Hong Kong Diploma of Secondary Education Examination

2018

Information and Communication Technology (Coursework)

Option D: Software Development

Title: Puzzle Games

Content Page

Chapter 1 - Introduction

1.1 Background

A Puzzle & Games Society would like to design a system that provides various puzzle games for children to play. As the designer of the system, I am needed to design the system and write the relevant report.

1.2 Objectives

After analysis and considering, it is decided that there will be several requirements when designing the system.

a) Three different games are included

b) Rules and Introduction of each game are included in the program

c) Different game modes of each game are provided

d) Login function is provided

e) Account registration function is provided

f) Scores of player in different games are recorded

Chapter 2 - Design

2.1 Brief Description

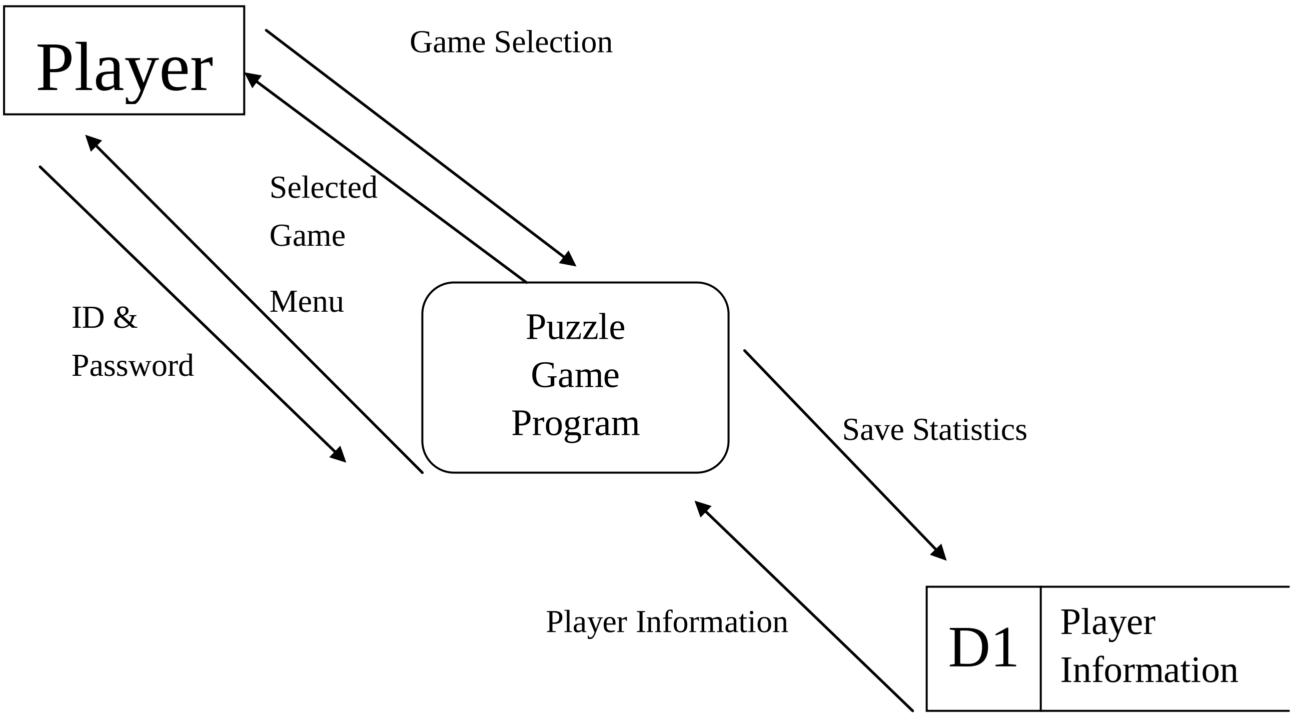
In this chapter, the designation of the system will be shown based on the functions and objectives stated in Chapter 1.

The following will be designed:

1. The Level 0 data flow of the system
2. The Level 1 data flow of the system
3. The Level 2 data flow of the system
4. The Structure Chart of the system
5. The System Flowchart of whole system
6. The System Flowchart of 3 Games

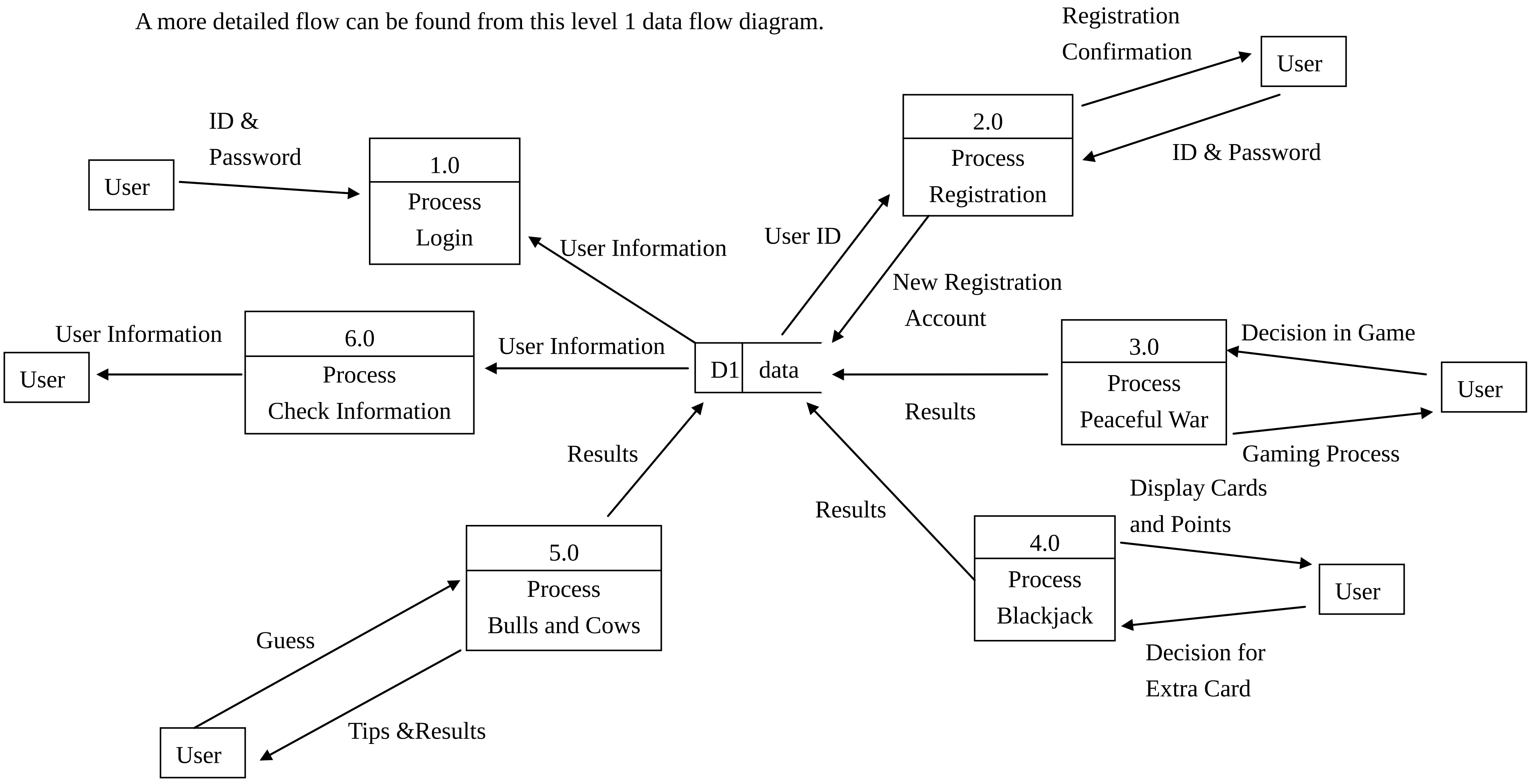
2.2 The Level 0 data flow of the system

The system consists of the puzzle games program and the personnel. The basic flow of the system can be seen from the following data flow diagram (level 0):



2.3 The Level 1 data flow diagram of the system

A more detailed flow can be found from this level 1 data flow diagram.



The details of sub-programs in the level 1 data flow diagram of the system are as follows.

Sub-program	Description
1.0 Process Login	This process allows user to input his/her ID and Password into the system. Then, the process will retrieve users' information from D1 data so as to do verification.
2.0 Process Registration	This process allows user to input the choice, ID and Password for registration. The process will then send new user's information to D1 data and send registration results to user.
3.0 Process Peaceful War	This process require user to input his/her decisions in the game. The process will then return the statistics in game to user. At the end of the game, the process will send the result of the game to D1 data.
4.0 Process Blackjack	This process require user to input his/her decisions in the game that whether extra card is needed or not. The process will then return the cards and relative points of the cards in game to user. At the end of the game, the process will send the result of the game to D1 data.
5.0 Process Bulls and Cows	This process require user to input his/her guess of the code in the game. The process will then return the tips and the guessing results in game to user. At the end of the game, the process will send the result of the game to D1 data.
6.0 Process Check Information	This process will retrieve user's information for D1 data. After that, the process will send the information to its own users.

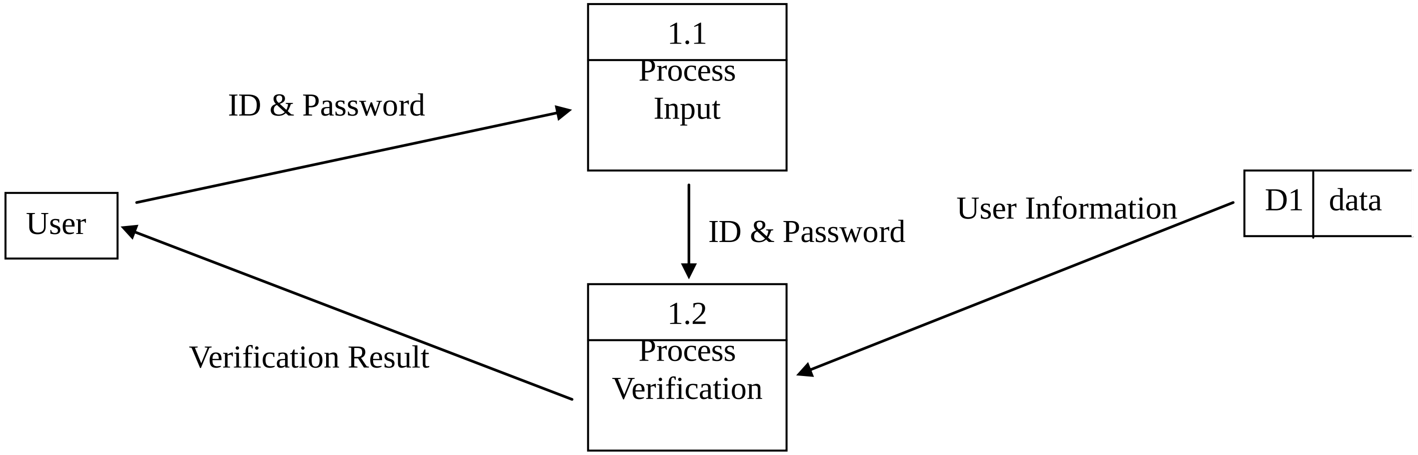
The details of the data file in the level 1 data flow diagram of the system are as follows.

Database File	Description
D1 data	This database file stores all the users' information. For example, the users' information like users' ID, Password, number of games won, lost and draw in Peaceful War and Blackjack and the winning and losing times in Bulls and Cows.

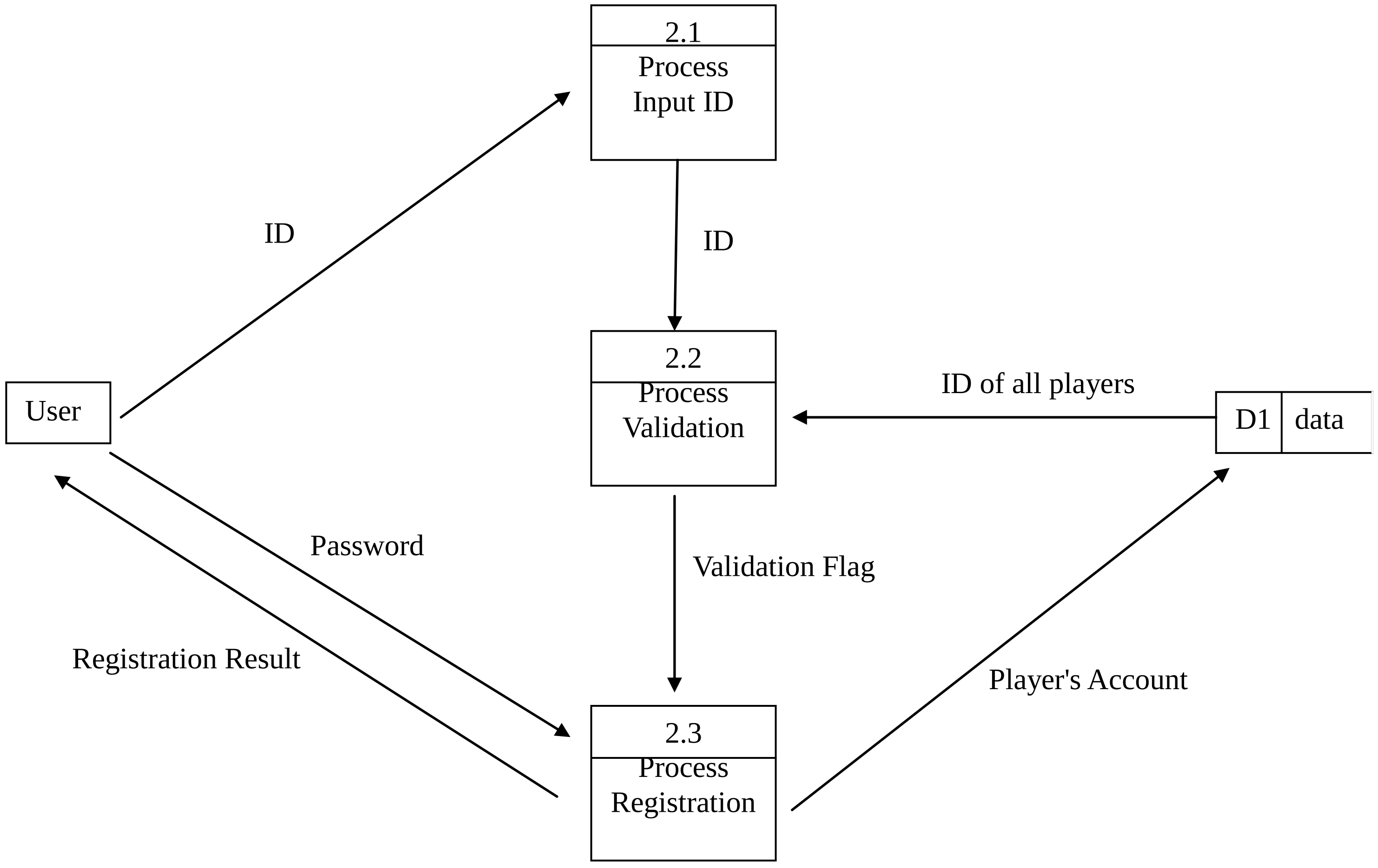
2.4 The level 2 data flow diagram of different process

A more detailed flow can be found from the following level 2 data flow diagrams.

