EEE101 Assessment 3 Report

Yimian Liu 1717608

December 15, 2018

1. Introduction

In the Assessment 2 of EEE101, we had developed a simple but challenging scissor-rock-paper game program and there were several functions it can achieve such as creating a user, play a game, manage user data online and play a background music. Aiming to build a powerful and user-friendly software, however, last time I did not have enough time to add more interesting modules to the game due to time limit. Thankfully assessment 3 gives me the opportunity to further enrich my game and put my several ideas into practice.

This report will try to provide a brief but clear perspective concerning what the new game elements are and how they have been figured, designed and developed step by step with the method of Software Development Process (SDP).

2. Problem Statement

2.1 Overall Scope

The most significant problem is that we need to add some new parts basing on an existed game structure. This would be quite challenging because the former game contains nearly 7000 lines code and any uncareful change on it would cause some of its former functions crashing. Therefore, any added code needs to be fully considered before writing. Thanks to that most of the former functions were packed into functions, it looks possible and more convenient to do some of the addition flexibly.

The second issue is that because this game was designed to be an online program, any change on the function that interacts with the server should be extremely careful. We need to make sure once a user who uses a former game vision want to update his or her local data, the program and the server can give an expected response.

2.2 Password Section

This is one of the requirements of assessment 3. In this scope, each account should have its own and unique password. Besides, security strategies such as how to prevent a violent crack behavior should also be considered. Then, several requirements had been proposed in the following list.

- The password should be limited in 3 to 30 words.
- The password should only contain English, numbers and symbols.
- When creating a password, the new password should be type for two times for verification.
- When a password is typing in, it should not be shown on the screen.

- File which storages password should be encoded.
- The password can only be changed by the one who created it.
- One account should only have one password.
- Every account should have a password.
- When the password had been wrong typed for more than 5 times, the program exits.
- There should be an approach for user to go back from the password screen.

2.3 History Section

Assessment 3 require to set up a history section which mainly contains the number of rounds, play wins, computer wins, draws, overall win or lose. The user should get the privilege to review his or her history, and clean his or her history. The data should be stored in files.

However, in assessment 2, I had established a data system which functions to record and extract data. Unfortunately, these data only contain what the user and the computer choose in each round of the game while not mark the start and the end of the game. This means that I need to add these new functions such as showing the result of each game on a exist data structure. Appear these seems to be the most challenge part of this section.

The detailed requirement of history is classified as the following list.

- Display the number of rounds, player and computer wins times, draws times and overall result of a game.
- Present detailed game behavior for each round.
- There should be an overall win rate.
- The data system should record what user and computer choose for each round of game.
- The data system should record the game start.
- The data system should store history data in files.
- The data system should extract the history from data file.
- Only the account own can access his or her detailed history.
- Any users can access the overall win rates of any account. (in Rank Section)
- The account owner can clean his or her own history.

3. Analysis

3.1 About Global Variables

Firstly, I want to state that there are already six global variables due to historical reason. These global variables are designed and put into practice in assessment 2. Among them, one is used to record user name, two of them record the game wins and game times, one records overall win rate, on control the background music and one record the rows of returned two-dimension array. Most of these global was used to substitute some functions of structure which I did not familiar with at that time. However, today it seems to be extremely difficult for me to remove these global variables since I have no idea what would happen if I remove some of them. This taught me a lesson that never trying to use a global variable which can greatly destroy function's encapsulation and make it nearly impossible to change an existed program! This time, I haven't added any new global variable in assessment 3.

3.2 About Macro Definition

In assessment 2, the macro definition had not been used since I did not realize its important at that time. However, this time I begin try to add some macro definition in assessment 3. A macro definition can help to make the program more modifiable, make it easier to change specific values in great number of functions. Basing on this time's understanding, I will attempt to make full use of macro definition in the approaching assessment 4.

3.3 Password Section

User Structure

To extend the former single username account to the architecture that one username with a password, I design a structure called user which contains two string variable named 'name' and 'passwd' to storage username and password.

Input and Output

To achieve that when user input his or her password, the screen not screen them, I decide to directly detect the keyboard event instead of using 'scanf()' or 'gets()'. I found a function called '_getch()' in <conio.h> can make the keyboard detection come true.

3.4 History Section

Data Structure

Because the history section is about the data, and the data is mainly store in the files. However, due to historical reason, for each round of games, the data produced are three continuing single number. The first one represents the user's choice, while 0 indicates scissor, 1 indicates rock and 2 indicates paper. The second number shows what the computer chooses, and its formation is the same with the first one. The third value indicate the result of this round for user, while 0 represents lose and 1 represents win. For example, '220' means that in this round both the user and the computer choose paper and the result is that the user not win.

