CMSC 202 Fall 2019

Project 2 – Wizards Unite

**Assignment:** Project 2 – Wizards Unite

**Due Date:** Thursday, October 10th at 8:59pm

**Value:** 80 points

# Overview

In this project, you will:

* Practice basic C++ syntax including branching structures
* Write classes and instantiate those classes using a constructor
* Use arrays to hold objects
* Use simple file input
* Practice breaking projects into multiple files
* Use a makefile to compile a project

# Background

Harry Potter: Wizards Unite is an augmented reality game released to iOS and Android in June 2019. Like Pokémon Go, Wizards Unite follows an Auror, Professor, or Magizoologist wandering around fighting against mythical beasts from the Harry Potter and Fantastic Beasts franchises.

For this project, you will be required to create a text-based version of this game. You may see a videos and various details of the game online here: <https://www.harrypotterwizardsunite.com/>

In this implementation of Wizards Unite, you will be implementing the role of the Wizard. For our purposes, the Wizard will be gathering Foundables which are a variety of people and artifacts.

# Assignment Description

Initially, you will need to read in a list of Foundables from a file and load them into an array. The list of Foundables is static, and you can assume that the size can be stored in a constant. The Wizard starts by having collected no Foundables although the goal is to collect all the Foundables in the game.

Additionally, this project has an additional extra credit opportunity where you implement the project where the Wizard’s array holds nothing but pointers.

The goal of the game is to collect all unique Foundables available across 5 different rarities. The rarer the Foundable, the higher the toughness!

# Requirements:

This is a list of the requirements of this application. For this project, you will be provided with header files to start you in the right direction. For you to earn all the points, however, you will need to meet all the defined requirements.

* You must follow the coding standard as defined in the CMSC 202 coding standards (found on Blackboard under course materials). This includes comments as required.
* The project must be turned in on time by the deadline listed above.
* The project must be completed in C++. You may not use any libraries or data structures that we have not learned in class. Libraries we have learned include **<iostream>, <fstream>, <iomanip>, ~~<vector>~~, <cmath>, <ctime>, <cstdlib>,** and **<string>**. You may use [**stod**](http://www.cplusplus.com/reference/string/stod/) and [**stoi**](http://www.cplusplus.com/reference/string/stoi/) in this project to convert strings to doubles or strings to integers. You may not use vectors – everything must be implemented in arrays. You should only use **namespace std**.
* Using the provided files, **Foundable.h, Wizard.h**, **Game.h**, **makefile**, **proj2.txt** and **proj2.cpp**, create Wizards Unite. You can copy the files from my directory in **/afs/umbc.edu/users/j/d/jdixon/pub/cs202/proj2**.
* To copy them, navigate to your project 2 folder and type:

**cp /afs/umbc.edu/users/j/d/jdixon/pub/cs202/proj2/\* .**

* You **must** use the provided header files (**Foundable.h, Wizard.h** and **Game.h**). You may only add HELPER functions and global constants to these files. Do not add member variables to any class.
* All user input must be validated. For example, if a menu allows for 1, 2, or 3 to be entered and the user enters a 4, it will re-prompt the user. However, the user is expected to always enter the correct data type. i.e. If the user is asked to enter an integer, they will. If they are asked to enter a character, they will. You do not need to worry about checking for correct data types.
* There is a single input files for this project named, “**proj2.txt**”. The file name can be stored as a constant. The file is already provided in Prof. Dixon’s course folder on GL.
* For the basic version of this project, pointers are not required, however, pass-by-reference may be required. For the extra credit version, you must use pointers for the Wizard’s array of Foundables. The extra credit details are below.
* The player’s name can have a space (use **getline**). Additionally, the names of the Foundables may have a space.
* A Wizard starts with a toughness somewhere between 150 and 300. There are constants in Wizard.h to help **MIN\_START\_TOUGH** and **MAX\_START\_TOUGH**.
* A Foundable is defeated when the Wizard has a higher toughness. Ties go to the Foundable. The game continues if the Wizard is defeated but that Foundable will still need defeating in order to win the game.
* A Wizard goes up in level when they successfully defeat another Foundable.
* The game is won when all Foundables have been defeated.
* When a Wizard goes up in level, their toughness increases from 0 – 99. There is a constant in Wizard.h for this called **LEVEL\_TOUGH\_INCREASE**
* Have a main menu that asks if the user wants to:
  + Display information about the Wizard including name, level, toughness, wins, losses, win rate (**2 decimal place percentage**) and number of Foundables found.
  + Display defeated Foundables in a numbered list.
  + Display all Foundables in the game in a numbered list.
  + Attack Foundables

