = Room{ 1,1,1,1,1,0,0,0,0,0, "\n Four doors are here go anywhere you please. \n~Which path shall you take?" };

Room Room18 = Room{ 1,1,1,1,0,0,0,0,0,0, "Four doors are here go anywhere you please. \*Giggles\* \n~Which path shall you take?" };

Room Room19 = Room{ 1,1,1,1,0,0,0,0,0,0, "Four doors are here go anywhere you please. \n~Which path shall you take?" };

Room Room20 = Room{ 0,1,1,1,0,0,0,0,0,0, "Three doors are in this room North, West, and South. \n~Which path will you take?" };

Room Room21 = Room{ 1,0,0,1,0,0,0,0,1,0, "" };

Room Room22 = Room{ 1,1,1,0,0,0,0,0,0,0, " -- WARNING -- WARNING -- \n \nThree doors are in this room, you may go West, North, and East. \n~Which path will you take?" };

Room Room23 = Room{ 1,1,1,0,0,0,0,0,0,0, "Three doors are in this room, you may go West, North, and East. \n~Which path will you take?" };

Room Room24 = Room{ 1,1,1,0,0,0,0,0,0,0, "Three doors are in this room, you may go West, North, and East. \n~Which path will you take?" };

Room Room25 = Room{ 0,0,0,0,0,0,0,0,0,1, "" };

Room Rooms[5][5] =

{

{ Room01,Room02,Room03,Room04,Room05 },

{ Room06,Room07,Room08,Room09,Room10 },

{ Room11,Room12,Room13,Room14,Room15 },

{ Room16,Room17,Room18,Room19,Room20 },

{ Room21,Room22,Room23,Room24,Room25 }

};

while (DidYouWin == false)

{

std::cout << std::endl;

Rooms[User.m\_heroY][User.m\_heroX].PrintInfo();

char Response[255];

std::cout << "\n->";

std::cin.getline(Response, 255);

system("cls");

MyString PResponse = MyString(Response);

//If the user presses Enter key without typing anything in game will close used this to fix a few errors

if (PResponse.getLength() == 0)

{

break;

}

PResponse.ToLower();

if (User.Response(PResponse, Rooms) == 0)

{

break;

}

}

system("pause");

return 0;

}

**-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.