Besides the above standard which set in the assessment 2 era, this time we introduce two new roles. The first one is hat '999' represent the start of a file. This is for the case that when the account currently haves no data. In the former game, if there is no data file for an account, the server will delete the account automatically. After include the '999' role, this problem can be solved. The second one is '888', which marks the beginning of a game. By marking the start of a game, we can then calculate the total rounds, user and computer wins, overall result etc. of a game.

Variables

The variables in history are mainly presented in a two-dimension form. To collaborate with the file data, we design the history data into three columns and use 'malloc()' to allocate corresponded rows. At the end rows we mark it as {0,0,0}.

4. Design

4.1 Password Input Algorithm

When there is an event on keyboard, the return value will be judged for twice. If it is a legal value, it will be added to a temp string variable and the feedback will be shown on the screen with an extra '*'. If the input is a 'enter' or 'esc', the function will return with different operations as it shown in Figure 1. If the input is not in these two types, the program will do nothing and waiting the next key to be input.

This algorithm mainly achieves two objects. The first one is the input password would not shown on the screen. The second one is that the input is also be filtered.

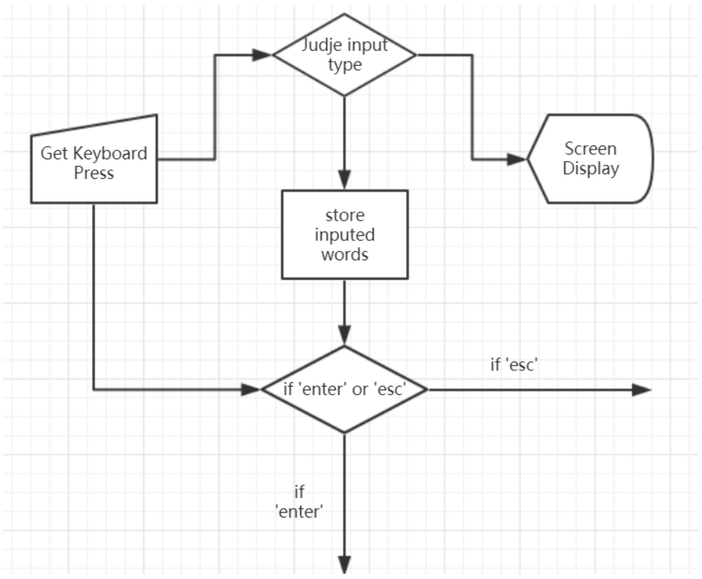


Figure 1 - Password Input Algorithm

4.2 Password Encode Algorithm

Directly storage user password in file is regarded as rude behavior. Before store user's password, we usually use a encode operation to process the user's password. This algorithm provides a way to encode user's password in C.

This algorithm mainly relies on the 'xor' operation, which is represented as '^' in C. As it shown in figure 2, after include a encode key provided by the program, this algorithm can generate a unique key for a username and its password, such as '2YU28gM9'. Then we mixed this string with some random string and store it in file. Then every time when the user login, we only need to use the username and user input password to generate the unique instantly. After check the password file if the key exists then we will know if the password is correct. You can have a look at the password file now to see how powerful this algorithm is through this url: https://cn.yimian.xyz/tmp/as3/psswd.php
You are also recommended to refresh the page to see what will happen!