*Asks which rarity of the Foundable you would like to attack. Each rarity generally has a higher toughness.*

* + - When attacking, the Foundable is randomly chosen from the list of all Foundables. There are two rules though: Each Foundable can only be defeated once and the Foundable fought will always be the rarity chosen (**unless all Foundables of that rarity have been defeated**).

*If the Wizard wins, the Wizard win count is incremented. If the Wizard loses, the Wizard loss count is incremented. The win rate is displayed as a percentage (with exactly 2 decimal places). HINT:* ***cout << noshowpoint****; will reset to no longer show two decimal places exactly.*

*Ties are considered a loss for the Wizard.*

* + - **If all Foundables *of a specific rarity* have been defeated, then the game tells the user that has happened.**
  + Exit

# Recommendations

You must use the provided header files (**Foundable.h, Wizard.h,** and **Game.h**) additionally, we provided you with the **makefile** and the **proj2.cpp**.

Here are some general implementation recommendations (do not follow these verbatim – these are GENERAL suggestions):

* + Read each of the header files in detail. Use paper to take notes. Each header file has tons of comments.
  + Design the solution (part is already designed, so make sure you understand how it works) **on PAPER**.
  + Start with the **Foundable.cpp** file and code everything. Don’t forget the constructor! Test everything incrementally. You can use the **makefile** to compile just **Foundable.o** to test it.
  + Start **Wizard.cpp** – start building the constructor. Incrementally build it as you are testing a function.
  + Once **Wizard.cpp** is written, start on **Game.cpp** – start with loading in the Foundables from the **proj2.txt** file. This will be tricky because of how the data is formatted. As a reminder, you can use **stoi** and **stod** to convert from strings to ints or doubles.
  + **AttackFoundable() and CheckFoundable** are a bit tricky in Wizard. **AttackFoundable()**in Game is a bit tricky.
  + After the files are successfully loaded (and displayed) work on the main menu.

# (Optional) Extra Credit

There is an option for extra credit in this project worth 5 points. **You may only attempt this after you have successfully completed the version described above.**

For the standard implementation of this project, you are copying foundables from the **m\_allFoundables** array in the **Game** class into the **m\_foundables** in the **Wizard** class. This is inefficient as you are using memory twice. A better implementation would have the Foundables in **m\_foundables** in the Wizard class to use pointers to point back to the **m\_allFoundables** array so that there is no data duplication. After you have implemented the basic version, you may attempt to implement the version using pointers. You may reuse any functions you have written for the basic version in the version with pointers. The output of the assigned version and the version using pointers should be identical.

To be clear, the assigned version of this project is worth 80 points, the extra credit is only worth 5. You must complete the assigned version before attempting the extra credit. If you do not complete the assigned version, we will not grade the extra credit. You may not turn in the extra credit late.

One hint: if you need to dereference a pointer and use the dot operator, you should use the arrow operator instead. For example, instead of using **\*m\_foundables[0].m\_name** you would use:  
 **m\_foundables[0]->m\_name**

To download the extra credit version of the project, create a new folder in your project 2 folder named **extra\_credit**. You must modify your base code so that all Foundables used in Wizard are pointers. The only place data about a Foundable is stored in is **m\_allFoundables** in **Game.cpp**.

