**Software Development Process**

Assignment\_3

**Problem Statement:**

1) The program will need to provide an interface and provide users with the ability to create accounts, which should include: username, password and game history.

2) The game history contains at least the following details:

* Number of rounds in the game
* Number of player’s wins
* Number of player’s loses
* Number of draws
* The game as a whole is win, lose or draw

3) All account information should be stored in data files and accessed by programs.

4) When users log in to the game, they should be able to:

* Start a new game
* Looking back on their game history
* sign out

5) If illegal input is detected, the program should allow the user to enter illegal input again until the input is correct or the execution of the program terminates. For example:

* Input exceeds legal restrictions.
* Accounts should be matched with passwords
* No specified character or number is entered.
* Not making the right choice.

**Analysis:**

On Input**:**

First, you need to print a message on the screen that lets the user choose whether to log in or register for an account. If execution detects illegal input, it can exit the screen. Then, after registration and login, the screen needs to input a menu to choose from. Finally, the account number, password and game information are entered into the file for preservation.

On Output:

In the selection, a valid output is given to avoid termination of illegal input. The most important point is to allow users to register accounts and passwords normally.

Data structure:

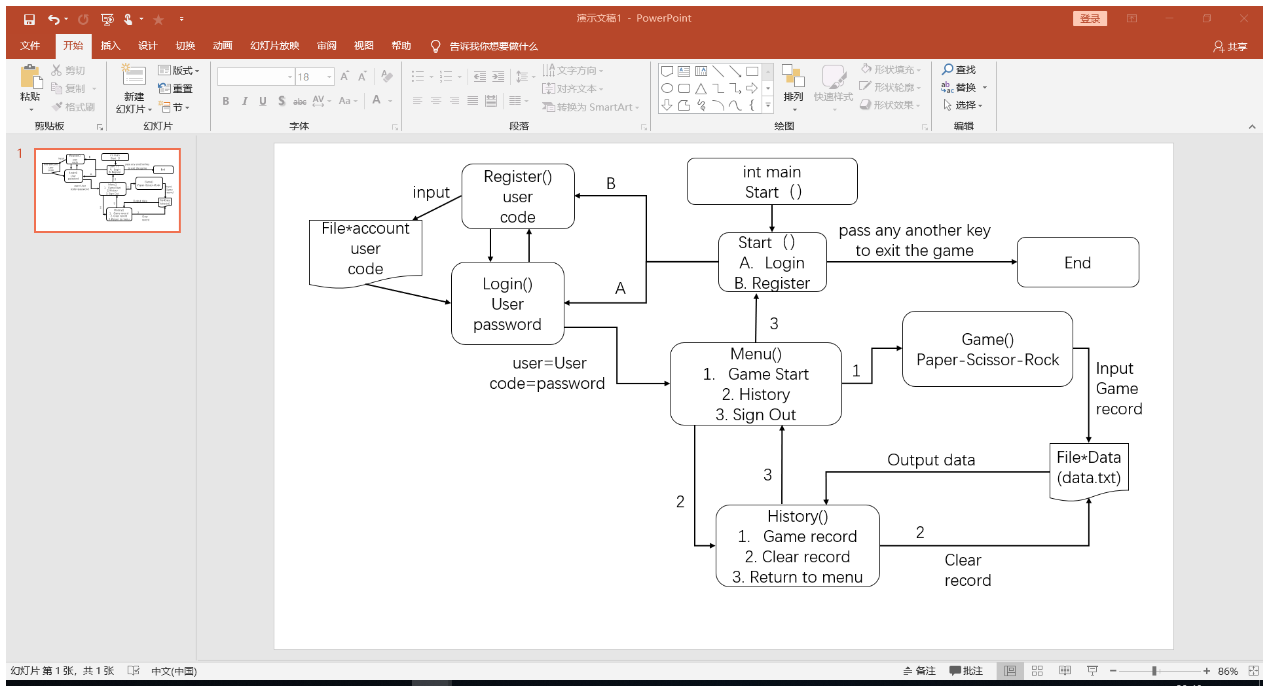
Data is divided into account passwords and historical records.

Algorithm:

No algorithm.

**Design:**

The basic structure of the design was shown in Figure 1 below.



**Figure 1:** Flow chart

First of all, the user information was saved with struct () structure. According to Figure 1, more details were added in each section. In terms of registration, the password will need to be verified twice, and if it is inconsistent, it will be re-registered or returned to the login page. Similarly, if () is used to avoid user duplication. In login, entering username and password needs to be consistent with the registration information before entering the menu (use strcmp () to verify). In addition, a point of two-dimensional array (array [][]) output is used in the part of historical records.