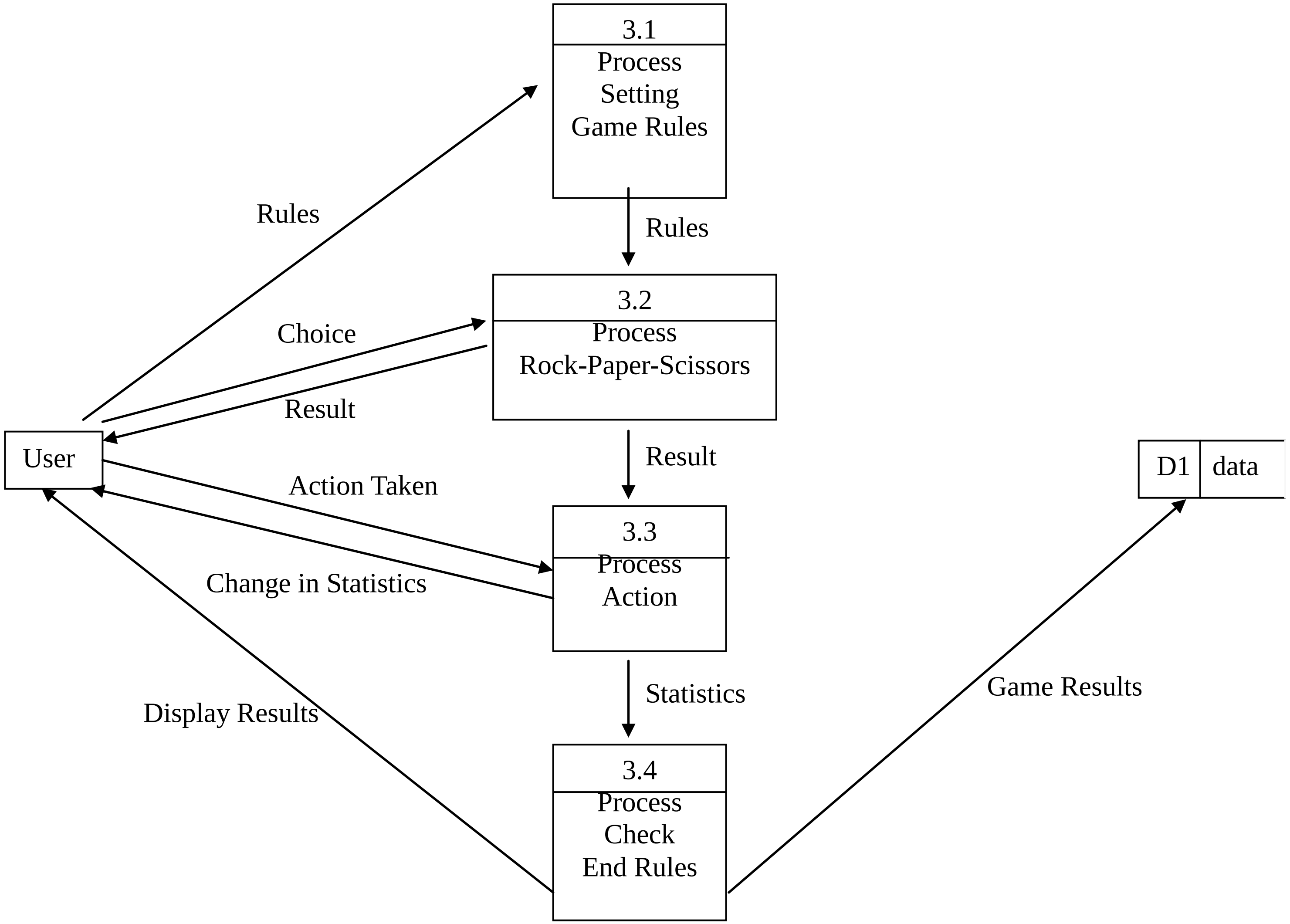
Process 1.0:



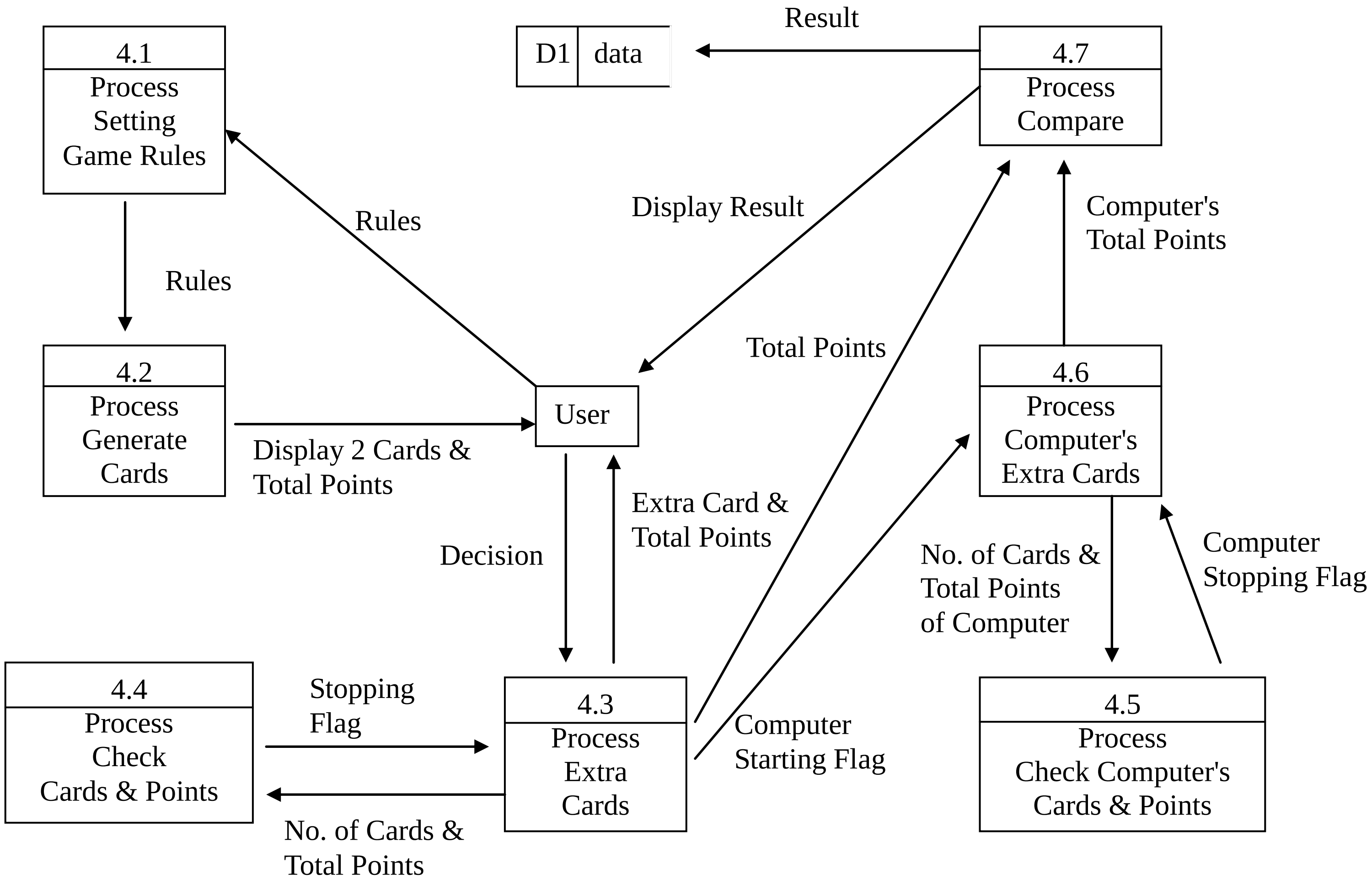
Process 2.0:


2.1
Process
Input ID
ID
ID
User
2.2
Process
Validation
ID of all players
D1
data
Password
Validation Flag
Registration Result
Player's Account
2.3
Process
Registration

Process 3.0:

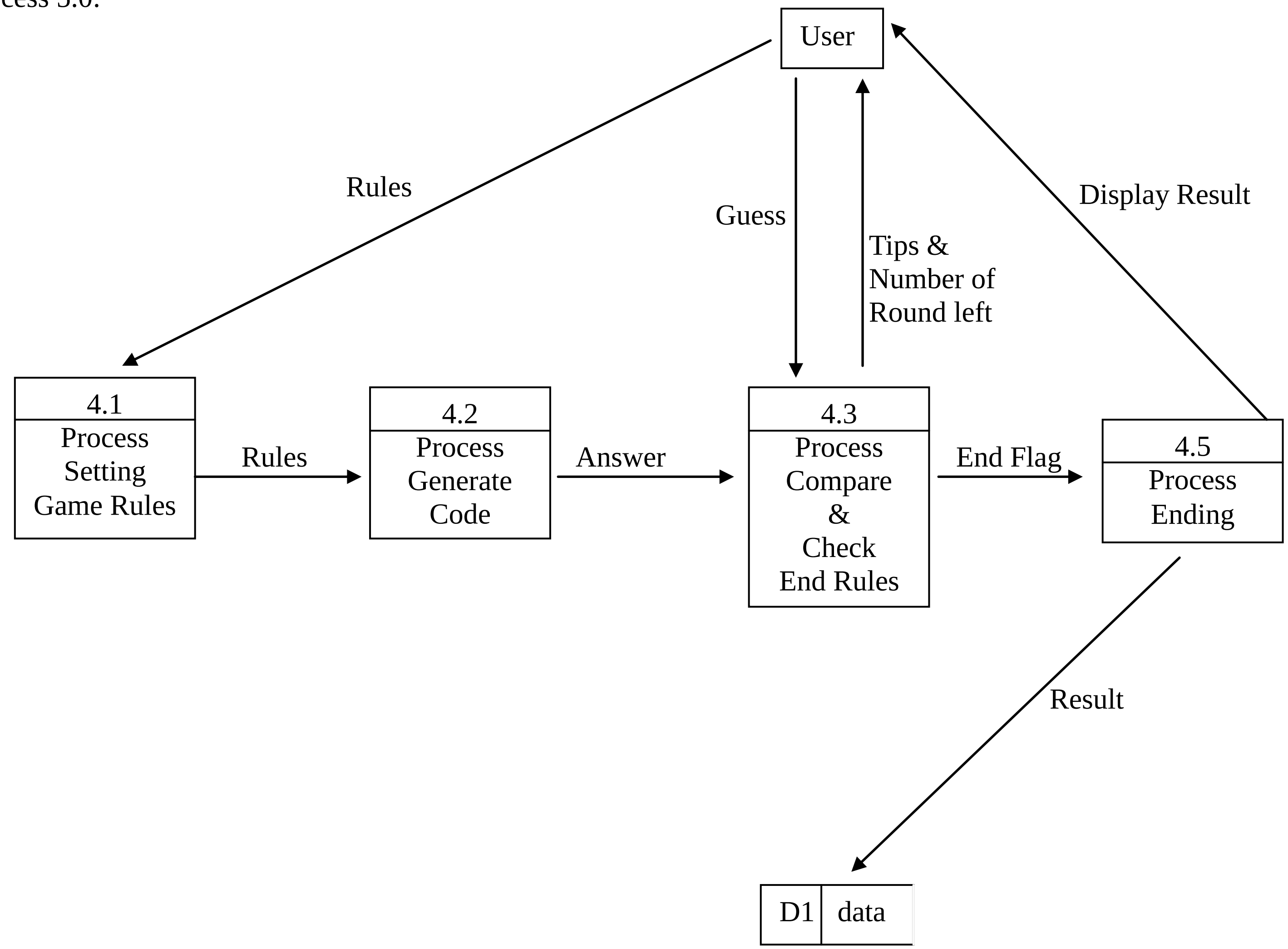

3.1
Process
Setting
Game Rules
Rules
Rules
3.2
Process
Rock-Paper-Scissors
Choice
Result
User
Result
Action Taken
D1
data
3.3
Process
Action
Change in Statistics
Statistics
Display Results
Game Results
3.4
Process
Check
End Rules

Process 4.0:

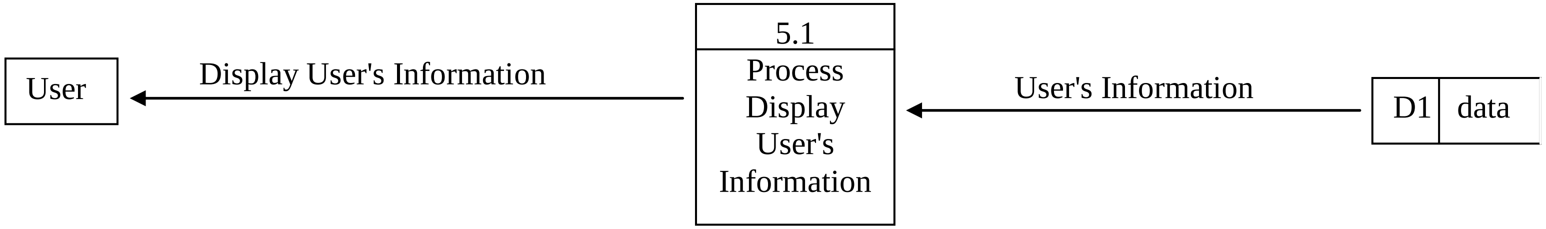


4.1
Process Setting Game Rules
D1
data
Result
4.7
Process Compare
Rules
Display Result
Computer's Total Points
Rules
Total Points
4.2
Process Generate Cards
User
4.6
Process Computer's Extra Cards
Display 2 Cards & Total Points
Extra Card & Total Points
Decision
No. of Cards & Total Points of Computer
Computer Stopping Flag
4.4
Process Check Cards & Points
Stopping Flag
4.3
Process Extra Cards
Computer Starting Flag
4.5
Process Check Computer's Cards & Points
No. of Cards & Total Points

Process 5.0:

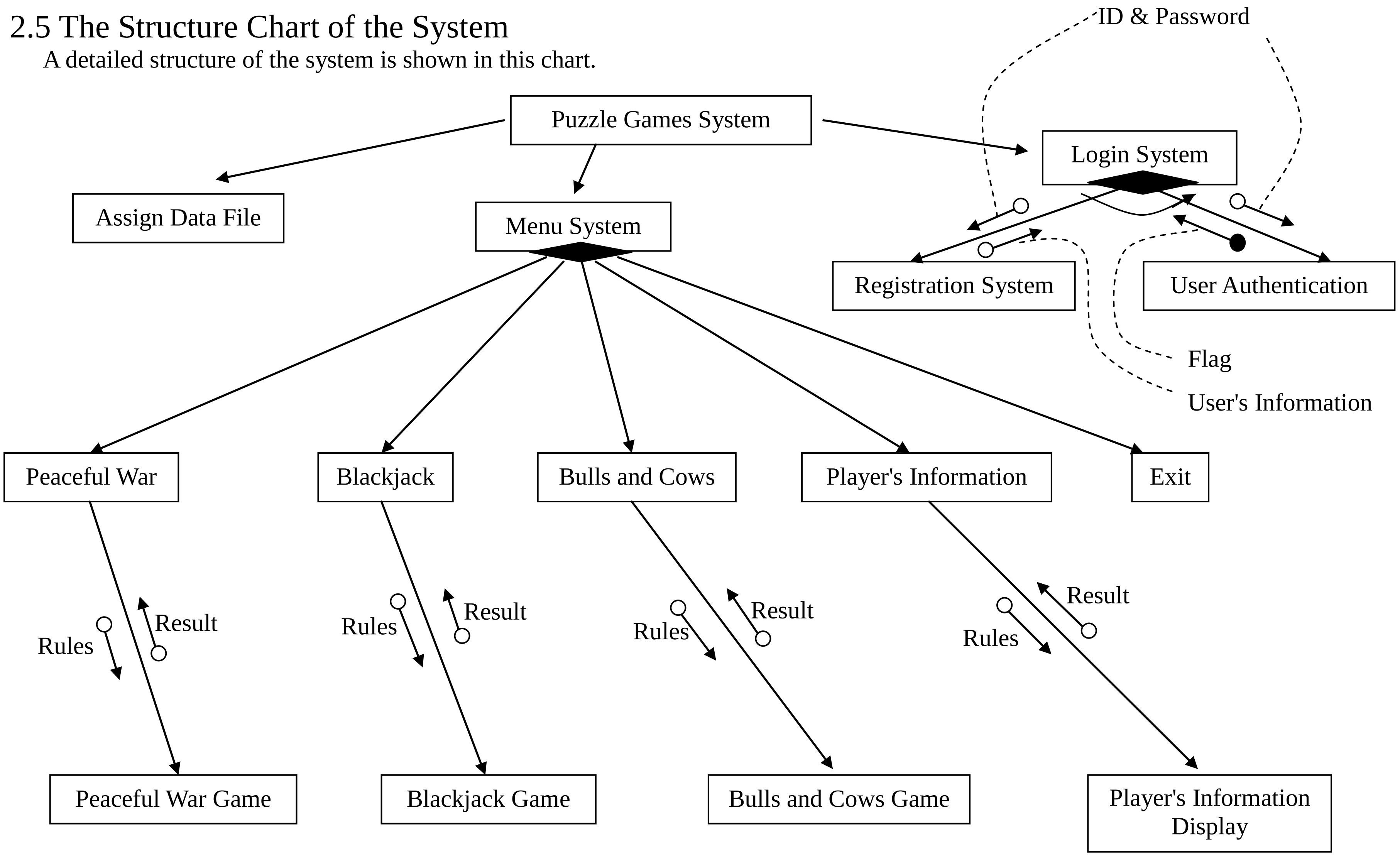

User
Rules
Guess
Tips &
Number of
Round left
Display Result
4.1
Process
Setting
Game Rules
Rules
4.2
Process
Generate
Code
Answer
4.3
Process
Compare
&
Check
End Rules
End Flag
4.5
Process
Ending
Result
D1
data

Process 6.0:


User
Display User's Information
5.1
Process Display User's Information
User's Information
D1
data

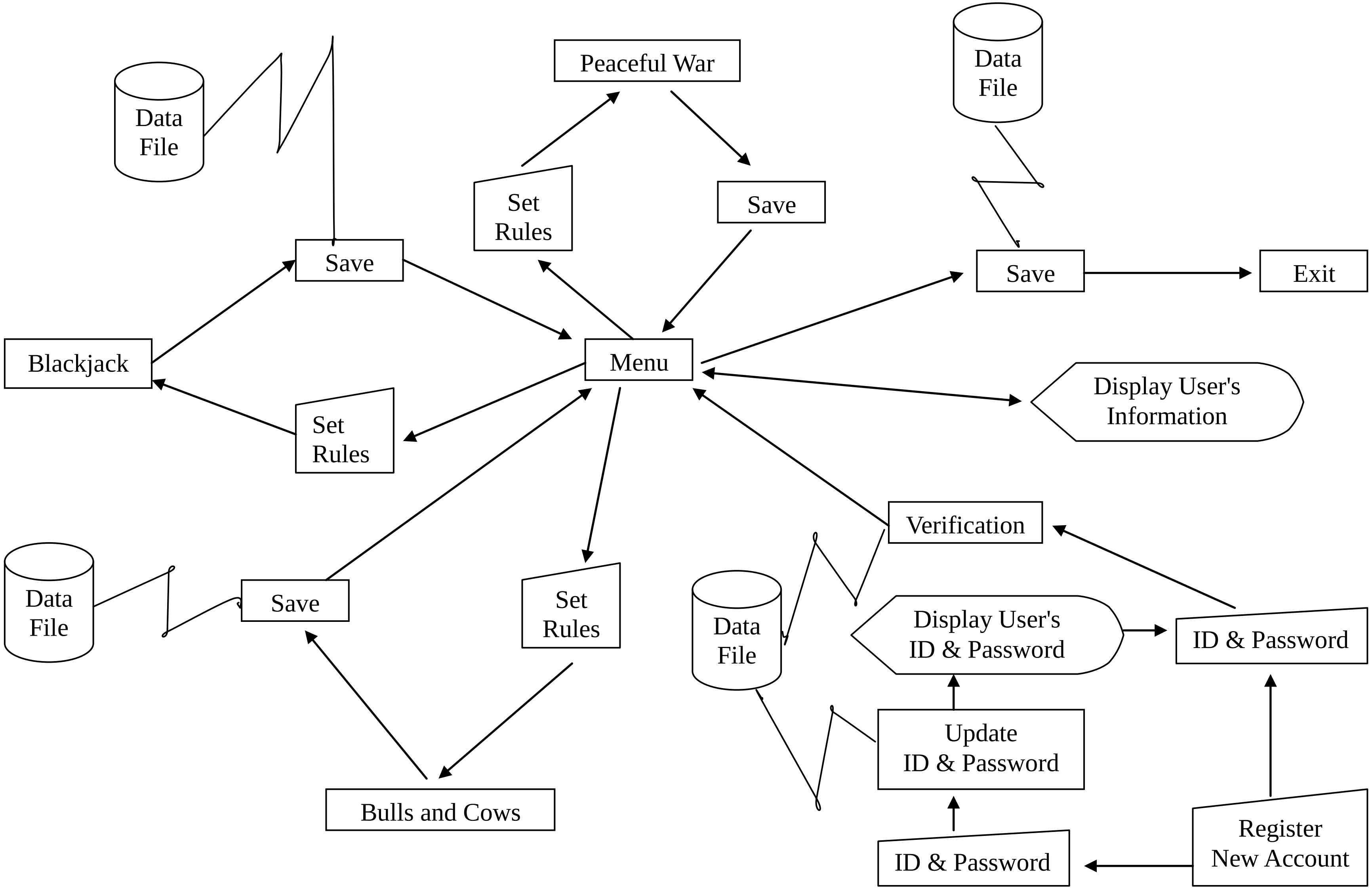
2.5 The Structure Chart of the System

A detailed structure of the system is shown in this chart.



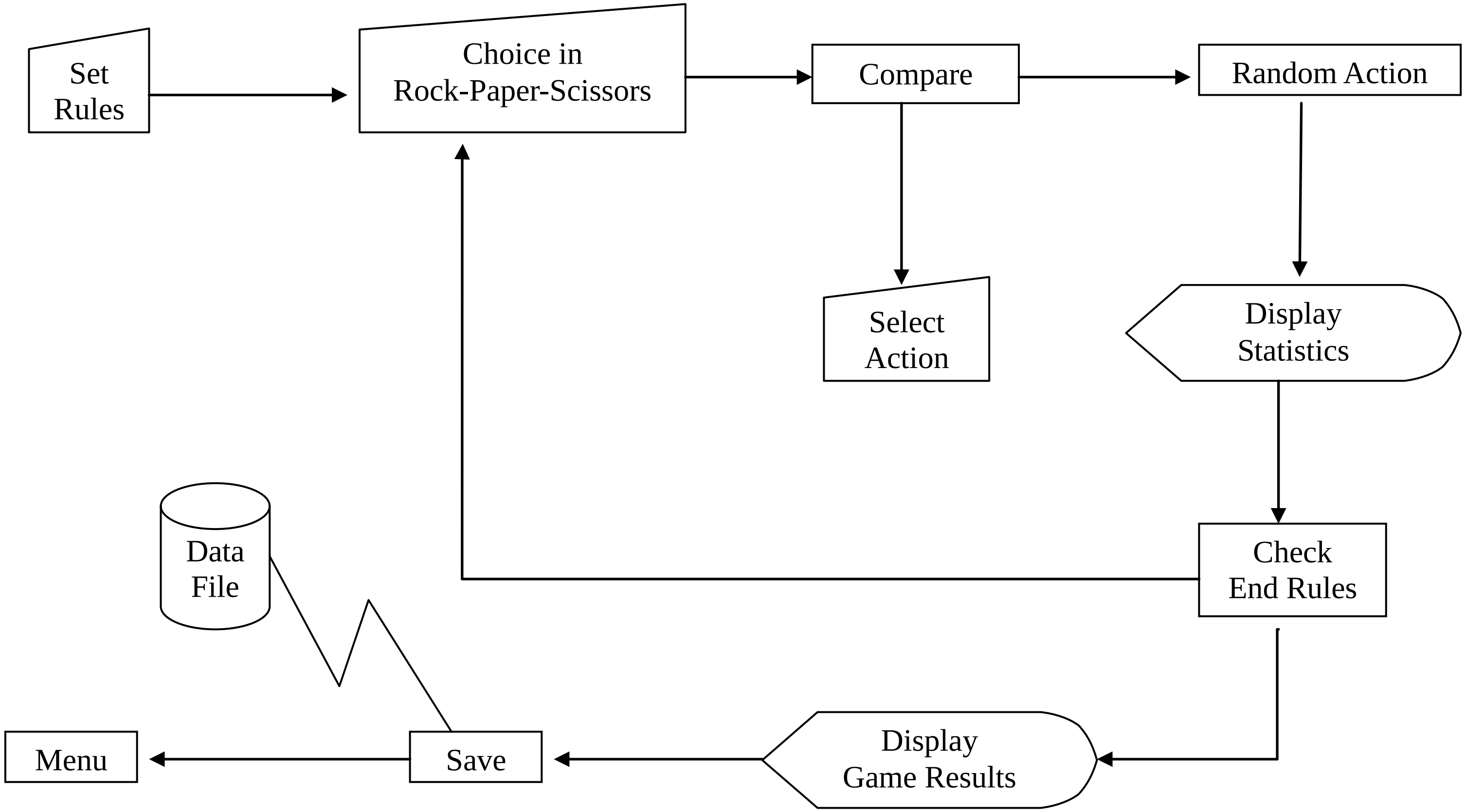
2.6 System Flowchart of Whole System

The flow of the whole system will be shown in the chart.