To turn in the extra credit, create an **extra\_credit** folder in your submission folder:

**mkdir ~/cs202proj/proj2/extra\_credit**

Then submit your **extra\_credit** files using this command:

**cp Foundable.h Foundable.cpp Wizard.h Wizard.cpp Game.h Game.cpp proj2.cpp ~/cs202proj/proj2/extra\_credit**

# Sample Input and Output

For this project, the input file is comma delimited. The only input file is called **proj2.txt**. You can code the file name as a constant. HINT: Look at the documentation for [getline](http://www.cplusplus.com/reference/istream/istream/getline/) when it is used with an input stream. It has a third parameter named “delimiter”. You may need to convert **strings** to **ints** or **doubles**.

The columns for **proj2.txt** are as follows: Foundable name, Foundable type, Foundable rarity and Foundable toughness. The list below is only a partial list of Foundables from the data file.

|  |
| --- |
| Albino Niffler,Mysteries,5,1460  Ancient Tome,Mysteries,1,250  Ashwinder,Mysteries,4,1608  Cerberus,Mysteries,5,1110  Daily Prophet,Mysteries,2,584  Hermione Granger,Mysteries,5,575 |

The file can be downloaded from Prof. Dixon’s data folder by navigating to your project 2 folder and typing the following command:

**cp /afs/umbc.edu/users/j/d/jdixon/pub/cs202/proj2/proj2\_\* .**

After you copy the data file, you can type “**cat proj2.txt**” and it should show you the entire data file.

In the sample output below, user input is colored blue for clarity. After compiling and running proj2, the output would look like this:

|  |
| --- |
| [jdixon@linux2 proj2]$ make run  ./proj2  Welcome to Harry Potter: Wizards Unite  192 Foundables loaded  What is your wizard's name?  JD  What would you like to do?  1. Wizard Information  2. List My Foundables  3. List All Foundables  4. Attack Foundable  5. Quit |

If you were to display Wizard’s Information, it would look like this:

|  |
| --- |
| What would you like to do?  1. Wizard Information  2. List My Foundables  3. List All Foundables  4. Attack Foundable  5. Quit  1  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  Information About Wizard JD  Level: 1  Toughness: 257  Foundables Found: 0  Wins: 0 Losses: 0  Win Rate: 0%  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  What would you like to do?  1. Wizard Information  2. List My Foundables  3. List All Foundables  4. Attack Foundable  5. Quit |

If the user would choose 2, List My Foundables (before defeating any), then the output would look like this:

|  |
| --- |
| What would you like to do?  1. Wizard Information  2. List My Foundables  3. List All Foundables  4. Attack Foundable  5. Quit  2  You have not defeated any foundables yet.  What would you like to do?  1. Wizard Information  2. List My Foundables  3. List All Foundables  4. Attack Foundable  5. Quit |

If the user would choose 3, and “List All Foundables” then the output would look like this:

|  |
| --- |
| What would you like to do?  1. Wizard Information  2. List My Foundables  3. List All Foundables  4. Attack Foundable  5. Quit  3  1. Albino Niffler| Mysteries| 5| 1460  2. Ancient Tome| Mysteries| 1| 250  3. Ashwinder| Mysteries| 4| 1608  4. Cerberus| Mysteries| 5| 1110  .\*\*OMMITTED FOR SPACE\*\*  189. Tom Riddle| Dark Arts| 2| 218  190. Tom Riddle Sr.'s Gravestone| Dark Arts| 3| 810  191. Vanishing Cabinet| Dark Arts| 2| 244  192. Wanted Poster of an Azkaban Escapee| Dark Arts| 2| 564  What would you like to do?  1. Wizard Information  2. List My Foundables  3. List All Foundables  4. Attack Foundable  5. Quit |

If we try to defeat a Foundable then we choose 4. Attack Foundable. Then the game should prompt the user for a rarity of Foundable to attack. The specific Foundable you attack is random, however, it will always be unique and of the rarity chosen.

|  |
| --- |
| What would you like to do?  1. Wizard Information  2. List My Foundables  3. List All Foundables  4. Attack Foundable  5. Quit  4  What rarity of Foundable would you like to challenge (1-5)?  1  The Wizard JD attacks Ancient Tome!!!  You defeat the Ancient Tome!!  What would you like to do?  1. Wizard Information  2. List My Foundables  3. List All Foundables  4. Attack Foundable  5. Quit |