**Implementation：**

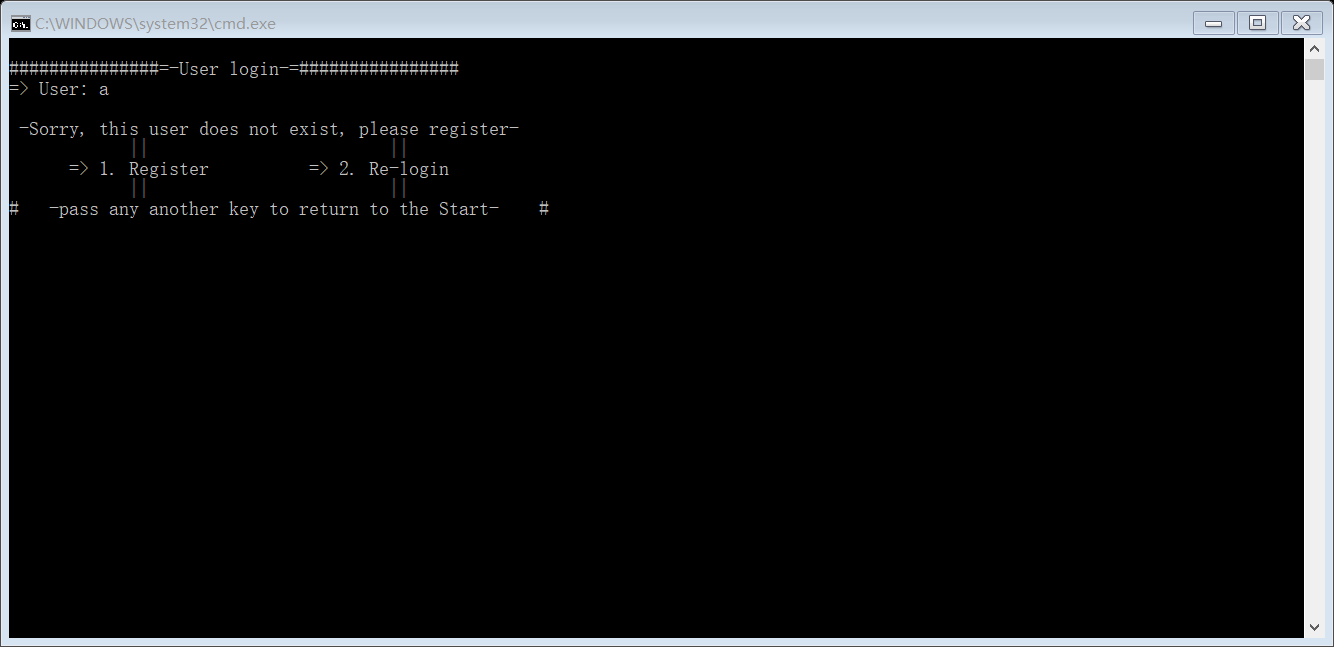
See the C code “1719177\_3.c” (at the end of this document) with comments.