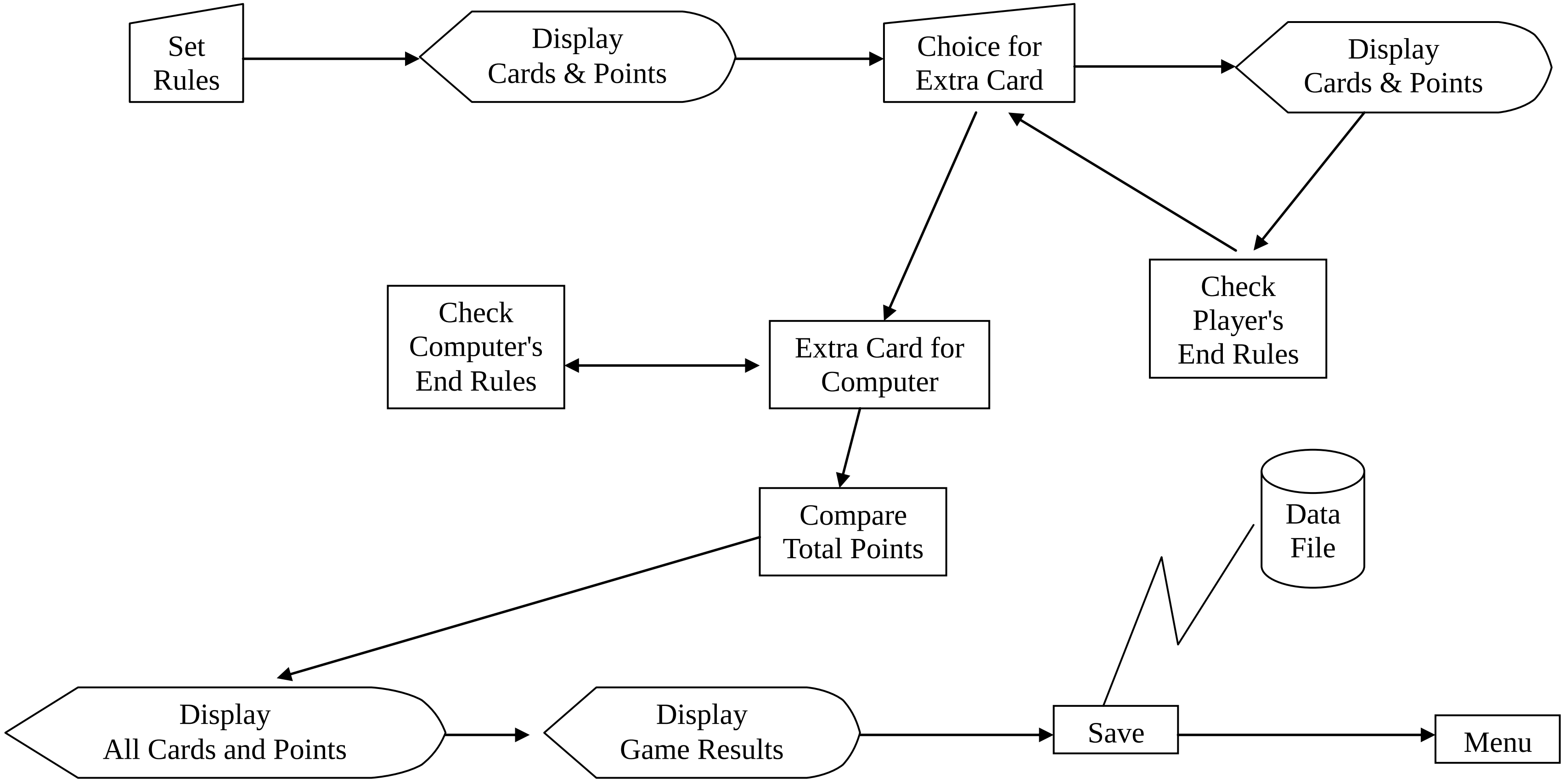
2.7.1 System Flowchart of Peaceful War

This is the detailed system flowchart of Peaceful War Game



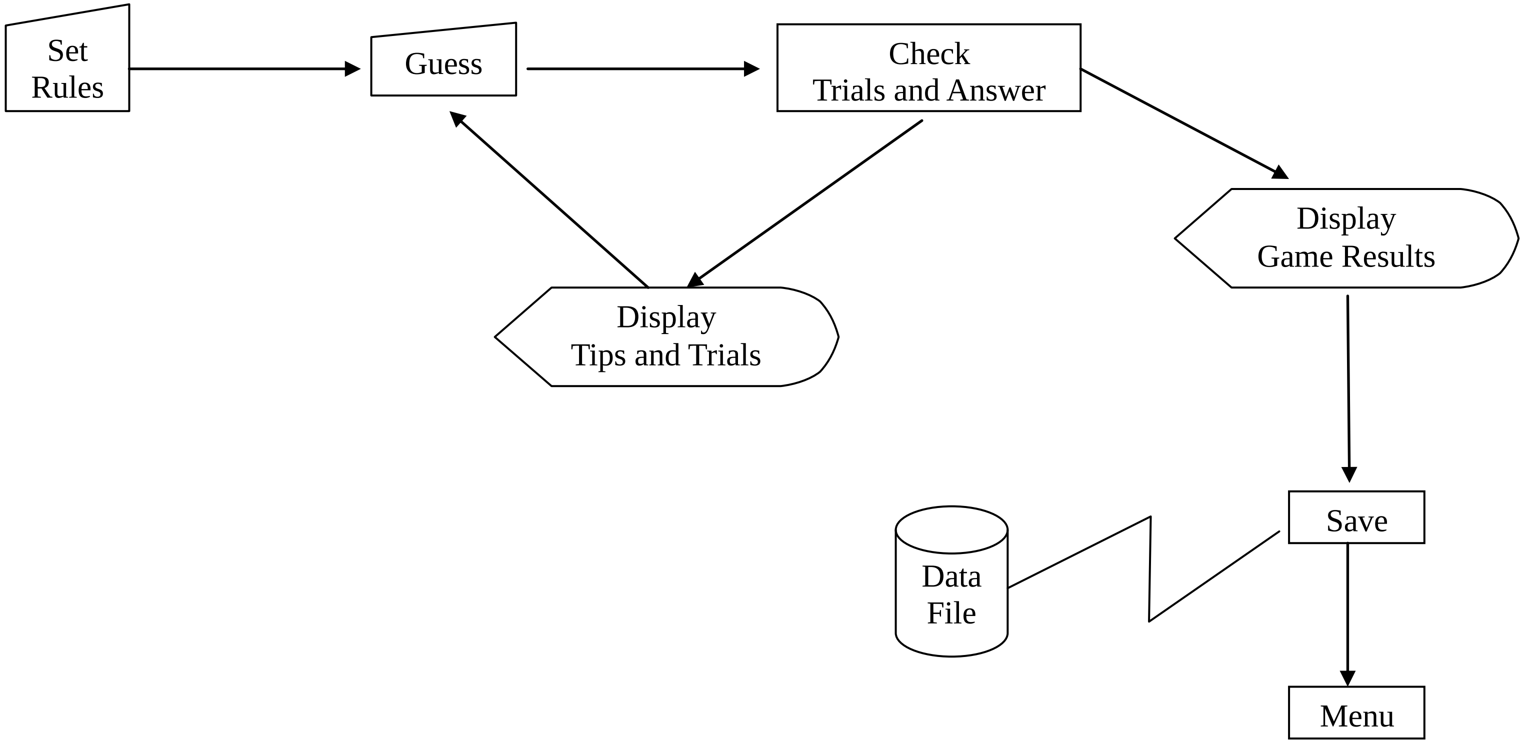
2.7.2 System Flowchart of Blackjack

This is the detailed system flowchart of Blackjack Game



2.7.3 System Flowchart of Bulls and Cows

This is the detailed system flowchart of Bulls and Cows Game



Chapter 3 - Implementation

3.1 Brief Description

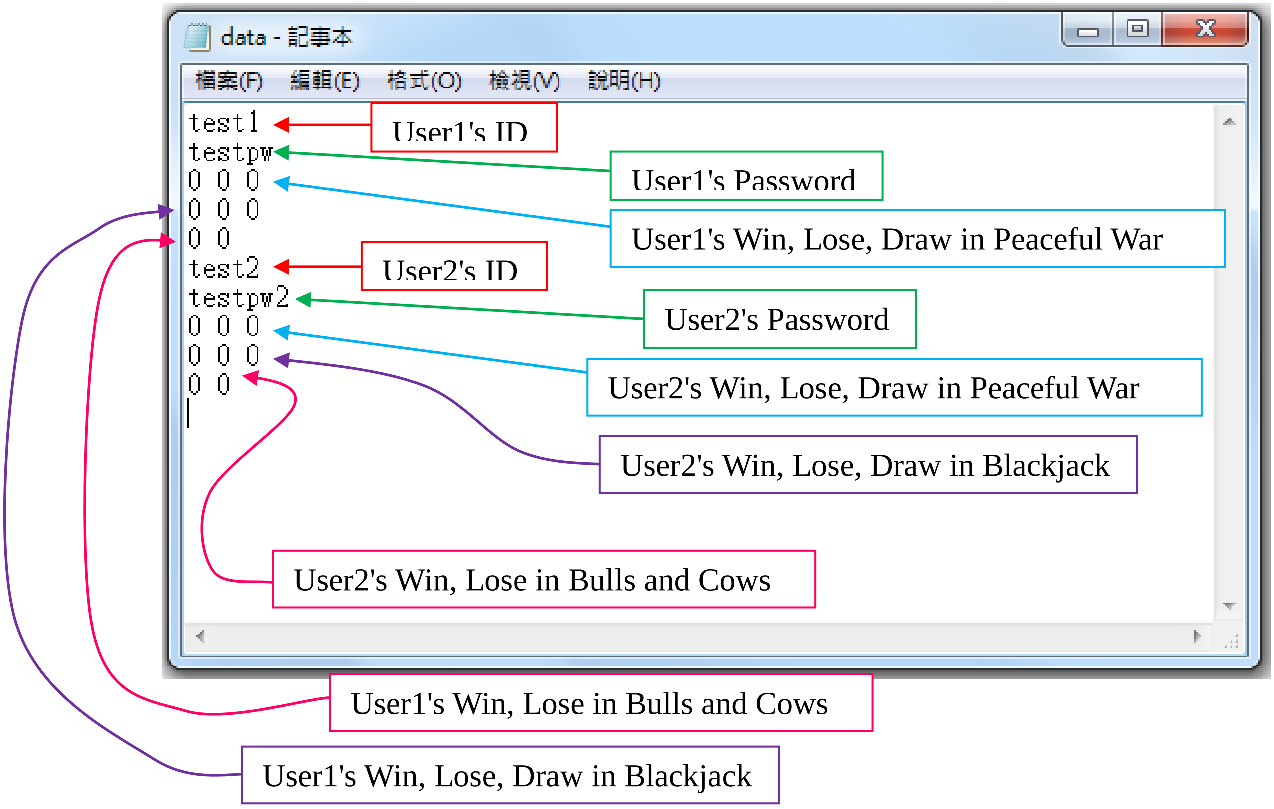
In this chapter, implementation of different aspects in the system will be shown.

The following will be shown:

1. Database Implementation
2. User Interface Implementation
3. Process Implementation

3.2 Database Implementation

The Puzzle Games System consists of 1 data file only. It exists as '.txt' file and it is named 'data.txt'. The details of the structure of the data file are as follows.



```
test1
testpw
0 0 0
0 0 0
0 0
test2
testpw2
0 0 0
0 0 0
0 0
```

data - 記事本
檔案(F)
編輯(E)
格式(O)
檢視(V)
說明(H)
User1's ID
User1's Password
User1's Win, Lose, Draw in Peaceful War
User2's ID
User2's Password
User2's Win, Lose, Draw in Peaceful War
User2's Win, Lose, Draw in Blackjack
User2's Win, Lose in Bulls and Cows
User1's Win, Lose in Bulls and Cows
User1's Win, Lose, Draw in Blackjack

The mentioned text file will communicate with the puzzle games system program through file related functions in Pascal. Under normal circumstances, the data in text file will first be read into the arrays inside the program, the program will then manipulate the data inside the arrays. The arrays serve as an intermediary between the text file and the program when reading or writing the text file is needed. The following is the explanation of the structure of several array lists used as links between text file and the program.

Array used in storing information of text file: user = array[1..100] of userinfo
The 'userinfo' is a record of the following variables:

Variable Name	Variable Type	Description
ID	String	Store user's ID
PW	String	Store user's password
winPWar	Integer	Store number of Peaceful War won by user
losePWar	Integer	Store number of Peaceful War user lost
drawPWar	Integer	Store number of Peaceful War that came to a draw
winBJ	Integer	Store number of Blackjack won by user
loseBJ	Integer	Store number of Blackjack user lost
drawBJ	Integer	Store number of Blackjack that came to a draw
winMM	Integer	Store number of Bulls and Cows won by user
loseMM	Integer	Store number of Bulls and Cows user lost

In all cases, the text file will be loaded to the array by the procedure 'ReadFile'. The statement 'while not eof(data)' is used to read the text file without losing any information.

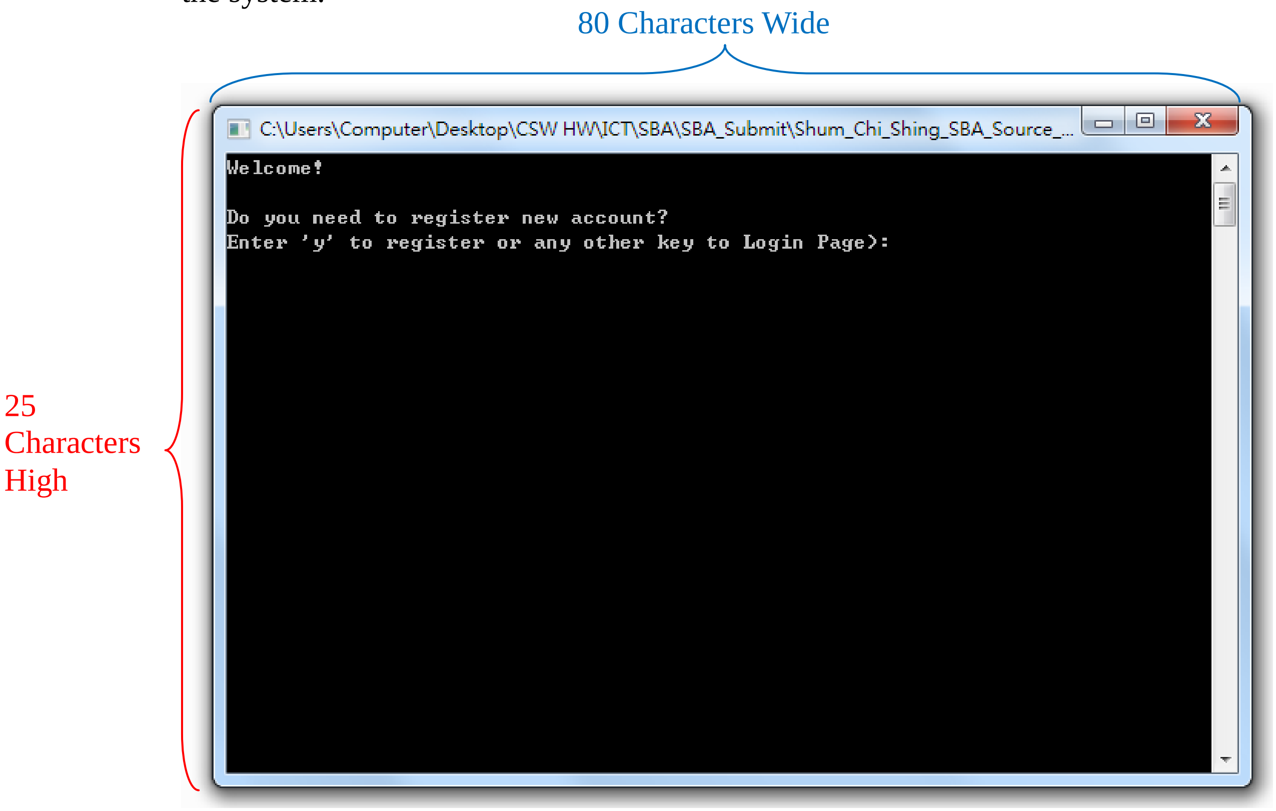
Meanwhile, the information in the text file can be modified using the information stored in the array list through the procedure 'SaveFile'. The pointer 'code' is used in counting the number of records to be written so it is guaranteed that there will be no missing after every running of procedure 'SaveFile'.

3.3 User Interface Implementation

The puzzle game system program is 80 characters wide and 25 character high. It provides a menu so as to give instructions and guidelines to its users. Users can simply follow the instructions and guidelines given by the system and input the corresponding command in order to enjoy the functions provided by the system.

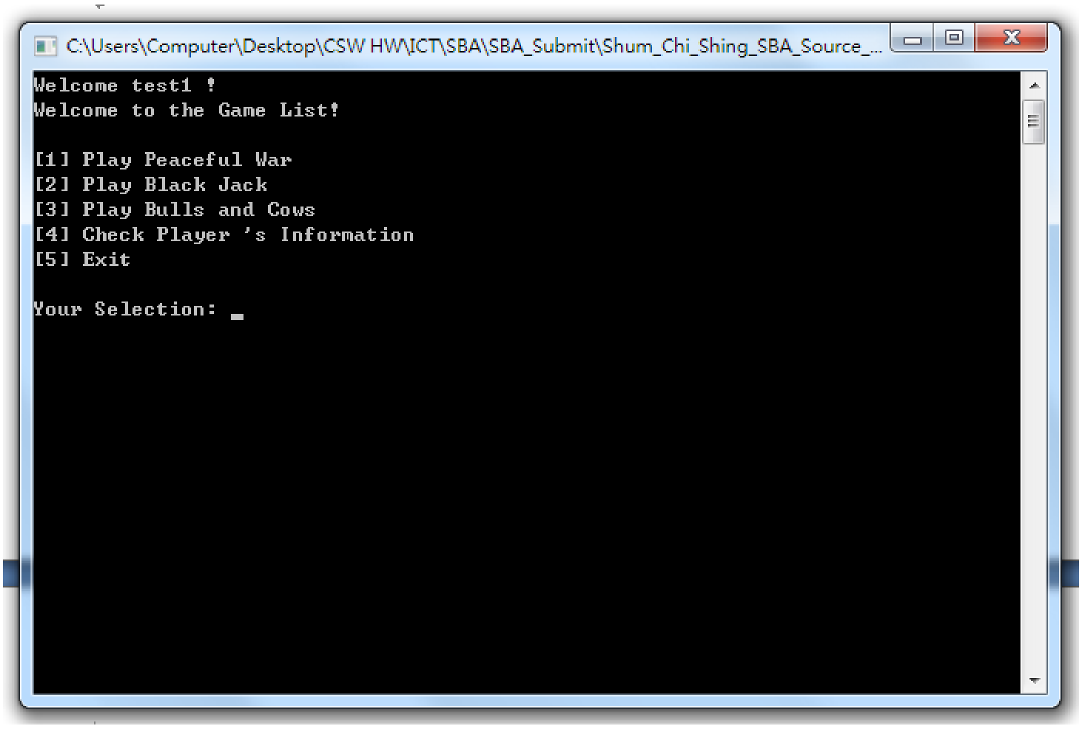
If errors related to invalid input occur, the relative sentence stating the guidelines and instructions will be shown and users will be kept requiring for the valid input unless they finally input the valid one.

Another feature is that when the system is needed to stop at particular moments, a sentence stating 'Press "Enter" to continue' will exist to offer time to users to make a look at the moment that the system hopes them to check carefully. Once they have finish reading the information, they can press 'Enter' key to continue the running of the system.