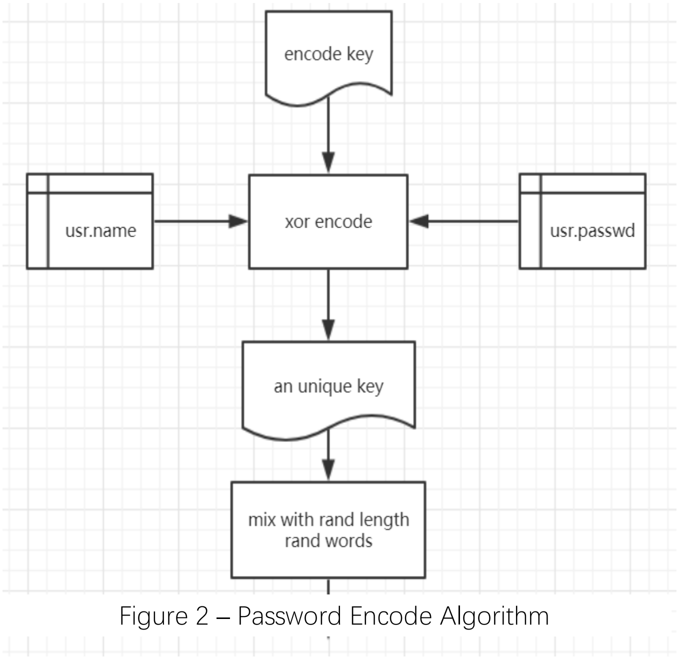


Figure 2 – Password Encode Algorithm

4.3 History Extract Algorithm

This algorithm as it shown in Figure 3 functions to extract user data from storage data structure to diverse data. Unsurprise, statistic such as rounds of game, user and computer wins times, draws times, final winner and overall win rate can be got from this algorithm.

The presentation of this algorithm is several functions which are responsible for nearly any user data process. This algorithm is utilized under many situations such as display user's game history, present a rank, compute overall win rate. After specify this algorithm, it would be easier to design and modify the history section.

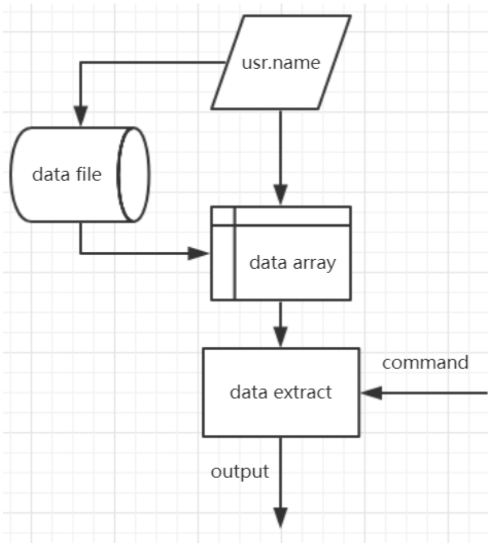


Figure 3 – History Extract Algorithm

4.4 Multithread Process Algorithm

When trying to insert a history data, especially when the insert Process contains an internet connecting behavior, if we still follow a traditional way to execute our program, it would cause a terrible user experience since the program will freeze the screen between the time that the program starts to upload the data and the server confirms the data.

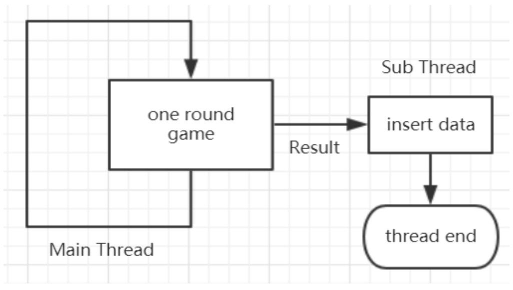


Figure 4 – Multithread Process to Insert Data

To solve this problem, we include the Multithread Process algorithm as shown in Figure 4. When it is time to insert new data, the program will not stop the whole game to wait for the internet, but try to create a new thread to execute this mission. In this case, when the program uploads the data underground, the game will continue the next round without hesitation.

Additionally, this algorithm is also utilized to improve the background music service. Since the method to control the music is using C to arouse and instruct serval vbs script, it would take a long time waiting the vbs script to be executed. In this assessment, I used multithread to optimize this.

5. Implementation

There are seven C file of this assessment. Since I have written some codes that belong to C11, it is recommended to use a complier such as gcc. I have prepared a gcc automatic compile script named compilergcc.bat for help.

1717608_3.c
This file contains main function and several defined function which aims to make to main function easier to read. You can access this file through the following url.
url: https://cn.yimian.xyz/tmp/as3/showcode.php?code=1717608_3.c

winprint.h
The code in this file mainly works to show a windows message box. There are several situations the main function needs to give the user a warning through a message box which is considered to be more conspicuous. You can access this file through the following url.
url: https://cn.yimian.xyz/tmp/as3/showcode.php?code=winprint.h

data.h
This file contains the functions which operate data. You can access this through the following url.
url: https://cn.yimian.xyz/tmp/as3/showcode.php?code=data.h

download.h
This file contains functions that mainly take charge of the interaction with server. It can be access through the following url.
url: https://cn.yimian.xyz/tmp/as3/showcode.php?code=download.h

input.h
This file includes code that control input event. Access through the following url.
url: https://cn.yimian.xyz/tmp/as3/showcode.php?code=input.h