**Testing：**

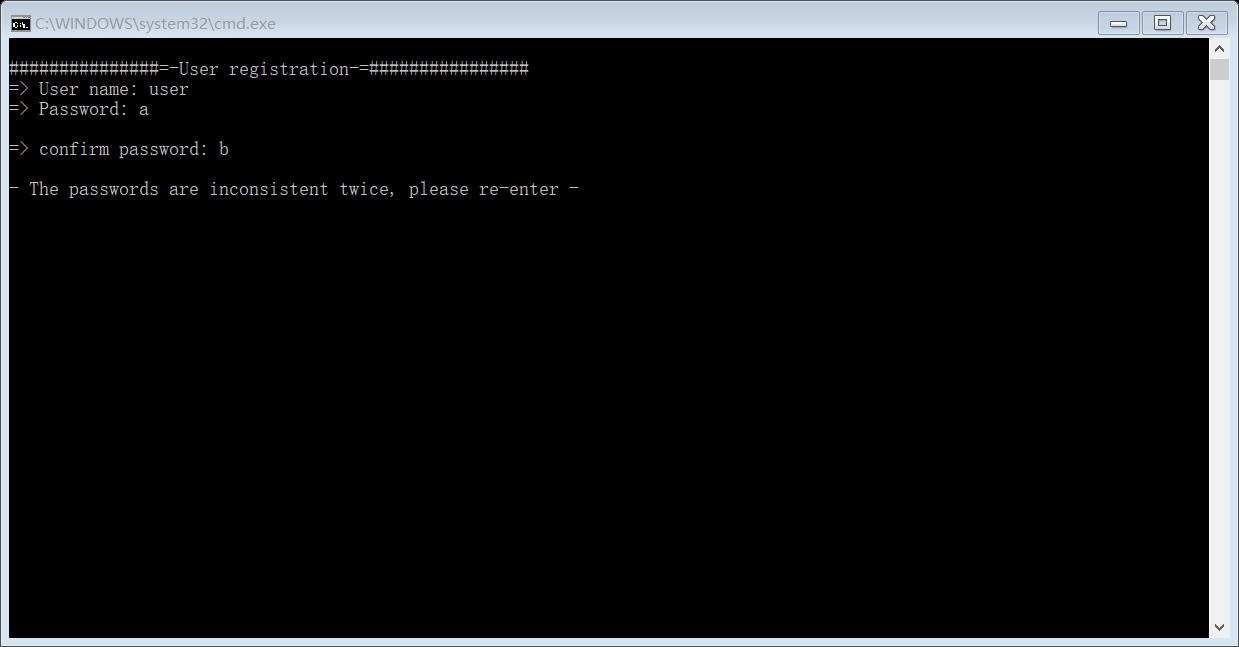
Test 1



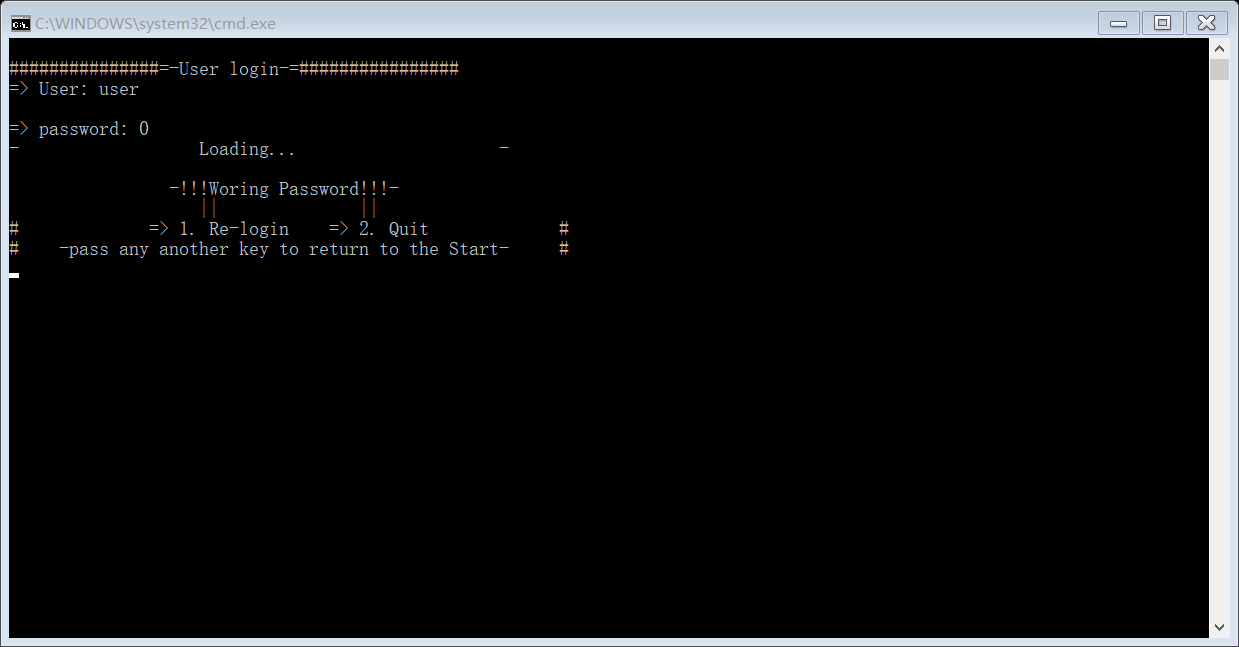
This is the start page. If you press capital A or B, you can jump directly to the corresponding picture.



First entered the login page, but there are no accounts at present. Thus, user will need to register an account number.