80 Characters Wide
25 Characters High
C:\Users\Computer\Desktop\CSW HW\ICT\SBA\SBA_Submit\Shum_Chi_Shing_SBA_Source_...

```
Welcome!

Do you need to register new account?
Enter 'y' to register or any other key to Login Page):
```

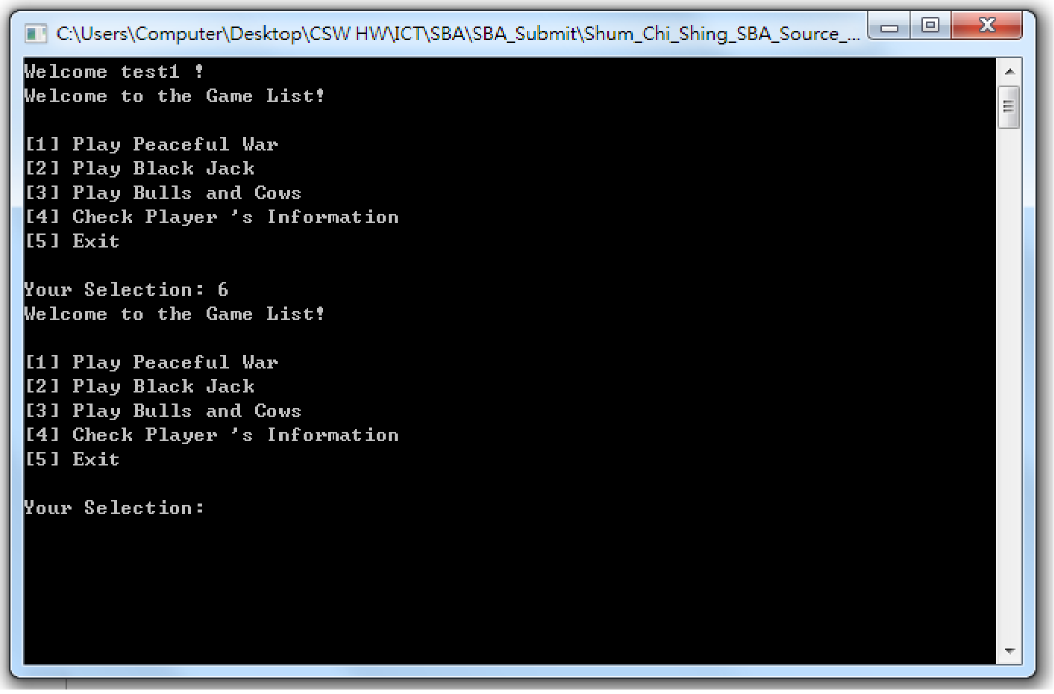
The window size of the program is 80*25.



C:\Users\Computer\Desktop\CSW HW\ICT\SBA\SBA_Submit\Shum_Chi_Shing_SBA_Source_...

```
Welcome test1 !
Welcome to the Game List!

[1] Play Peaceful War
[2] Play Black Jack
[3] Play Bulls and Cows
[4] Check Player 's Information
[5] Exit

Your Selection: _
```

A menu is used to give instruction and guidelines to its users.



C:\Users\Computer\Desktop\CSW HW\ICT\SBA\SBA_Submit\Shum_Chi_Shing_SBA_Source_...

```
Welcome test1 !
Welcome to the Game List!

[1] Play Peaceful War
[2] Play Black Jack
[3] Play Bulls and Cows
[4] Check Player 's Information
[5] Exit

Your Selection: 6
Welcome to the Game List!

[1] Play Peaceful War
[2] Play Black Jack
[3] Play Bulls and Cows
[4] Check Player 's Information
[5] Exit

Your Selection:
```

Same and repeated instructions and guidelines will be shown if invalid input is found.

3.4 Process Implementation

Implementation of the process can be described through the explanation of every procedure. The function and characteristics of every procedure will be mentioned one by one in the following part.

1) Data File Reader (Used Procedure: ReadFile; Program Line 28-48)

Function: To read all the information stored in text file into the array list of program.

Algorithm features:

1) The statement 'while not eof(data)' is used to read the text file without losing any information in all cases.

2) Array used in storing information of text file is 'user = array[1..100] of userinfo' while the 'userinfo' is a record of different variables.
(*For the list of variables, please refer to 3.2 Database Implementation.*)

2) Data Saving (Used Procedure: SaveFile; Program Line 50-64)

Function: To save all the players' records from the array list in the program into the text file.

Algorithm features:

1) The pointer 'code' is used in counting the number of records to be written so it is guaranteed that there will be no missing after every running of procedure 'SaveFile'.

2) The procedure will be run after running every game so that records can be saved immediately after its update.

3) Player's Information Checking (Used Procedure: Info; Program Line 676-690)

Function: To provide a page for users to check all their own information.

Algorithm features

1) The procedure mainly used 'writeln' to show the information.

2) 'Clrscr' is used to clear the screen after user leave the page.

4) Login and Registration System (Main Program; Program Line 698-758)

Function: To provide login and registration services to users.

Algorithm features:

1) The login and registration system is written in the main program in order to prevent bugs and errors when calling other procedures.

2) The login system and the registration system are linked so as to let users to register a new account if they cannot access to their old one.

5) Menu System (Main Program; Program Line 761-782)

Function: To provide a menu for users to use the functions of the system.

Algorithm features:

1) The function 'repeat' is used to show the page and let users select the function of the system.

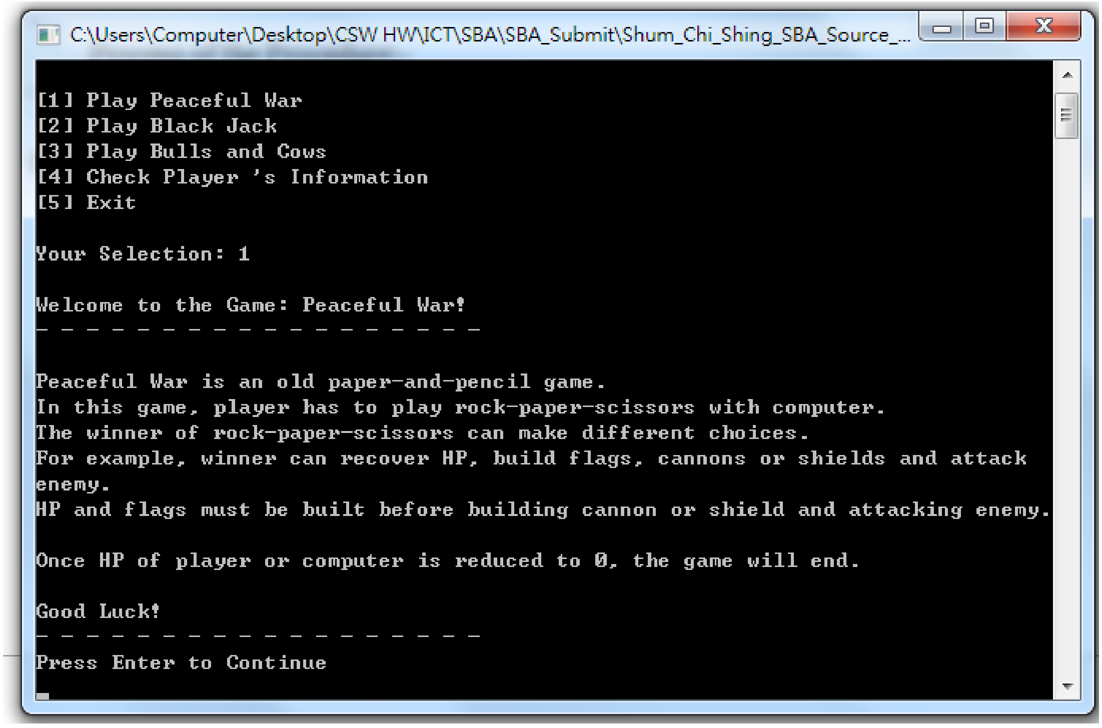
2) When users decide to end the system, 'clrscr' function is used to clear the page and a message stating 'Press Enter to Continue' is displayed. Meanwhile, 'SaveFile' is run to ensure no data is missed from updating to text file.

6) Peaceful War Game (Used Procedure: Peaceful_War; Program Line 66-339)

This is the first game to be provided in the system. The game, Peaceful War, is designed in this procedure. The running process of the game will be explained in the following part.

Process of the Procedure:

1) Input '1' in the menu so as to select the game Peaceful War. Once the game is selected, a clear explanation of the game is shown in the page.

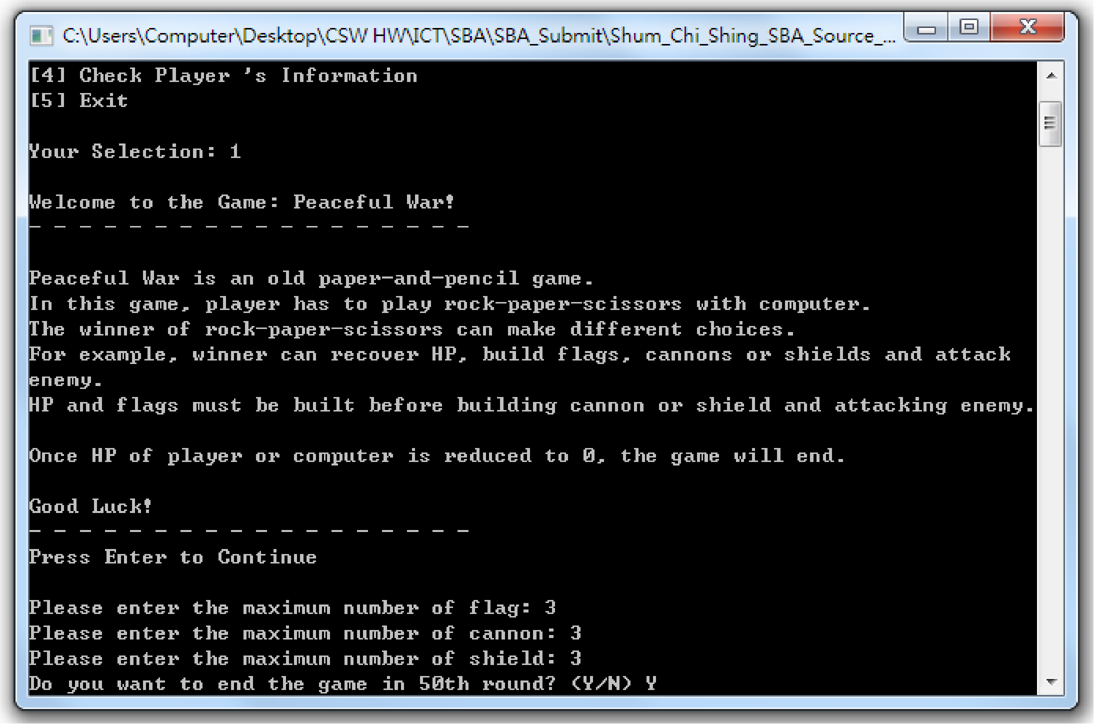


C:\Users\Computer\Desktop\CSW HW\ICT\SBA\SBA_Submit\Shum_Chi_Shing_SBA_Source_...

```
[1] Play Peaceful War
[2] Play Black Jack
[3] Play Bulls and Cows
[4] Check Player 's Information
[5] Exit

Your Selection: 1

Welcome to the Game: Peaceful War!
- - - - - - - - - - - - - - - - - -

Peaceful War is an old paper-and-pencil game.
In this game, player has to play rock-paper-scissors with computer.
The winner of rock-paper-scissors can make different choices.
For example, winner can recover HP, build flags, cannons or shields and attack
enemy.
HP and flags must be built before building cannon or shield and attacking enemy.

Once HP of player or computer is reduced to 0, the game will end.

Good Luck!
- - - - - - - - - - - - - - - - - -
Press Enter to Continue
```

Page showing the Process 1

2) After pressing 'Enter' to continue, users have to set the rules of the game. There are totally 4 rules to be set by the users. They are the maximum number of flag, cannon and shield and whether the game should be ended in the 50th round. Users need to input the setting one by one according to the instructions given by the system.

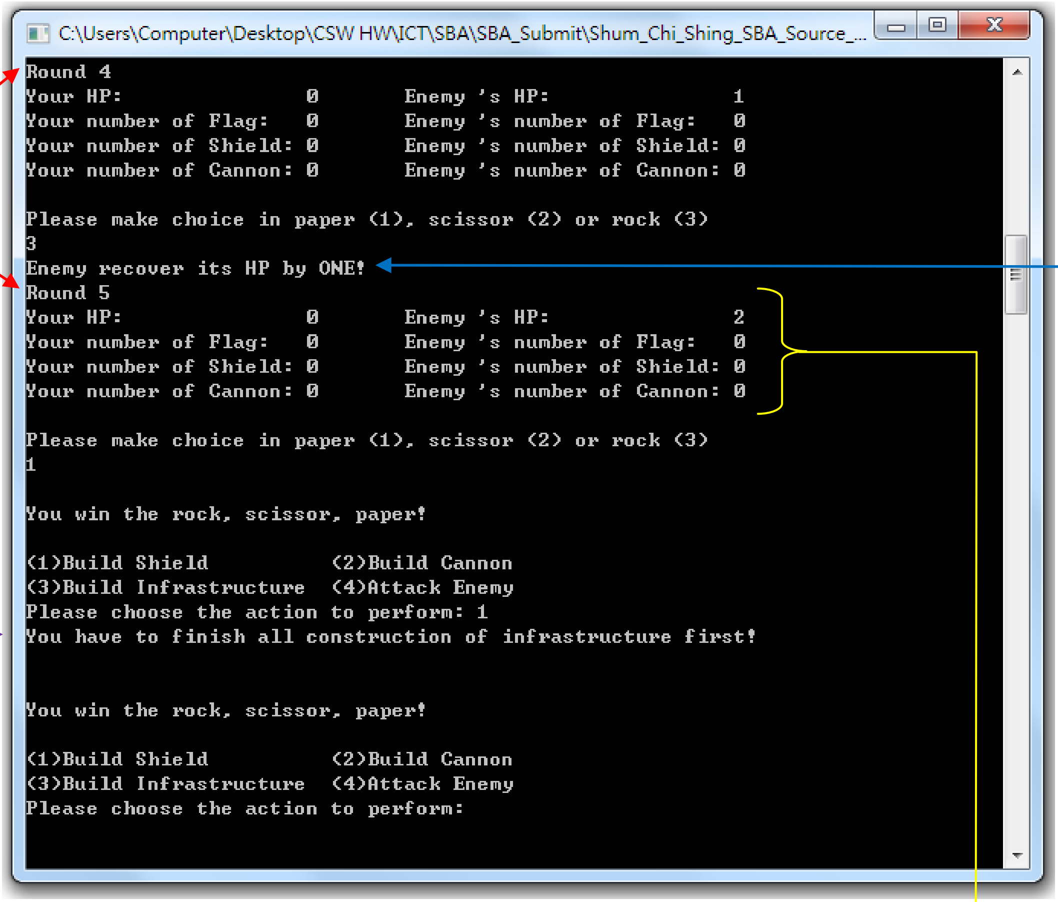


C:\Users\Computer\Desktop\CSW HW\ICT\SBA\SBA_Submit\Shum_Chi_Shing_SBA_Source_...

```
[4] Check Player 's Information
[5] Exit

Your Selection: 1

Welcome to the Game: Peaceful War!
- - - - - - - - - - - - - - - - - -

Peaceful War is an old paper-and-pencil game.
In this game, player has to play rock-paper-scissors with computer.
The winner of rock-paper-scissors can make different choices.
For example, winner can recover HP, build flags, cannons or shields and attack
enemy.
HP and flags must be built before building cannon or shield and attacking enemy.

Once HP of player or computer is reduced to 0, the game will end.

Good Luck!
- - - - - - - - - - - - - - - - - -
Press Enter to Continue

Please enter the maximum number of flag: 3
Please enter the maximum number of cannon: 3
Please enter the maximum number of shield: 3
Do you want to end the game in 50th round? (Y/N) Y
```

Page showing the Process 2

3) Once users finished setting the game rules, the game will start immediately. Player has to select its choice in Rock-Paper-Scissors. If player wins, he/she can choose to have different actions such as constructing shield, cannon or flag, recovering HP or attacking enemy. However, under no circumstance can player take actions other than '(3) Build Infrastructure' if they do not have full HP and Flags. On the other hand, if computer wins, its action will be taken automatically and it has to follow the same rules of the game, which is having full HP and Flags before taking other actions.