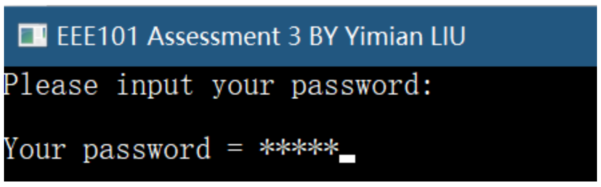
print.h
This file contains codes that conduct the screen display behavior. Access through following url.
url: https://cn.yimian.xyz/tmp/as3/showcode.php?code=print.h

sound.h
This file takes charge of the background music. Access through url.
url: https://cn.yimian.xyz/tmp/as3/showcode.php?code=sound.h

6. Test

6.1 Password Section

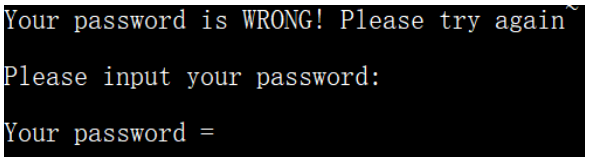
Input a Password
When input a password, the password should not be shown on the screen.
Test result:



```
EEE101 Assessment 3 BY Yimian LIU
Please input your password:

Your password = *****_
```

Figure 5 – Test Result of Input a Password

When the password is wrong, there should be a hint.
Test result:

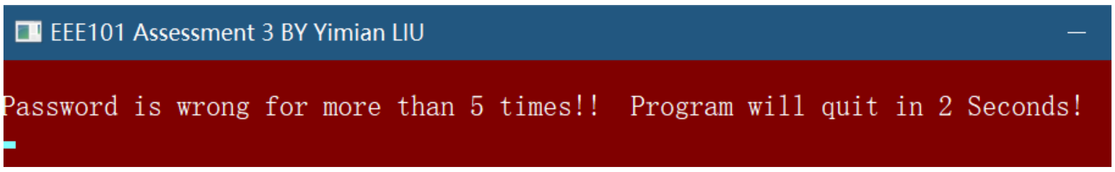


```
Your password is WRONG! Please try again

Please input your password:

Your password =
```

Figure 6 – Test Result of Input a Wrong Password

When the password is wrong for more than five times, there should be a hint and then the program will exit.

Test result:



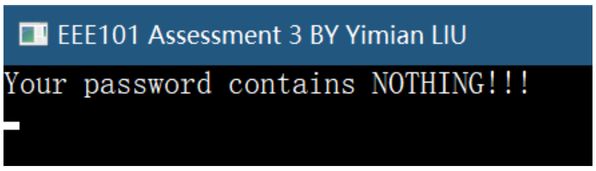
```
EEE101 Assessment 3 BY Yimian LIU
Password is wrong for more than 5 times!!  Program will quit in 2 Seconds!
```

Figure 7 – Test Result of Input Too Many Wrong Password

Create a Password

When creating a password, if input nothing, program should forbid this and give hint.

Test result:

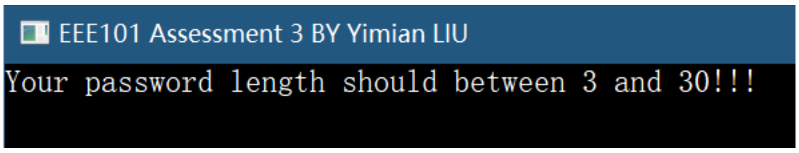


```
EEE101 Assessment 3 BY Yimian LIU
Your password contains NOTHING!!!
```

Figure 8 – Test Result of Input Nothing

When creating a password, if input less than 3 or more than 30, program should give hint.

Test result:

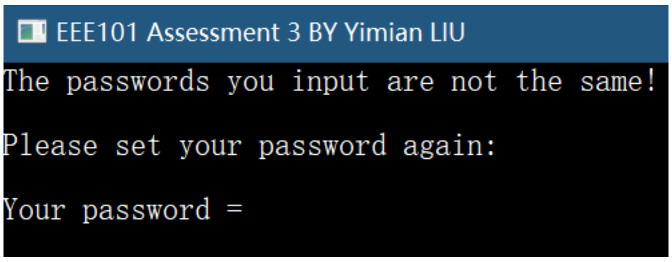


```
EEE101 Assessment 3 BY Yimian LIU
Your password length should between 3 and 30!!!
```

Figure 9 – Test Result of Input Less Than 3 And More Than 30

When creating a password, if the two times input are not the same, program should give hint.