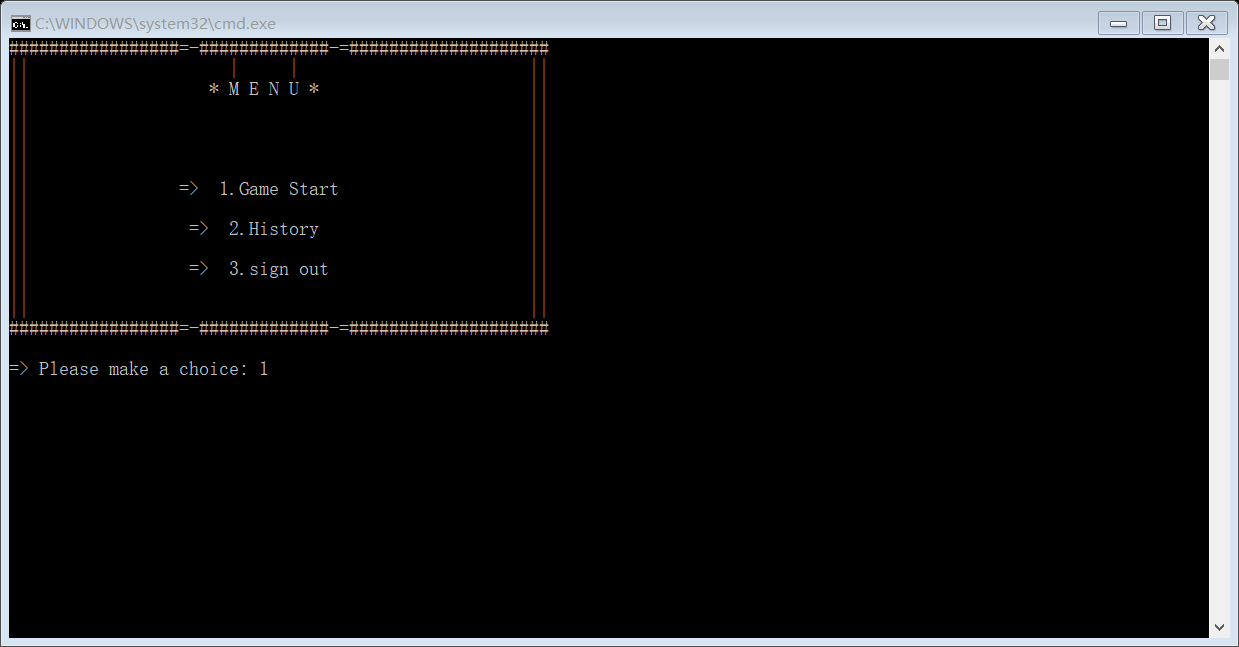


If two passwords are inconsistent, it will return to the registration page.

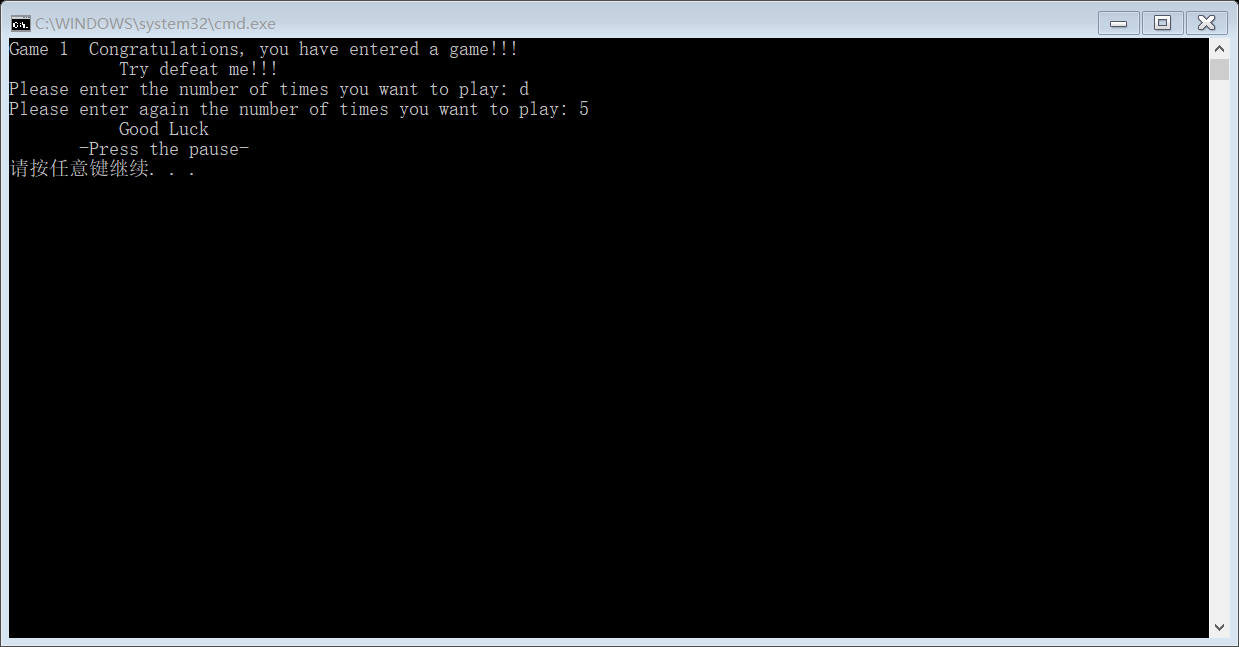


After successful registration, if the password input is inconsistent with the registration, you will be given two options.