C:\Users\Computer\Desktop\CSW HW\ICT\SBA\SBA_Submit\Shum_Chi_Shing_SBA_Source_...

```
Round 4
Your HP:                0       Enemy 's HP:            1
Your number of Flag:    0       Enemy 's number of Flag:   0
Your number of Shield: 0        Enemy 's number of Shield: 0
Your number of Cannon: 0        Enemy 's number of Cannon: 0

Please make choice in paper (1), scissor (2) or rock (3)
3
Enemy recover its HP by ONE!
Round 5
Your HP:                0       Enemy 's HP:            2
Your number of Flag:    0       Enemy 's number of Flag:   0
Your number of Shield: 0        Enemy 's number of Shield: 0
Your number of Cannon: 0        Enemy 's number of Cannon: 0

Please make choice in paper (1), scissor (2) or rock (3)
1

You win the rock, scissor, paper!

(1)Build Shield          (2)Build Cannon
(3)Build Infrastructure  (4)Attack Enemy
Please choose the action to perform: 1
You have to finish all construction of infrastructure first!


You win the rock, scissor, paper!

(1)Build Shield          (2)Build Cannon
(3)Build Infrastructure  (4)Attack Enemy
Please choose the action to perform:
```

From the page above, there are several elements shown.

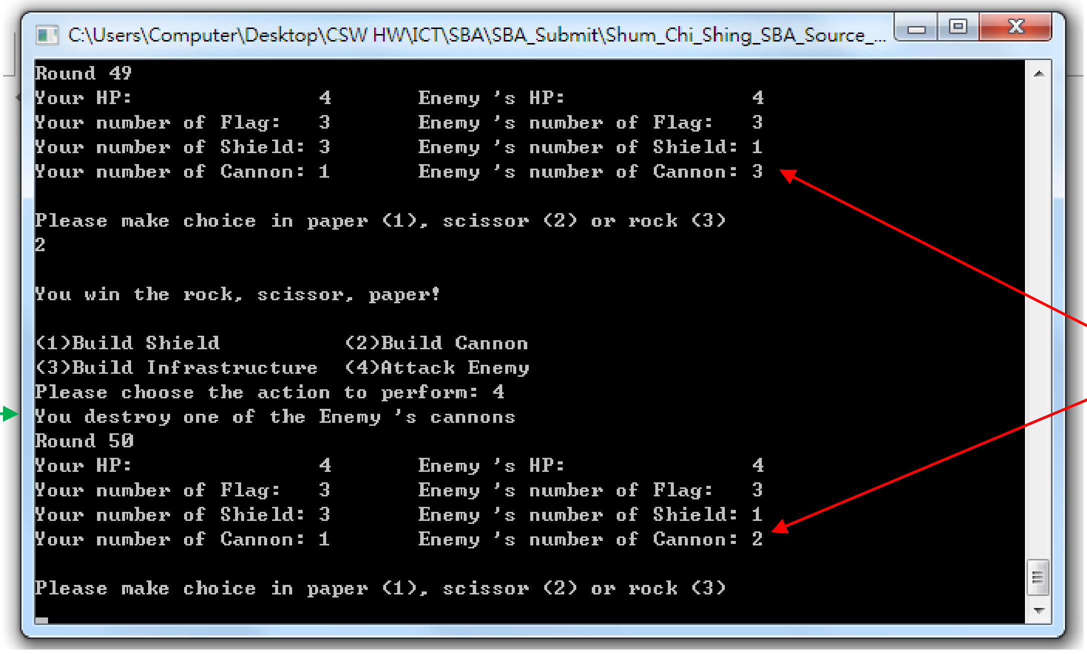
- The number of round is shown.

- The statistics of both player and computer are shown.

- 'Asking for choice in Rock-Paper-Scissors' statement is shown straight after showing the status.

- If computer wins, its action taken will be shown before starting next round.

- If player wins but the action taken is invalid, a warning stating 'You have to finish all construction of infrastructure first!' will be shown and the list of actions is repeated to shown.


C:\Users\Computer\Desktop\CSW HW\ICT\SBA\SBA_Submit\Shum_Chi_Shing_SBA_Source_...

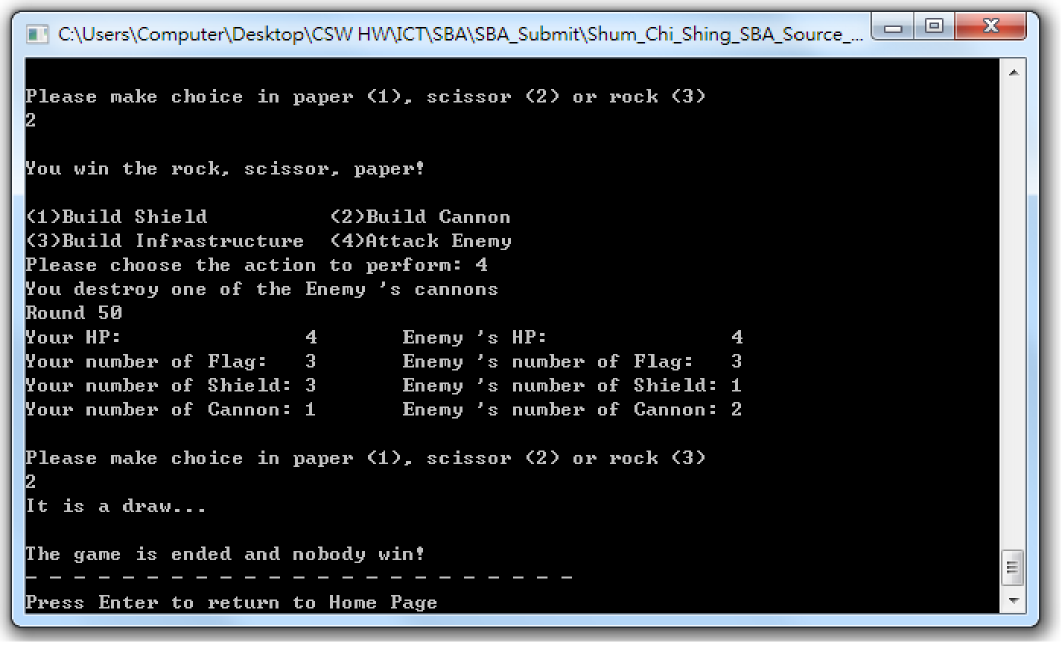
```
Round 49
Your HP:               4       Enemy 's HP:               4
Your number of Flag:   3       Enemy 's number of Flag:   3
Your number of Shield: 3       Enemy 's number of Shield: 1
Your number of Cannon: 1       Enemy 's number of Cannon: 3

Please make choice in paper (1), scissor (2) or rock (3)
2

You win the rock, scissor, paper!

(1)Build Shield          (2)Build Cannon
(3)Build Infrastructure  (4)Attack Enemy
Please choose the action to perform: 4
You destroy one of the Enemy 's cannons
Round 50
Your HP:               4       Enemy 's HP:               4
Your number of Flag:   3       Enemy 's number of Flag:   3
Your number of Shield: 3       Enemy 's number of Shield: 1
Your number of Cannon: 1       Enemy 's number of Cannon: 2

Please make choice in paper (1), scissor (2) or rock (3)
```

From the above page, there are two elements to be shown

- The change in statistics of computer is shown

- The action of attacking enemy is shown.

4) The page below shows the end of the game. Since the game is set to end after 50th round, the statement 'The game is ended and nobody win!' is shown. After that, the procedure 'SaveFile' is run to update the information in text files. 'Press Enter to return to Home Page' statement is shown to instruct player to return to menu.


C:\Users\Computer\Desktop\CSW HW\ICT\SBA\SBA_Submit\Shum_Chi_Shing_SBA_Source_...

```

Please make choice in paper (1), scissor (2) or rock (3)
2

You win the rock, scissor, paper!

(1)Build Shield          (2)Build Cannon
(3)Build Infrastructure  (4)Attack Enemy
Please choose the action to perform: 4
You destroy one of the Enemy 's cannons
Round 50
Your HP:               4       Enemy 's HP:               4
Your number of Flag:   3       Enemy 's number of Flag:   3
Your number of Shield: 3       Enemy 's number of Shield: 1
Your number of Cannon: 1       Enemy 's number of Cannon: 2

Please make choice in paper (1), scissor (2) or rock (3)
2
It is a draw...

The game is ended and nobody win!
- - - - - - - - - - - - - - - - - - - - - - -
Press Enter to return to Home Page
```

Algorithm features:

1) 'Randomize' is used in the procedure to ensure that the choice and the action taken by the computer will be different every round.

2) Initialization is done straight after showing the game description in the page and before any input except the 'readln' for player to continue from the description.

3) 'Repeat' loop is used to make the game continuing running before it comes to an end.

4) 3 separated 'if' statements which contain 3 different rules are used to simulate all situations in the Rock-Paper-Scissors in order to process the winner of Rock-Paper-Scissors.

5) 'Repeat' loop and 'nested-if' statement are used to ensure player making valid action after winning Rock-Paper-Scissors. The flag 'mvalidmv' is used to indicate the valid action of player. This part of designation is also applied in action taken by computer while the flag 'evalidmv' is used to indicate whether the action generated is valid or not.

6) The 'case' statement is used to recognise the action taken by the player.

7) If the action of attacking enemy is taken by the player, another 'case' statement and a 'random' statement will be used to select the target to be attacked. The 'random' statement will generate a code within 1 to 100. If the code is in the range of 1 to 45, the enemy's cannon will be the attacking target. If the code lies in the range of 46 to 90, there are several objects can be the target according to their orders. The shield will be the target if it exists. If not, the flag will be the second choice and the enemy's HP will be the final target. After all, if the code is within 91 to 100, the attack will come to a miss. This part of designation is also applied in action taken by computer.

8) The flag 'limround' and the counter 'step' are used to check whether the game is end when player sets that the game is going to end after 50th round.

9) If the rule that the game will end after 50th round is not activated, the variables 'mhit', 'ehit', 'mHP' and 'eHP' are used to defined whether player wins the game or loses the game. The variable 'mhit' and 'ehit' represent the times player and computer attack each other successfully respectively. The variable 'mHP' and 'eHP' represent the HP of player and computer respectively. The player wins when he/she successfully attacks enemy at least once and enemy's HP becomes negative. On the other hand, the player loses when computer successfully attacks player at least once and HP of player becomes negative.

10) The record of player in the array list will be updated immediately after the 'if' statement deciding the ending of game is achieved. After that, the procedure 'SaveFile' is run straight after the declaration of game ending in order to update the records of players in the text file immediately.

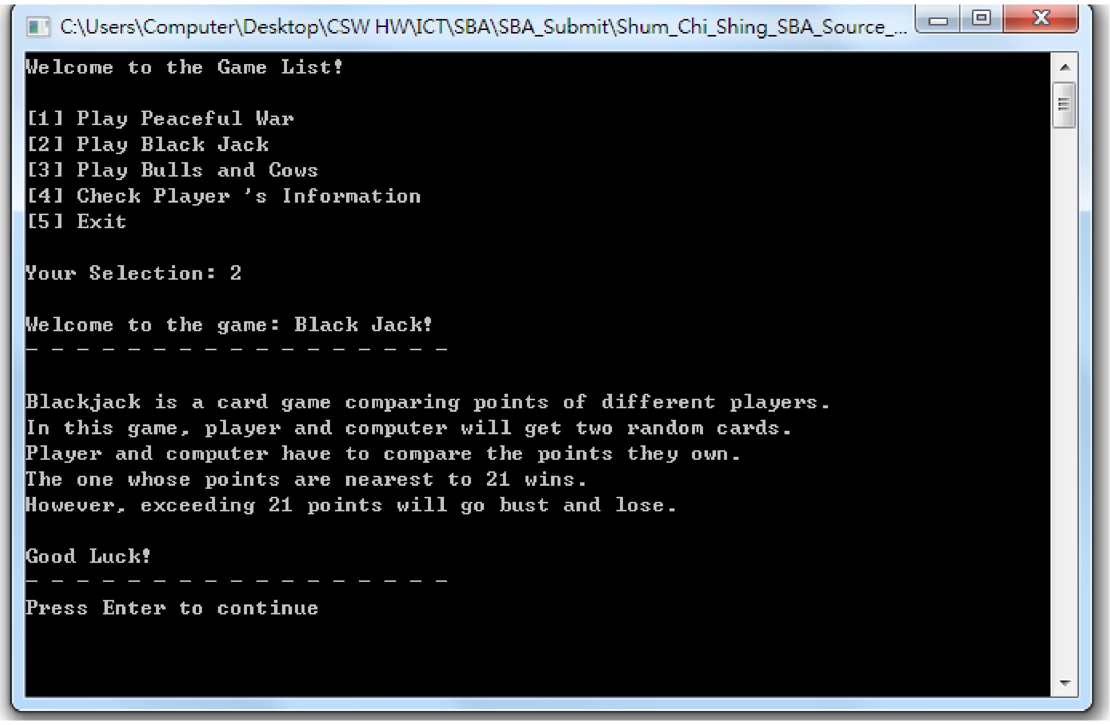
11) At the end of the game, the statement 'Press Enter to return to Home Page' is shown to instruct user to return to menu. The function 'clrscr' is used to clear the game process in the page so as to provide a clean menu page to the user of the system.

7) Blackjack Game (Used Procedure: Black_Jack; Program Line 341-562)

This is the second game to be provided in the system. The game, Blackjack, is designed in this procedure. The running process of the game will be explained in the following part.

Process of the Procedure:

1) Input '2' in the menu so as to select the game Blackjack. Once the game is selected, a clear explanation of the game is shown in the page.