Now if we go and look at “List My Foundables” then the new Foundable should be listed.

|  |
| --- |
| 1. Wizard Information  2. List My Foundables  3. List All Foundables  4. Attack Foundable  5. Quit  2  1. Ancient Tome| Mysteries| 1| 250  What would you like to do?  1. Wizard Information  2. List My Foundables  3. List All Foundables  4. Attack Foundable  5. Quit |

After all of the Foundables of a specific rarity have been found, it will look like this:

|  |
| --- |
| What rarity of foundable would you like to challenge (1-5)?  4  The Wizard JD attacks Young Acromantula!!!  You defeat the Young Acromantula!!  What would you like to do?  1. Wizard Information  2. List My Foundables  3. List All Foundables  4. Attack Foundable  5. Quit  4  What rarity of foundable would you like to challenge (1-5)?  4  You have found all Foundables at that level of rarity!  What would you like to do?  1. Wizard Information  2. List My Foundables  3. List All Foundables  4. Attack Foundable  5. Quit  4 |

Here are some example runs where additional input validation is being shown and where the name of the Wizard has a space:

|  |
| --- |
| ./proj2  Welcome to Harry Potter: Wizards Unite  192 Foundables loaded  What is your wizard's name?  Michael Jackson  What would you like to do?  1. Wizard Information  2. List My Foundables  3. List All Foundables  4. Attack Foundable  5. Quit  4  What rarity of foundable would you like to challenge (1-5)?  0  What rarity of foundable would you like to challenge (1-5)?  6  What rarity of foundable would you like to challenge (1-5)?  5  The Wizard Michael Jackson attacks Albus Dumbledore!!!  You do not successfully defeat the foundable  What would you like to do?  1. Wizard Information  2. List My Foundables  3. List All Foundables  4. Attack Foundable  5. Quit |

# Compiling and Running

We have provided you with a sample **makefile** which should help you compile all of the classes and the program itself.

Once you have compiled using the provided **makefile**, enter the command **make run** or **./proj2** to run your program. If your executable is not proj2, you will lose points. It should look like the sample output provided above.

# Completing your Project

When you have completed your project, you can copy it into the submission folder. You can copy your files into the submission folder as many times as you like (before the due date). We will only grade what is in your submission folder.

For this project, you should submit the following files to the **proj2** subdirectory:

**Foundable.cpp, Foundable.h**

**Wizard.cpp, Wizard.h**

**Game.h, Game.cpp , proj2.cpp**

For this project, you should only modify the header files to add helper functions and additional constants but do not add any additional variables.

You do not need to submit the **makefile**.

As you should have already set up your symbolic link for this class, you can just copy your files listed above to the submission folder.

* 1. cd to your project 2 folder. An example might be cd **~/202/projects/proj2**
  2. **cp Foundable.h Foundable.cpp Wizard.h Wizard.cpp Game.h Game.cpp proj2.cpp ~/cs202proj/proj2**

You can check to make sure that your files were successfully copied over to the submission directory by entering the command

ls ~/cs202proj/proj2

Make sure that the required files are submitted by the deadline. If the copy command provided does not work, it is your responsibility to figure out what is wrong and that all required files have been submitted.

You can check that your program compiles and runs in the **proj2** directory, but please clean up any **.o** and executable files. Again, do not develop your code in this directory and you should not have the only copy of your program here. Uploading of any **.gch** files will result in a severe penalty.

**IMPORTANT:** If you want to submit the project late (after the due date), you will need to copy your files to the appropriate late folder. If you can no longer copy the files into the proj2 folder, it is because the due date has passed. You should be able to see your proj2 files but you can no longer edit or copy the files in to your proj2 folder. (They will be read only)

* If it is 0-24 hours late, copy your files to **~/cs202proj/proj2-late1**
* If it is 24-48 hours late, copy your files to **~/cs202proj/proj2-late2**
* If it is after 48 hours late, it is too late to be submitted.