Test result:



```
EEE101 Assessment 3 BY Yimian LIU
The passwords you input are not the same!
Please set your password again:
Your password =
```

Figure 10 – Test Result That Two Input Password Not Same

Change Password

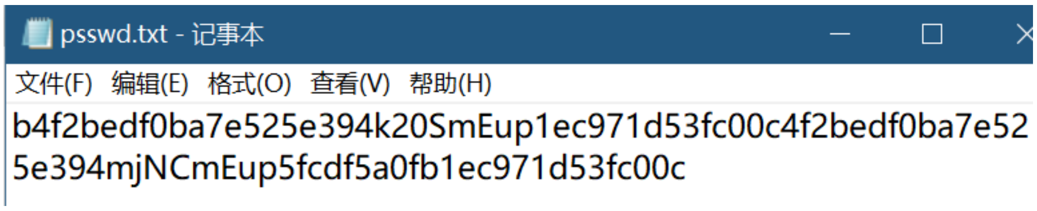
The code of changing a password is nearly the same as the combination of Input a password and Create a password, its test result is shown as the result in the former two part.

Besides, for the former three parts, when press ESC, they all can go back to the former screen.

Encode Password

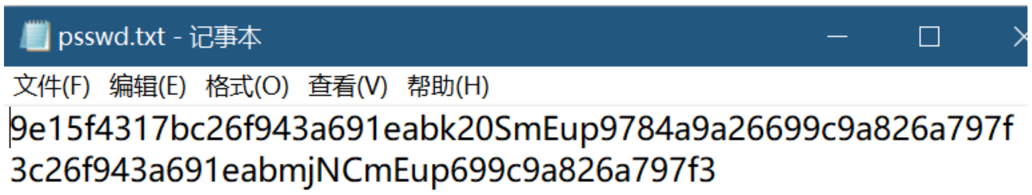
The password in the file should be encoded. And each time when someone open the password file the content should be almost not the same.

Test result:



```
b4f2bedf0ba7e525e394k20SmEup1ec971d53fc00c4f2bedf0ba7e52
5e394mjNCmEup5fcdf5a0fb1ec971d53fc00c
```

Figure 11 – The First Time Open Password File



```
9e15f4317bc26f943a691eabk20SmEup9784a9a26699c9a826a797f
3c26f943a691eabmjNCmEup699c9a826a797f3
```

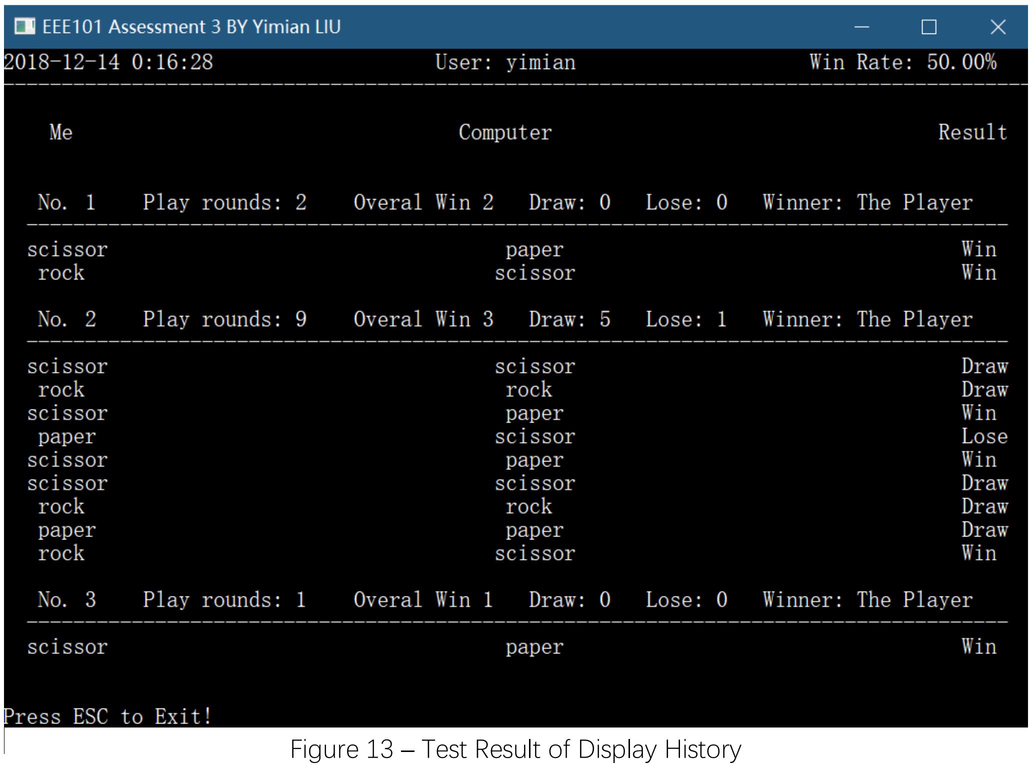
Figure 12 – The Second Time Open Password File

6.2 History Section

Display a History

The program should display a classified history which contains game rounds, player and computer wins times, draws times, final winner and other detailed record.