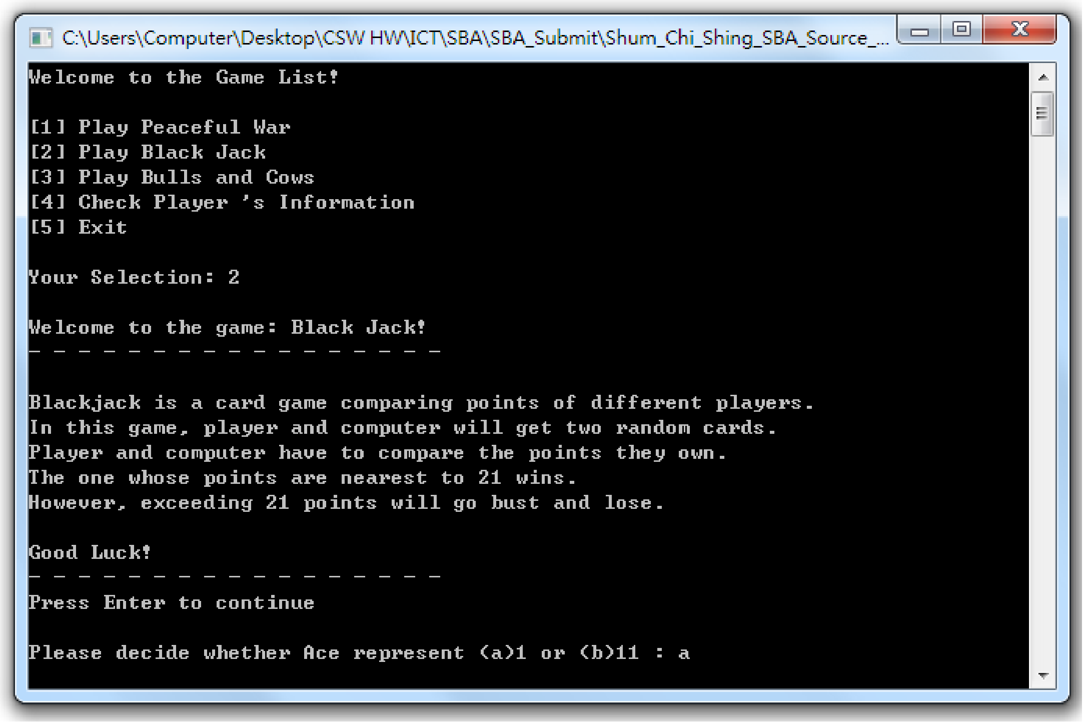


C:\Users\Computer\Desktop\CSW HW\ICT\SBA\SBA_Submit\Shum_Chi_Shing_SBA_Source_...

```
Welcome to the Game List!

[1] Play Peaceful War
[2] Play Black Jack
[3] Play Bulls and Cows
[4] Check Player 's Information
[5] Exit

Your Selection: 2

Welcome to the game: Black Jack!
- - - - - - - - - - - - - - - - -

Blackjack is a card game comparing points of different players.
In this game, player and computer will get two random cards.
Player and computer have to compare the points they own.
The one whose points are nearest to 21 wins.
However, exceeding 21 points will go bust and lose.

Good Luck!
- - - - - - - - - - - - - - - - -
Press Enter to continue
```

Page showing the Process 1

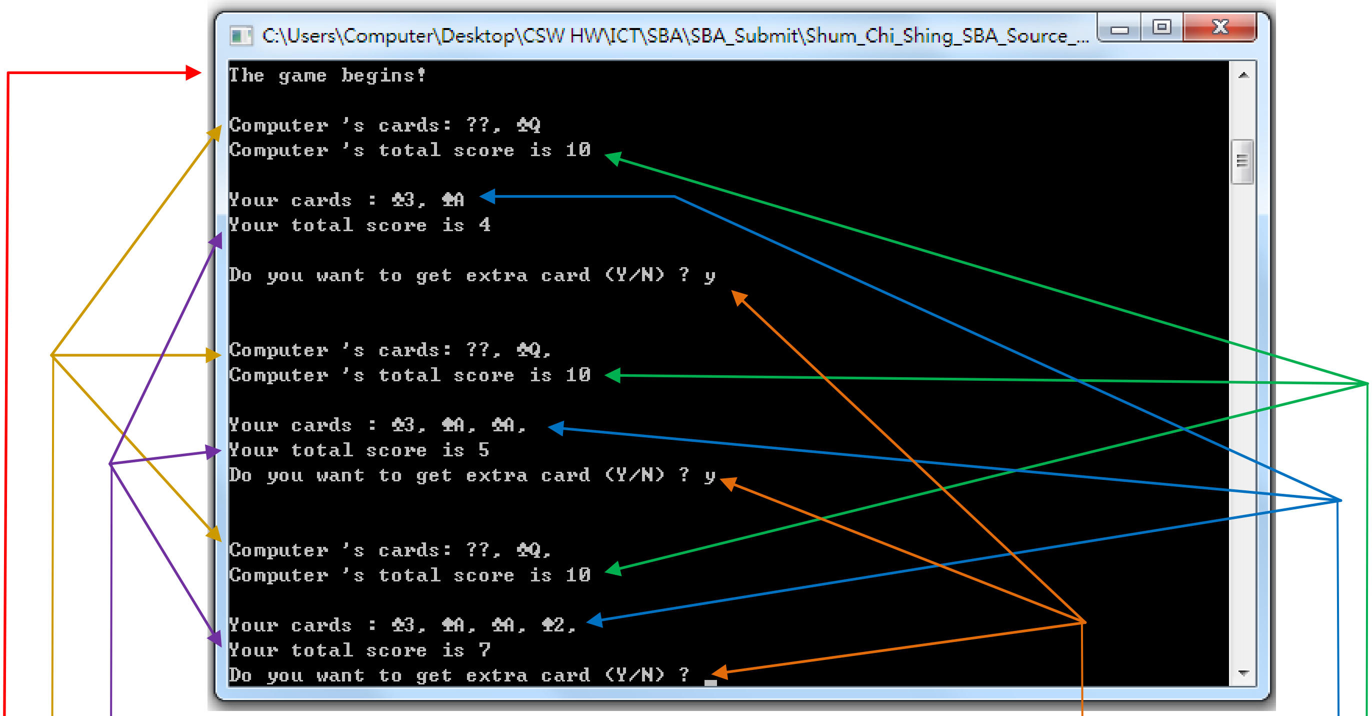
2) After pressing 'Enter' to continue, users have to set the rules of the game. There is only one rule needing player to set, which is deciding whether Ace card should contain 1 point or 11 points. If player hopes Ace card contains 1 point, 'a' is needed to be inputted while 'b' is needed to be inputted for deciding that Ace card should contain 11 points.


C:\Users\Computer\Desktop\CSW HW\ICT\SBA\SBA_Submit\Shum_Chi_Shing_SBA_Source_...

```
Welcome to the Game List!

[1] Play Peaceful War
[2] Play Black Jack
[3] Play Bulls and Cows
[4] Check Player 's Information
[5] Exit

Your Selection: 2

Welcome to the game: Black Jack!
- - - - - - - - - - - - - - - - -

Blackjack is a card game comparing points of different players.
In this game, player and computer will get two random cards.
Player and computer have to compare the points they own.
The one whose points are nearest to 21 wins.
However, exceeding 21 points will go bust and lose.

Good Luck!
- - - - - - - - - - - - - - - - -
Press Enter to continue

Please decide whether Ace represent (a)1 or (b)11 : a
```

Page showing the Process 2

3) Once users finished setting the game rules, the game will start immediately. The statement 'The game begins!' is shown after setting the rules. It indicates the starting of the game. After that, there are several elements to be shown in the page. They are a card own by computer, the total score of computer, the first two cards you have and your total score. A question asking you whether extra card is needed is appeared below the cards and the total scores of computer and you. Inputting 'Y' or 'y' can let you obtain one extra card while inputting 'N' or 'n' will stop you from obtaining extra card and skip to computer's turn for getting extra card. Player will have no chance for getting extra card once 'N' or 'n' is inputted.



C:\Users\Computer\Desktop\CSW HW\ICT\SBA\SBA_Submit\Shum_Chi_Shing_SBA_Source_...

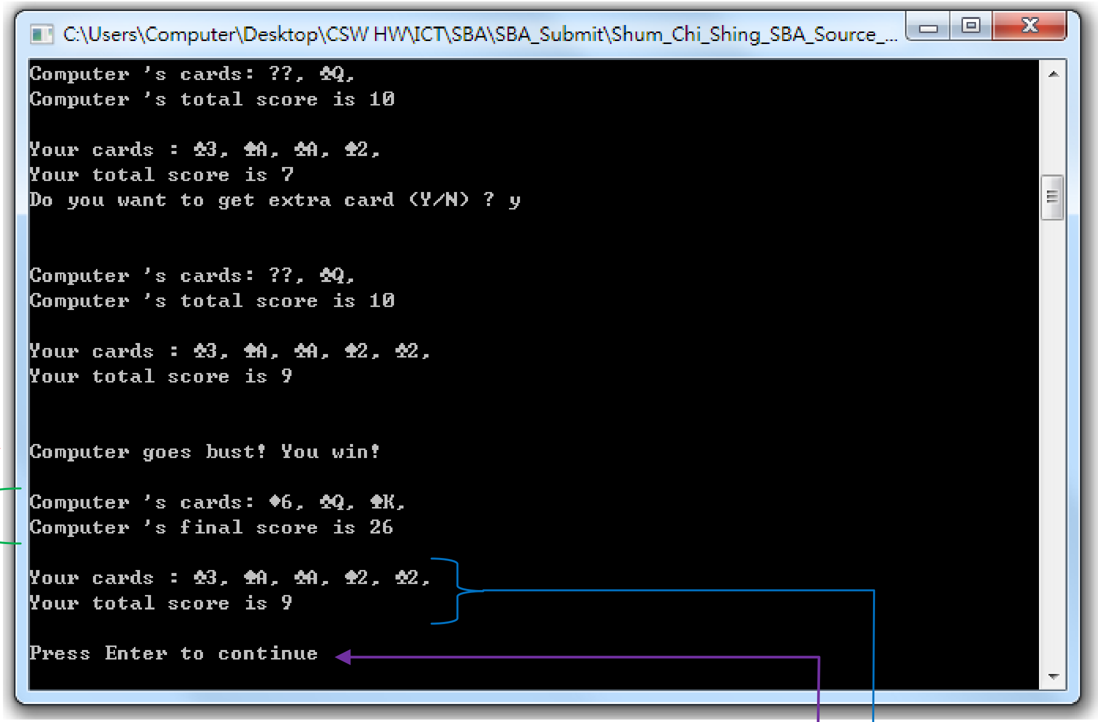
```
The game begins!

Computer 's cards: ??, ♧Q
Computer 's total score is 10

Your cards : ♧3, ♠A
Your total score is 4

Do you want to get extra card (Y/N) ? y


Computer 's cards: ??, ♧Q,
Computer 's total score is 10

Your cards : ♧3, ♠A, ♤A,
Your total score is 5
Do you want to get extra card (Y/N) ? y


Computer 's cards: ??, ♧Q,
Computer 's total score is 10

Your cards : ♧3, ♠A, ♤A, ♠2,
Your total score is 7
Do you want to get extra card (Y/N) ? _
```

Page showing the Process 3

From the above page, there are several elements shown as mentioned in Process 3.

- 'The game begins!' statement
- Viewable cards owned by computer
- Computer's total score
- Cards owned by player
- Player's total score
- Question asking for extra card

4) After getting an extra card, the system will check whether player goes bust and the number of cards player owns. If player doesn't go bust and he/she gets less than 5 cards, the question asking for extra card will keep showing until anyone of the above rules is satisfied. If yes, the system will automatically stop asking player about obtaining extra card and skip to the round for computer to get extra card. The rule for computer to get extra card is that keep obtaining extra card until computer's total score meet 17 or 5 cards are owned by computer. After this section, the system will compare the total score of computer and the one of player. The one who get highest score without going bust will win the game. After comparison, the system will declare the ending of the game and show all cards of computer and player as well as their total scores. Lastly, the procedure 'SaveFile' will be run and the statement 'Press Enter to continue' will be shown in the page. After pressing 'Enter', the function 'clrscr' will be run and player will return to clean menu page as it is the end of the game.


C:\Users\Computer\Desktop\CSW HW\ICT\SBA\SBA_Submit\Shum_Chi_Shing_SBA_Source_...

```
Computer 's cards: ??, ♤Q,
Computer 's total score is 10

Your cards : ♤3, ♠A, ♤A, ♠2,
Your total score is 7
Do you want to get extra card (Y/N) ? y


Computer 's cards: ??, ♤Q,
Computer 's total score is 10

Your cards : ♤3, ♠A, ♤A, ♠2, ♤2,
Your total score is 9


Computer goes bust! You win!

Computer 's cards: ♦6, ♤Q, ♠K,
Computer 's final score is 26

Your cards : ♤3, ♠A, ♤A, ♠2, ♤2,
Your total score is 9

Press Enter to continue
```

Page showing the Process 3

From the above page, several elements are shown.

- Ending statement

- Computer's all cards and total score

- Player's all cards and total score

- 'Press Enter to continue' statement

Algorithm Features:

1) 'Randomize' is used in the procedure to ensure that the card obtained by the computer and player will be different in every round.

2) Initialization is done straight after showing the game description in the page and before any input except the 'readln' for player to continue from the description.

3) ‘While’ loop is used to ensure player input a valid character when setting the game rule of the game.

4) The flag ‘ace11’ is used to hold the game rule that whther Ace card contains 1 point or 11 points.

5) ‘For’ loop and ‘if’ statements are used to generate all the cards and its points containing in the array lists ‘cdnum’ and ‘cdtext’. The ‘if’ statements are used to modify the texts and the values stored in the cards. For example, the value for Ace cards will be changed to 11 if the rule that Ace card contains 11 points is activated. The texts of Ace, King, Queen and Jack cards will be changed to ‘A’, ‘K’, ‘Q’ and ‘J’ instead of ‘1’, ‘11’, ‘12’ and ‘13’.

6) Before player seeing any card, the 10 cards that are going to be distributed to player and computer will be generated. In fact, the action that player asks for extra card is asking for showing the next card generated. The array list 'usedcd' will be used to check whether the card randomly selected has been included in the list of those 10 cards before. The array lists 'mcd', 'mvalue', 'ecd', 'evalue', 'msum' and 'esum' will be used to hold player's cards, player's score of different cards, computer's cards, computer's score of different cards, player's total score and computer's total score respectively. All of the above variable will be shown to player except the 'esum', which will be replaced by the variable 'eshowsum'. The variable 'eshowsum' is used to hold computer's score of its owned cards except the first card. 'Repeat' loop and 'for' loop are used to generate the 10 cards of player and computer.

7) 'Repeat' loop and 'if' statement are used to ask player for obtaining the extra card. If yes while player does not get five cards and does not go bust, the system will show the next card stored in the array list 'mcd' to player and the total score of player will be calculated again. After that, the statistics of both computer and player will be printed out again in the page. Meanwhile, the pointer 'mcount' will be added by 1 to indicate the number of cards player owns. After printing the statistics, the system will check whether player goes bust or not. This 'repeat' loop will continue running until player goes bust, 5 cards are owned by player or player refuses to obtain extra card anymore.

8) 'While' loop is used to determine whether computer is needed to obtain extra card or not. The rules that the loop checks are whether the total score owned by computer is less than 17 and the number of computer's cards is less than 5. The loop will continue running and no information will be updated in the page until the rule is not satisfied.