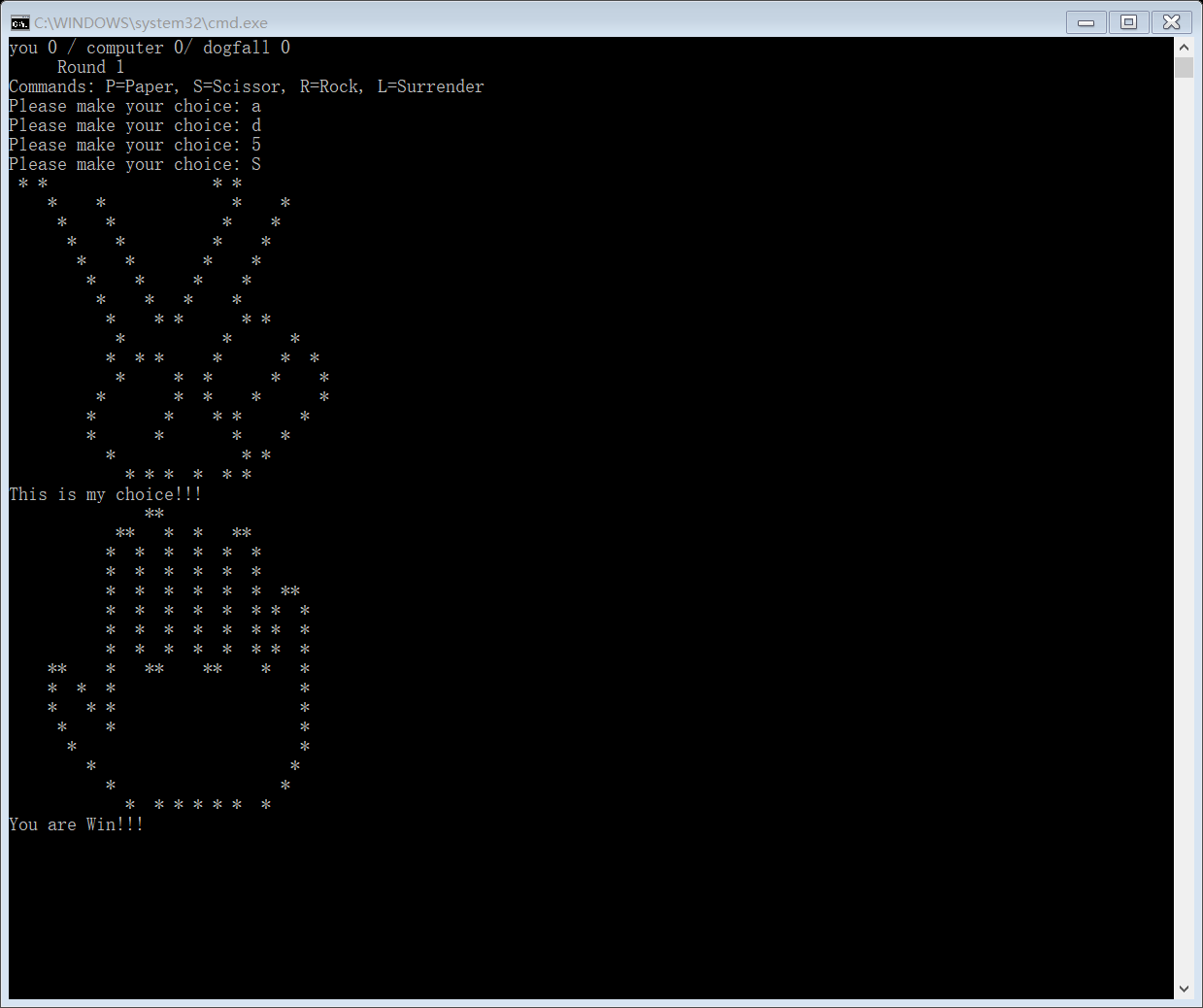
True => (User: user password: 00)



This is a menu page.