Test result:



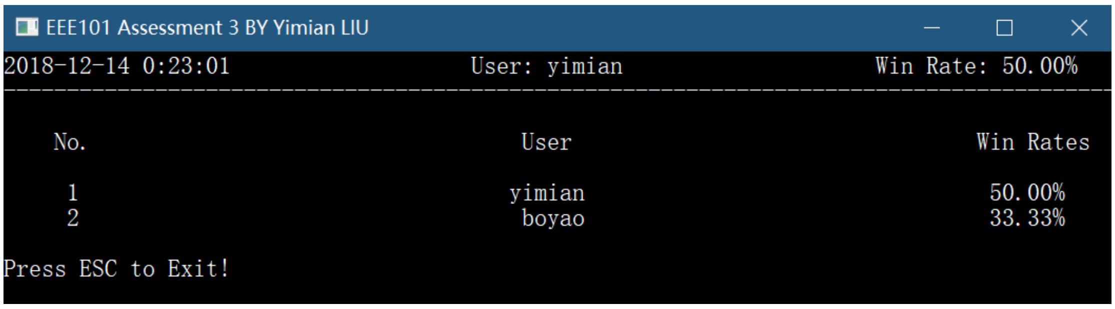
```
2018-12-14 0:16:28                  User: yimian                    Win Rate: 50.00%
--------------------------------------------------------------------------------------

   Me                                 Computer                                  Result

  No. 1    Play rounds: 2    Overal Win 2    Draw: 0    Lose: 0    Winner: The Player
 ------------------------------------------------------------------------------------
 scissor                                 paper                                    Win
  rock                                  scissor                                   Win

  No. 2    Play rounds: 9    Overal Win 3    Draw: 5    Lose: 1    Winner: The Player
 ------------------------------------------------------------------------------------
 scissor                                scissor                                   Draw
  rock                                   rock                                     Draw
 scissor                                 paper                                    Win
  paper                                 scissor                                   Lose
 scissor                                 paper                                    Win
 scissor                                scissor                                   Draw
  rock                                   rock                                     Draw
  paper                                  paper                                    Draw
  rock                                  scissor                                   Win

  No. 3    Play rounds: 1    Overal Win 1    Draw: 0    Lose: 0    Winner: The Player
 ------------------------------------------------------------------------------------
 scissor                                 paper                                    Win


Press ESC to Exit!
```

Figure 13 – Test Result of Display History

Display a Rank

The program should display a rank that shows all users' overall win rate and rank them by order.

Test result:



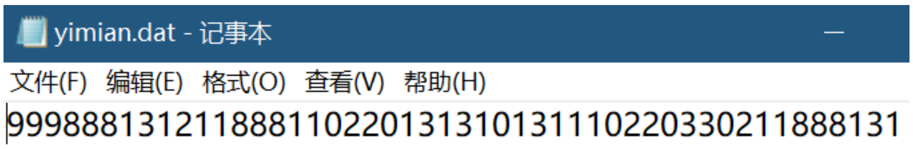
```
2018-12-14 0:23:01                    User: yimian                    Win Rate: 50.00%
--------------------------------------------------------------------------------------

    No.                                  User                                Win Rates

     1                                  yimian                                 50.00%
     2                                   boyao                                 33.33%

Press ESC to Exit!
```

Figure 14 – Test Result of Display a Rank

Check History File

In history file, there should be data such as '999', '888', '220'etc.

Test result:



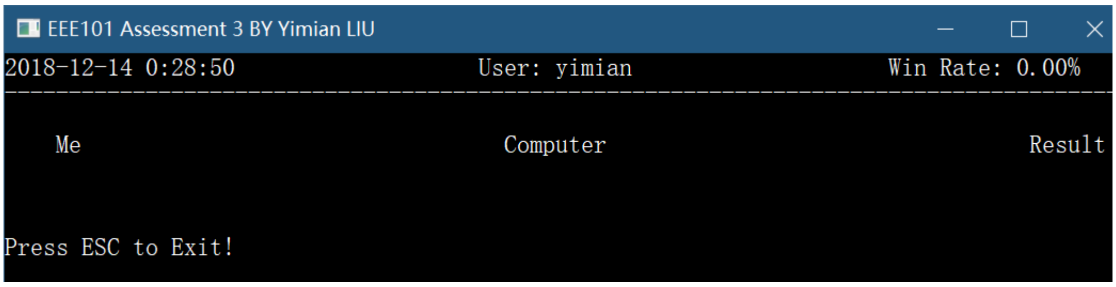
```
9998881312118881102201313101311102203302118881 31
```

