EEE101 C Programming Report
Assessment 3

Dec 16, 2018

Name: Ziqi Yang
ID: 1718112

Contents

1 Problem Statement

1.1 Introduction

The program provide an interface through which user can play a user determined number of games of rock, scissors, paper against the computer. In the program, user need to sign up an account first, and log in the account to play games, view the game history, clear the game history or log out. The picture of characters which user and computer select, current scores of user and computer will be display in each game. After chosen number of games have been played, the program will determine who is the final winner and output the result and scores on the screen, and go back to the menu to ask user to start a new game, view the game history, clear the game history or log out. This program has been compiled and run successfully in Visual Studio 2013, with the project which turn off the SDL mode. The limitations of the input data are as follows:

1. In welcome UI (to choose create or log in a account or exit game), user can input "a", "b" or "c".

2. In login UI (to log in the account which has been created), user can input the user name and password of the account they has sign up.

3. In sign up UI (to sign up a new account which has not been created), user can input any character but a full-space string as the user name of the account, and a string without space between 6 and 15 bit as the password of the account.

4. In menu UI (to choose what to do next), user can input "a", "b", "c" or "d" to start a new game, view the history, clear the history or log out.

5. In times UI (to input the times of playing games), user can input a positive number less than 50 as the game times.

6. In rounds UI (to choose the character), user can input "r"(rock), "s"(scissors) and "p"(paper) to represent the character they choose.

7. In final UI, user can input "Enter" to back to the menu UI.

8. In review UI (to view the game history), user can input "Enter" to back to the menu UI.

9. In clear UI (to clear the game history), user can input "y" or "n" to choose whether to clear the game history or not.

1.2 Inputs

If an illegal input is detected, the program will display the illegal input prompt for a few seconds, and then asks user to re-enter again. Being illegal input may be defined as follows:

1. In welcome UI: the input is not "a" or "b".
2. In information input UI:
    1) For the name input: the input is only consisting of space or nothing is input.
    2) For the game times input: the input is not a positive number less than 50.
3. In rounds UI: the input is not "r", "s", or "p".
4. In final UI: the input is not "y" or "n".

1.3 Outputs

The activities after each input are as follow:

1. If input is legal, go to the next step of the program.
2. If input is illegal, prompt user to re-enter.

1.4 Exit

After the game finished, ask user to choose play again or exit.

2 Analysis

2.1 On inputs

Firstly, a message needs to be printed on the screen to let the users know what data/action this program expects. Because the execution is not terminated immediately once an illegal input is detected, but need to ask user to input again. Therefore, the program should accept all kinds of data which user input, so that it can check the legality of all data.

2.2 On outputs

Because the data input by user cannot be predicted before they enter, the outputs are need to be discussed in two types of result, which are correct output with legal input and incorrect output with illegal input. The details of the two types of outputs will be shown as follows:

1. The inputs are legal and outputs are correct:

   1) In welcome UI, it will display the program will start or exit depends on user choice.
   2) In information input UI, it will output nothing and go to next step.
   3) In rounds UI, it will output the score and picture of choice of computer and user.
   4) In final UI, it will output winner and display the program will start or exit depends on user choice.

   All of these outputs are needed to printed on the screen with clear text messages.

2. The inputs are illegal and outputs are incorrect:

   Display the input is illegal and ask user to input again.

2.3 Data structure

1. Enumerate:
   Four enumerate types are used to declare different types of constants and treat them as elements of different sets. The enumerate types defined in the program are as follows:

   - General_Select: represent the character that computer or user choose. elements: rock, scissors, paper.
   - Character_Size: represent the size of character picture to print. elements: mini, normal.
   - Game_Player: represent the winner of current or final. elements: computer, user, none.
   - General_Result: represent the result of input legality check. elements: result_OK(the input is legal), result_Error(the input is illegal).

2. Pointer:
   The definition pointer corresponds to the address of the variable (as a linear structure),

and is passed as a parameter to the function to operate and process the value of the corresponding variable in the function, so that the value of the variable can be changed in the incoming function and read in a layer function.

3. Array:
   Define array to organize and store several variables of the same type in an orderly fashion (as elements of sets). The array in the program is used to store user input and the coordinate of the points in character pictures.

4. Some single data to save local variable data.

2.4 Algorithm

Rock, scissors and paper algorithm. These three characters are mutual restraint. The relationship among them is as Figure 1:

3 Design

3.1 Declare enumerate types

1. General_Select

2. Character_Size

3. Game_Player

4. General_Result

3.2 Go to welcome UI

1. Display welcome messages to user.

2. Ask user to choose start or exit.

3.3 Go to information input UI

1. Ask user input name and do legality check.

2. Ask user input game times and do legality check.

3.4 Go to rounds UI

1. Display current score list and introduce to user that how to input.
2. Ask user to input a character and do legality check.
3. Generate a random number to generate what character the computer chooses.
4. Display the picture of computer and user choose on screen.
5. Compare the characters are chosen, give current winner and add the score.
6. Display current winner this time.
7. Ask user whether to continue game now.
8. The remain game times minus 1.
9. Repeat from step 1 to step 8 until all game times are consumed.

3.5 Go to final UI

1. Compare scores of computer and user and judge the final winner.
2. Display if the user winner.
3. Ask user whether to play again.

3.6 Play again (repeat from 3.3 to 3.5) or exit.

4 Implementation

See the C code "1718112_2.c" with comments.

5 Testing

5.1 In welcome UI

- Input: a
  Return: text: The game will start!

- Input: b
  Return: text: The game will exit...
- Input: sdfgh (any character not "a" or "b")
  Return: text: Your input is illegal, please try again!

5.2 In information input UI

- Input: Ziqi Yang (any character but full-space)
  Return: (go to game times input).
- Input: (space of nothing)
  Return: text: The name cannot be space, please try again!
- Input: Ziqi Yang; 12(a positive integer less than 50)
  Return: (go to rounds UI).
- Input: Ziqi Yang; 0(a number not less than 50, a space, letters)
  Return: text: The times you input is illegal, please try again!

5.3 In rounds UI

- Input: r, s, p
  Return:
  1) the character picture of computer and user chosen with character name below each picture
  2) text: Computer/User/Nobody win(s) this time!
  3) text: Press "Enter" to continue game...
- Input: sdf (anything is not r, s, p)
  Return: text: Your input is illegal, please try again!

5.4 In final UI

- Input: y
  Return: The game will start!

- Input: n
  Return: The game will exit...
- Input: asdf (anything is not y, n)
  Return: Your input is illegal, please try again!