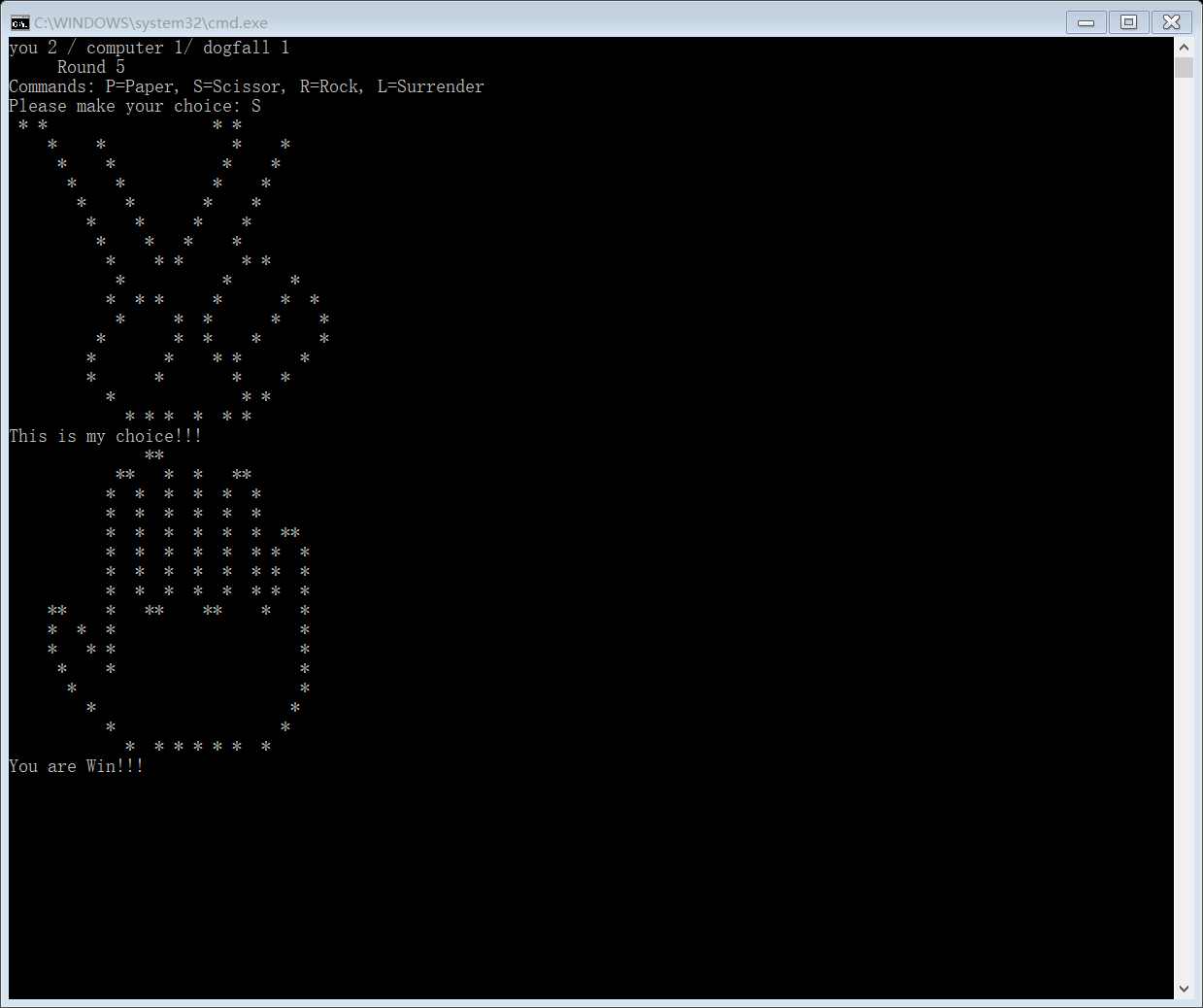


After entering the game, the player needs to type in several times to play. (Input errors can be re-entered)

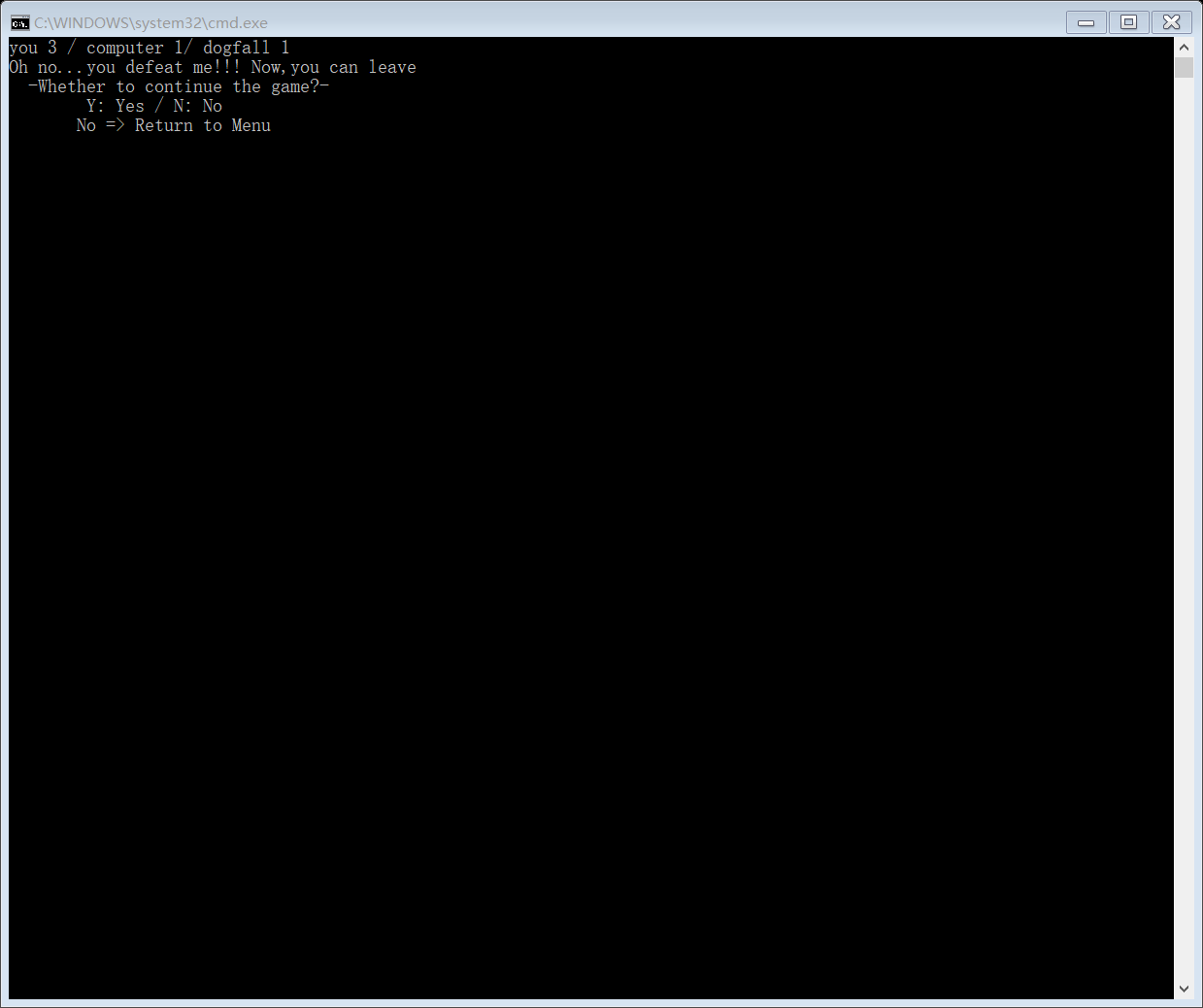


Computers will choose randomly. Then, output the result.