Figure 15 – History Data of User yimian in File

Clean History

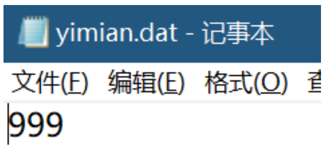
After operating the clean history selection, there should not be any history.

Test result:



```
2018-12-14 0:28:50                    User: yimian                    Win Rate: 0.00%
--------------------------------------------------------------------------------------

    Me                                 Computer                                 Result


Press ESC to Exit!
```

Figure 16 – Test Result of History Display After Cleaning History



```
999
```

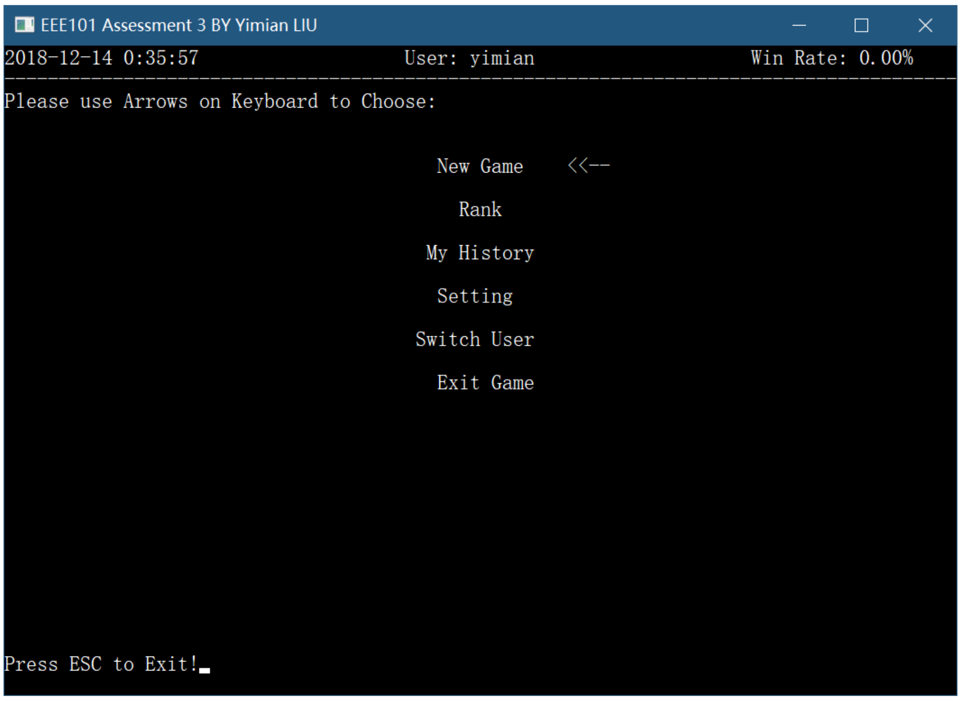
Figure 17 – Test Result of Data File After Cleaning History

As it shown in Figure 16, there is only a '999' which marks the start of a data file.