(Input errors can be re-entered)



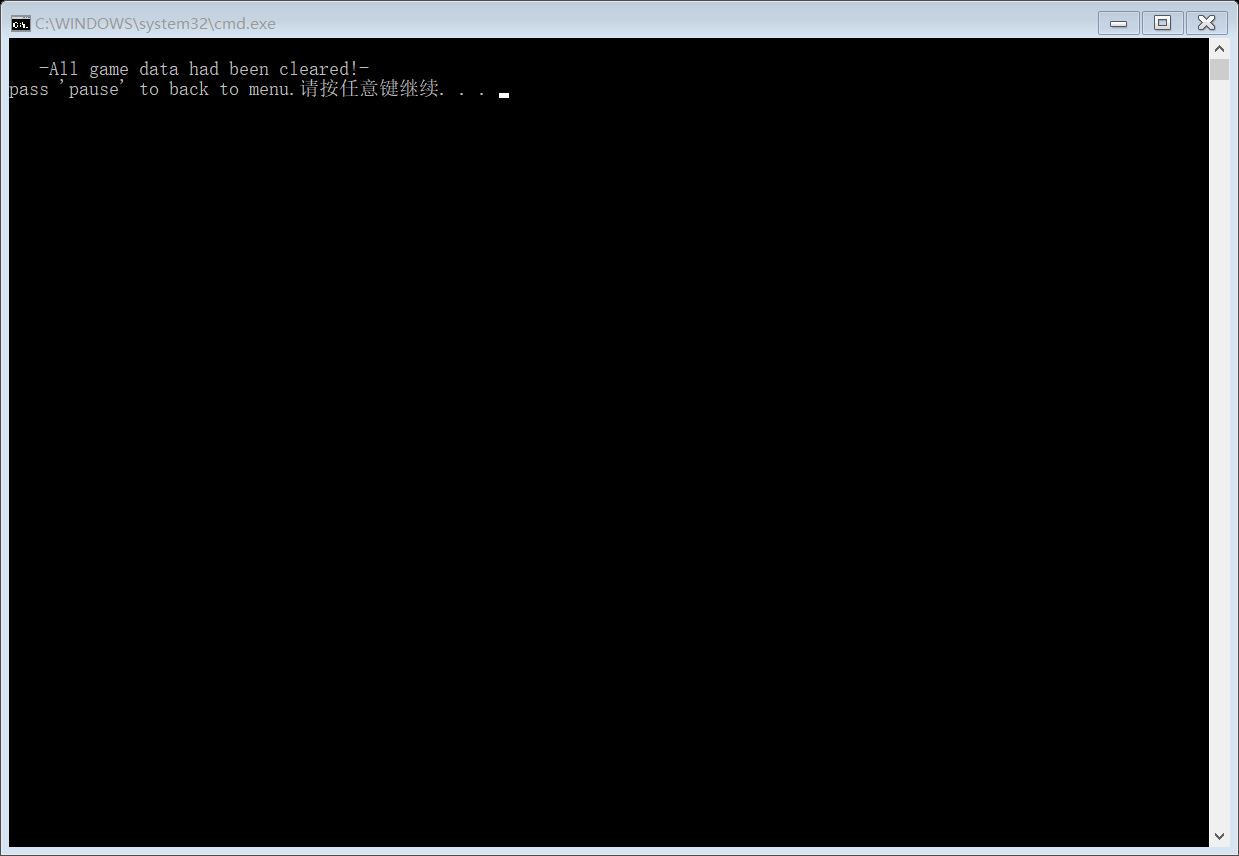
The results are recorded at the top of the page.