Main Menu

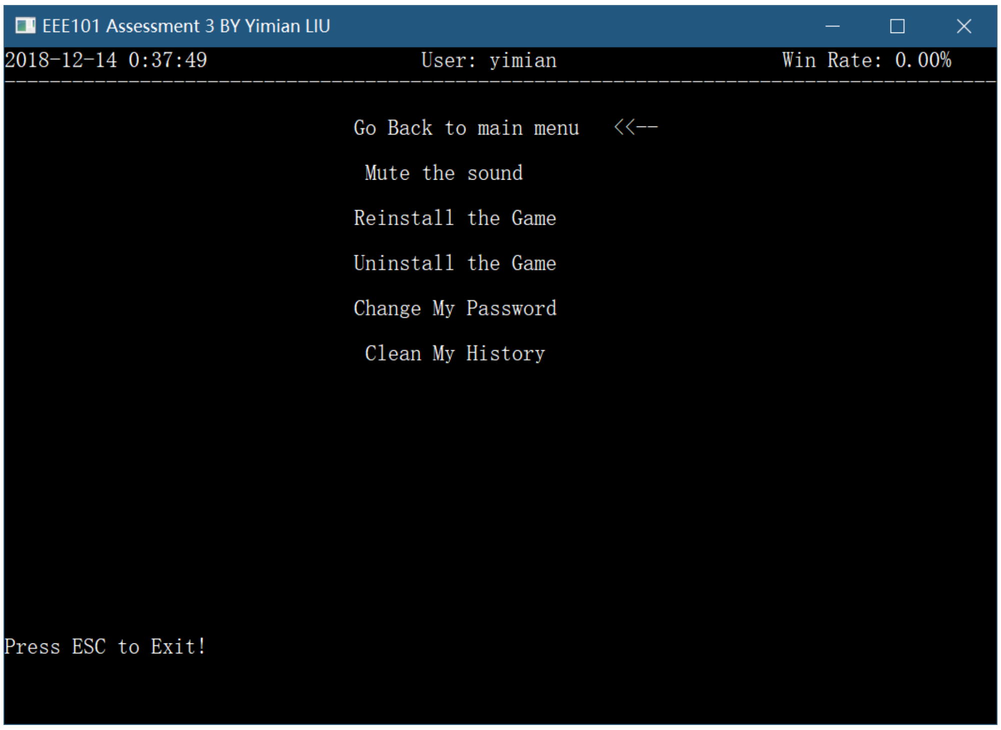
The main menu should contain options such as new game, rank, history, setting (including mute the sound, reinstall the game, uninstall the game, change the password, clean the history), switch users (logout), exit game.

Test result:



```
EEE101 Assessment 3 BY Yimian LIU
2018-12-14 0:35:57                    User: yimian                        Win Rate: 0.00%
-----------------------------------------------------------------------------------------
Please use Arrows on Keyboard to Choose:

                                        New Game    <<--

                                          Rank

                                       My History

                                        Setting

                                      Switch User

                                        Exit Game










Press ESC to Exit!
```

Figure 18 -Test Result of Display the Main Menu



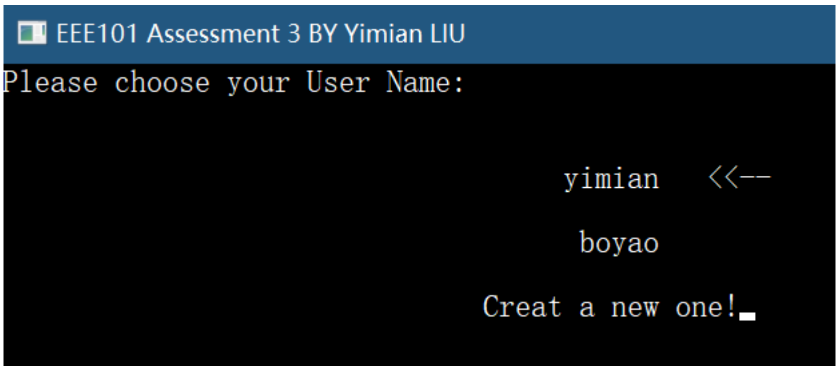
```
EEE101 Assessment 3 BY Yimian LIU
2018-12-14 0:37:49                    User: yimian                        Win Rate: 0.00%
-----------------------------------------------------------------------------------------

                              Go Back to main menu   <<--

                               Mute the sound

                              Reinstall the Game

                              Uninstall the Game

                              Change My Password

                               Clean My History










Press ESC to Exit!
```

Figure 19 – Test Result of Display Setting

Switch Users (Logout)

At the main menu select switch users, program should go to the user select screen.

Test result:



```
Please choose your User Name:

                                   yimian   <<--

                                    boyao

                               Creat a new one!_
```

Figure 20 – Test Result of Switch Users (Logout)

7. Conclusion

In this report, we have introduced how the game was optimized from problem classification, challenges analyses, algorithm design and final test. After this assessment, I obtained a deeper understanding concerning C programing and developing methodology. Besides, I realized that its also meaningful and practical not only to be skillful on C language, but to be familiar with how C can interreact and collaborate with other programing languages such as vbs, php etc. As each language has its pros and cons, if we can master different languages and use them to handle the situation which they are expert in, our developing ability would considerably improve.

There are still shortages of this program. The most significant one is that the program can only be executed in an online condition, which means an offline vision is not available yet. Then, the user game data are not encoded and directly store in the file. This may result in that others can easily access the file and figure the game history of any users. Apart from these, there should be more structure instead of global variables be used. Hopefully that these issues can be solve in the next round of game development.