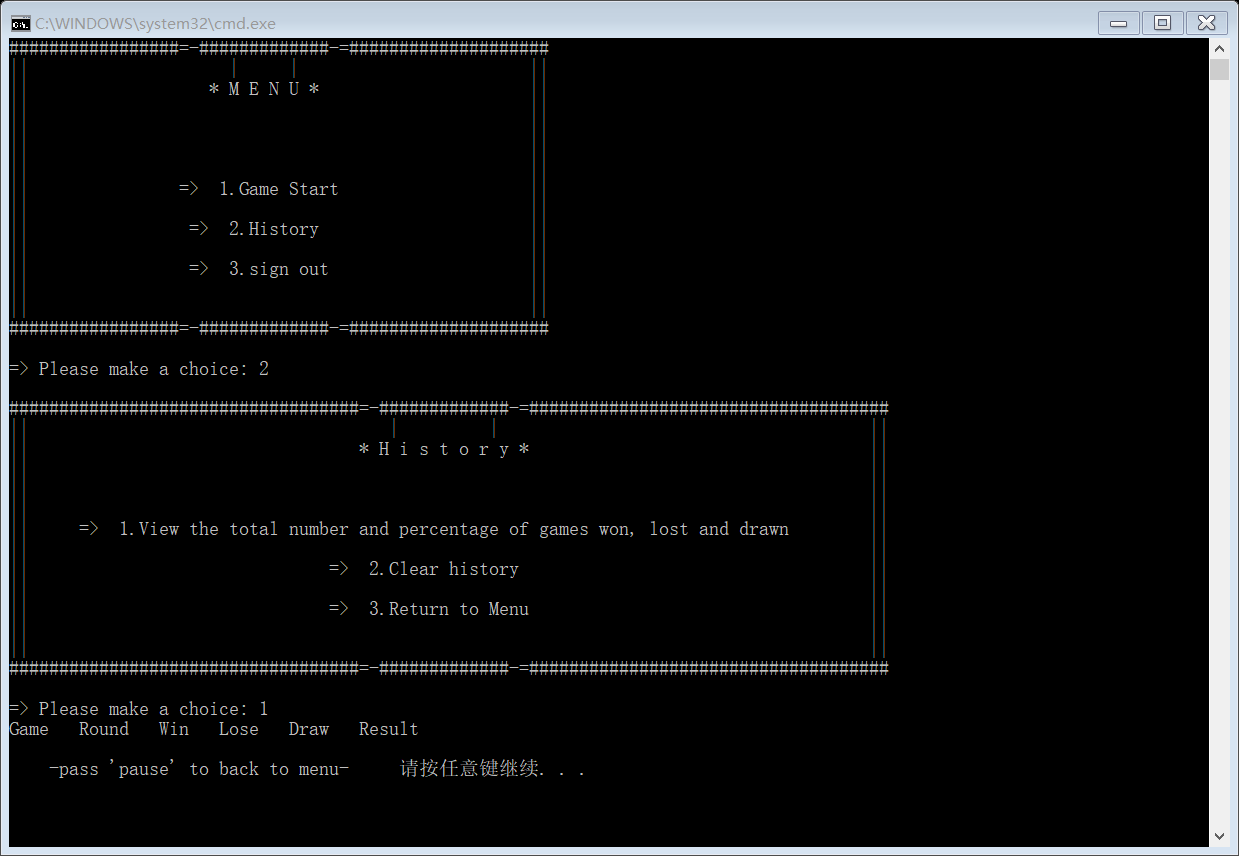
User can choose to play again or not.



Check the history of the game after choosing.



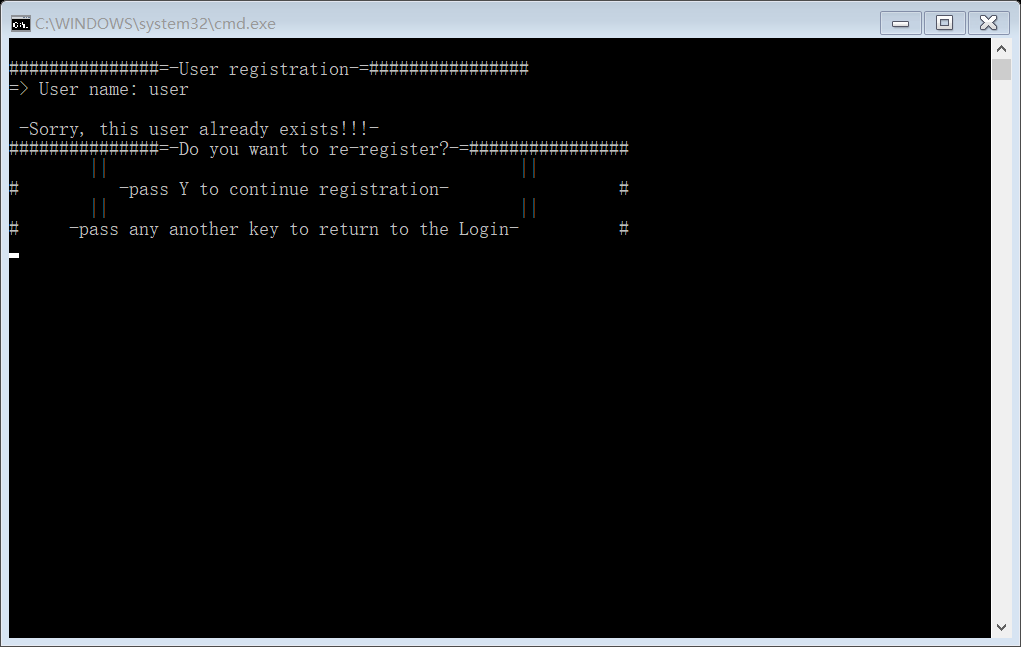
If there is too much data, it can be cleaned up.



After cleaning up, the records are gone.

Test 2

Testing some others.



Registered usernames cannot be re-registered.



Select 3 on the menu page and return to the original page.