9) After the rounds for player and computer to obtain extra cards, a sequence of 'if' statements are used to determine the winner of the game. There are several variable flags used in these statements. The flags 'mex' and 'eex' are used to indicate whether player goes bust or computer goes bust respectively. The flags 'mwin', 'ewin' and 'draw' are used to determined whether the game is won by player or computer or it is a draw.

10) After determining the ending of the game, the system will declare the ending of the game and print the message in the page. Meanwhile, the player's records stored in the array list of the program will be updated automatically.

11) All the statistics of both computer and player will be shown straight after showing the ending of the game. Player can view all the cards of computer, the total score of computer including its first card, player's own cards and total score.

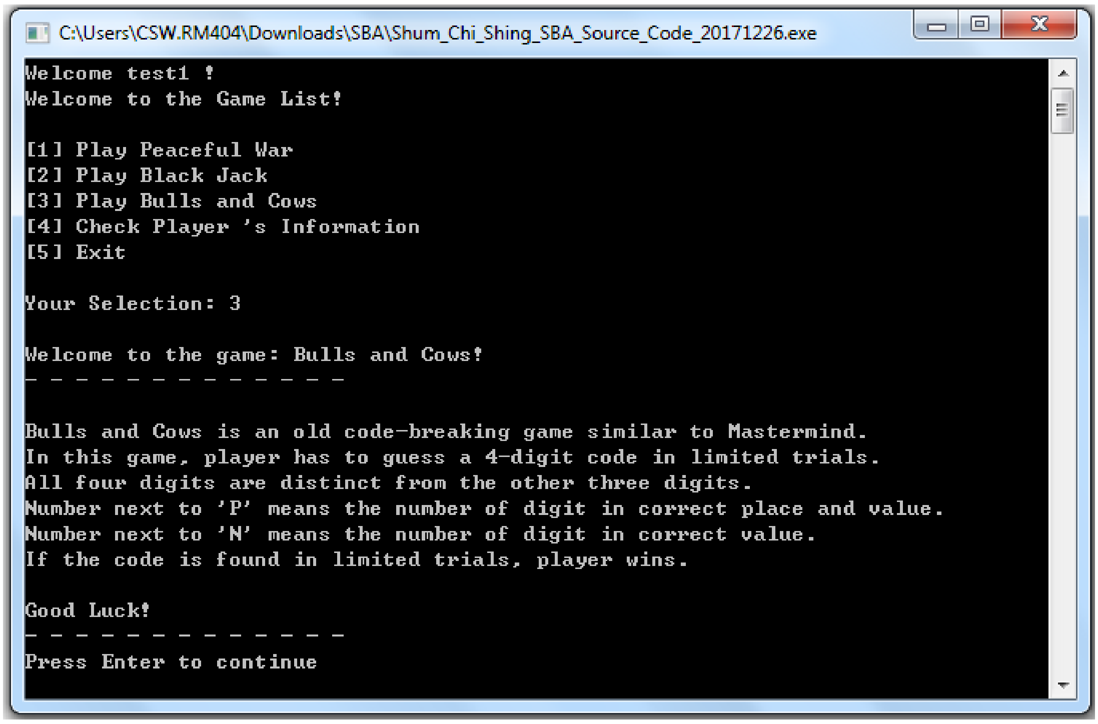
12) The procedure 'SaveFile' will be run automatically and the 'Press Enter to continue' will exist in an attempt to instruct player to return to the menu page. After user input the 'Enter' key, the function 'clrscr' will be run in order to clear the page of game process and return player a clean menu page.

8) Bulls and Cows Game (Used Procedure: Bulls_and_Cows; Program Line 564-674)

This is the third game as well as the last game provided in the system. The game, Bulls and Cows, is designed in this procedure. The running process of the game will be explained in the following part.

Process of the Procedure:

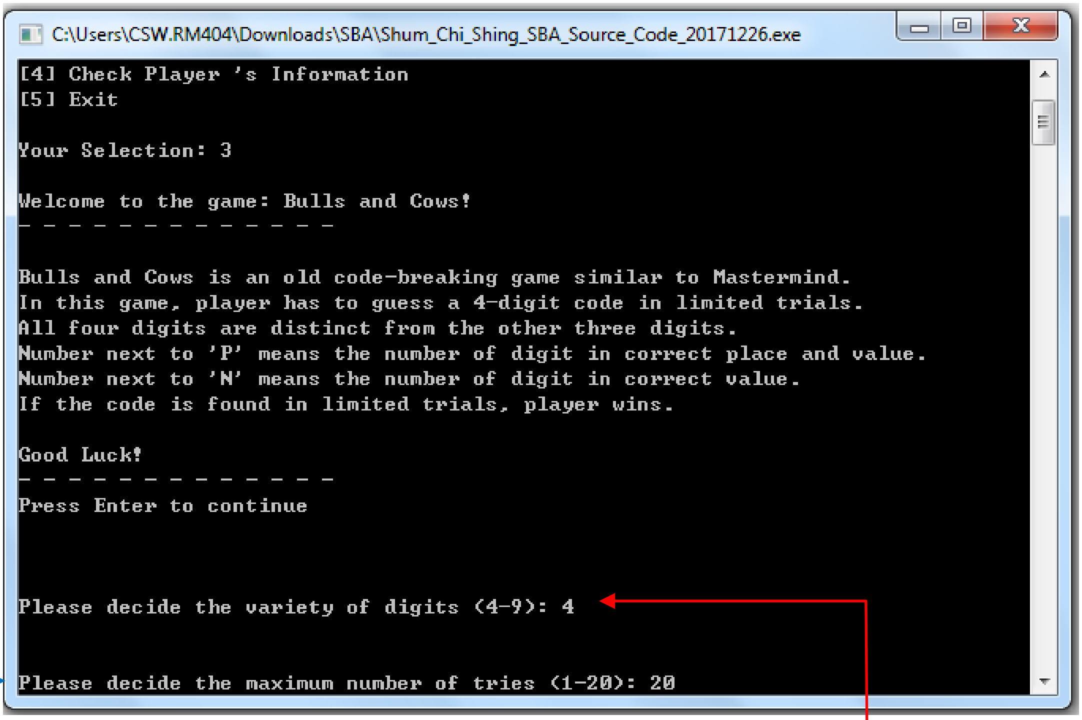
1) Input '3' in the menu so as to select the game Bulls and Cows. Once the game is selected, a clear explanation of the game is shown in the page.


C:\Users\CSW.RM404\Downloads\SBA\Shum_Chi_Shing_SBA_Source_Code_20171226.exe

```
Welcome test1 !
Welcome to the Game List!

[1] Play Peaceful War
[2] Play Black Jack
[3] Play Bulls and Cows
[4] Check Player 's Information
[5] Exit

Your Selection: 3

Welcome to the game: Bulls and Cows!
- - - - - - - - - - - - -

Bulls and Cows is an old code-breaking game similar to Mastermind.
In this game, player has to guess a 4-digit code in limited trials.
All four digits are distinct from the other three digits.
Number next to 'P' means the number of digit in correct place and value.
Number next to 'N' means the number of digit in correct value.
If the code is found in limited trials, player wins.

Good Luck!
- - - - - - - - - - - - -
Press Enter to continue
```

Page showing the Process 1

2) After pressing 'Enter' to continue, users have to set the rules of the game. Player has to set the variety of the digits of the code, which is going to be guessed by the player in the following game. The variety of the code means that the digits will vary from the range of 1 to 4 to the range of 1 to 9. After setting the variety, the player has to set the maximum of tries of the game. It means how many times player can guess the code.


C:\Users\CSW.RM404\Downloads\SBA\Shum_Chi_Shing_SBA_Source_Code_20171226.exe

```
[4] Check Player 's Information
[5] Exit

Your Selection: 3

Welcome to the game: Bulls and Cows!
- - - - - - - - - - - - - -

Bulls and Cows is an old code-breaking game similar to Mastermind.
In this game, player has to guess a 4-digit code in limited trials.
All four digits are distinct from the other three digits.
Number next to 'P' means the number of digit in correct place and value.
Number next to 'N' means the number of digit in correct value.
If the code is found in limited trials, player wins.

Good Luck!
- - - - - - - - - - - - - -
Press Enter to continue



Please decide the variety of digits (4-9): 4


Please decide the maximum number of tries (1-20): 20
```

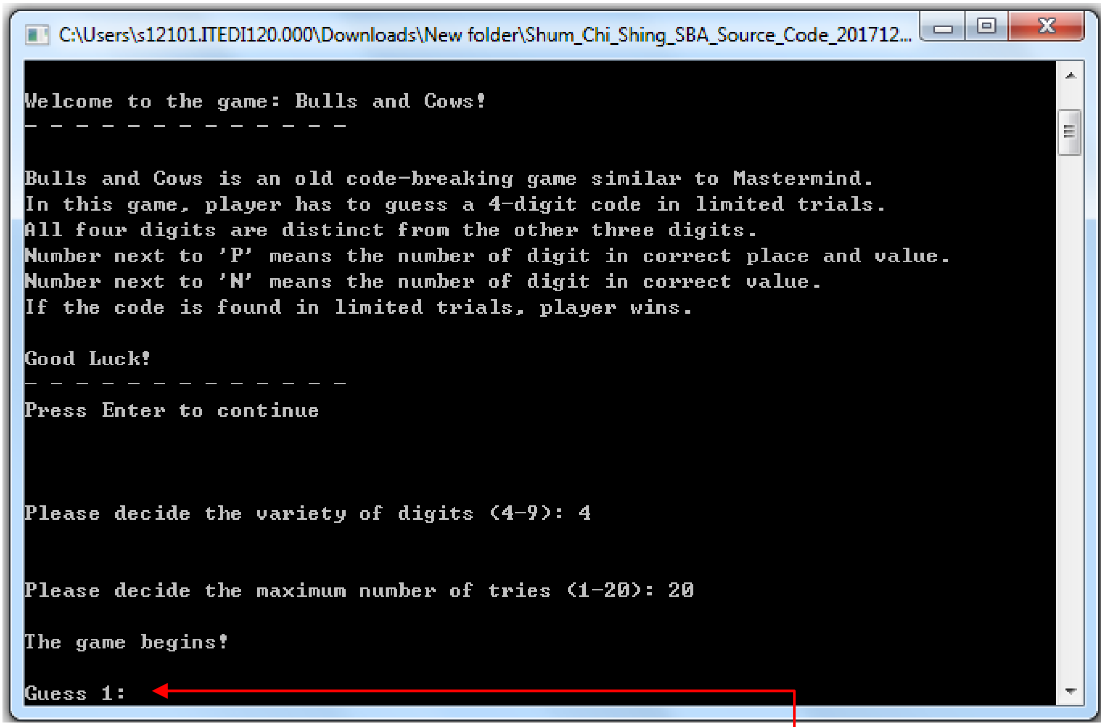
Page showing the Process 2

From the above page, several elements are shown.

- Setting variety of digits of the code

- Setting the maximum times for player to guess

3) Once users finished setting the game rules, the game will start immediately. The statement 'The game begins!' is shown after setting the rules. It indicates the starting of the game. After that, there is a statement ‘Guess 1:’ to be shown in the page. It instructs player to input the guess while the number ‘1’ represents the number of guess.



C:\Users\s12101.ITEDI120.000\Downloads\New folder\Shum_Chi_Shing_SBA_Source_Code_201712...

```
Welcome to the game: Bulls and Cows!
- - - - - - - - - - - - - -

Bulls and Cows is an old code-breaking game similar to Mastermind.
In this game, player has to guess a 4-digit code in limited trials.
All four digits are distinct from the other three digits.
Number next to 'P' means the number of digit in correct place and value.
Number next to 'N' means the number of digit in correct value.
If the code is found in limited trials, player wins.

Good Luck!
- - - - - - - - - - - - - -
Press Enter to continue



Please decide the variety of digits (4-9): 4


Please decide the maximum number of tries (1-20): 20

The game begins!

Guess 1:
```

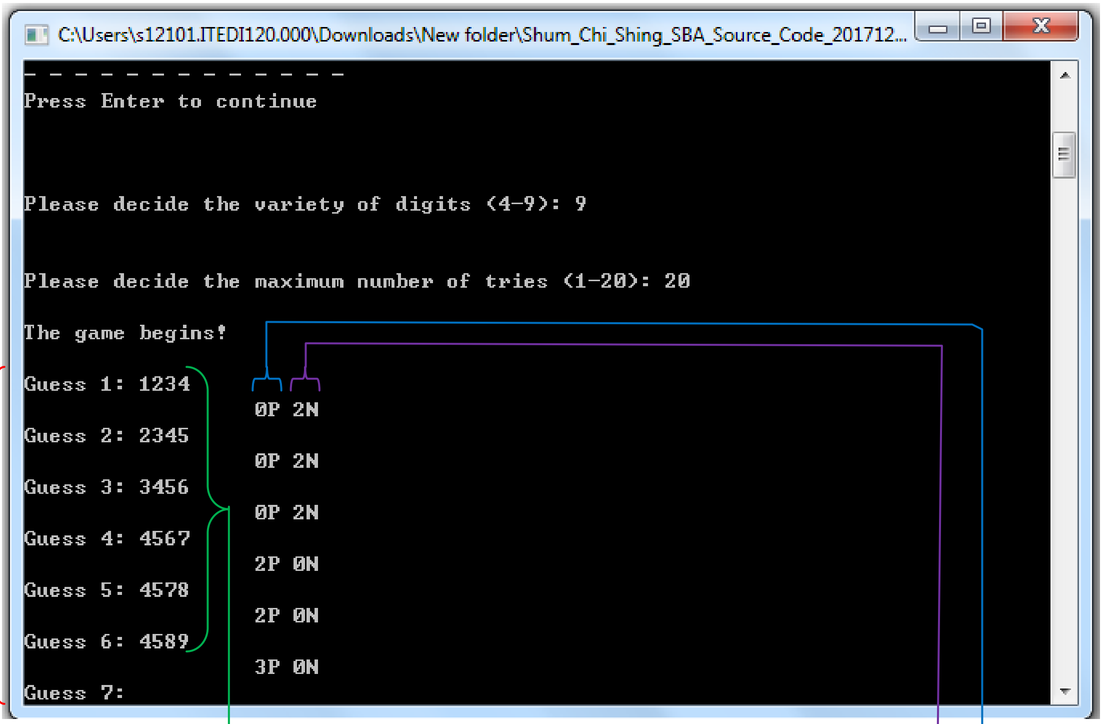
Page showing the Process 3

From the above page, several elements are shown.

- 'The game begins!' statement

- The statement ‘Guess [Number of Guess]:’

4) After seeing the statement of asking for guess, player can now input the 4-digit guess into the system. The system can thus compare the generated code and the guess inputted by the player. After the comparison, the system will print the following elements in the page. The statement ‘Guess [Number of Guess+1]:’, the number of correct digit in the correct place and the number of correct digit but in the wrong place. The number printed on the left side the letter ‘P’ represents the number of correct digit in the correct place. The number printed on the left side the letter ‘N’ represents the number of correct digit but in the wrong place. After viewing the information shown by the system, player can continue inputting the guess one by one until the maximum tries set in the game or the code generated is successfully guessed.


C:\Users\s12101.ITEDI120.000\Downloads\New folder\Shum_Chi_Shing_SBA_Source_Code_201712...

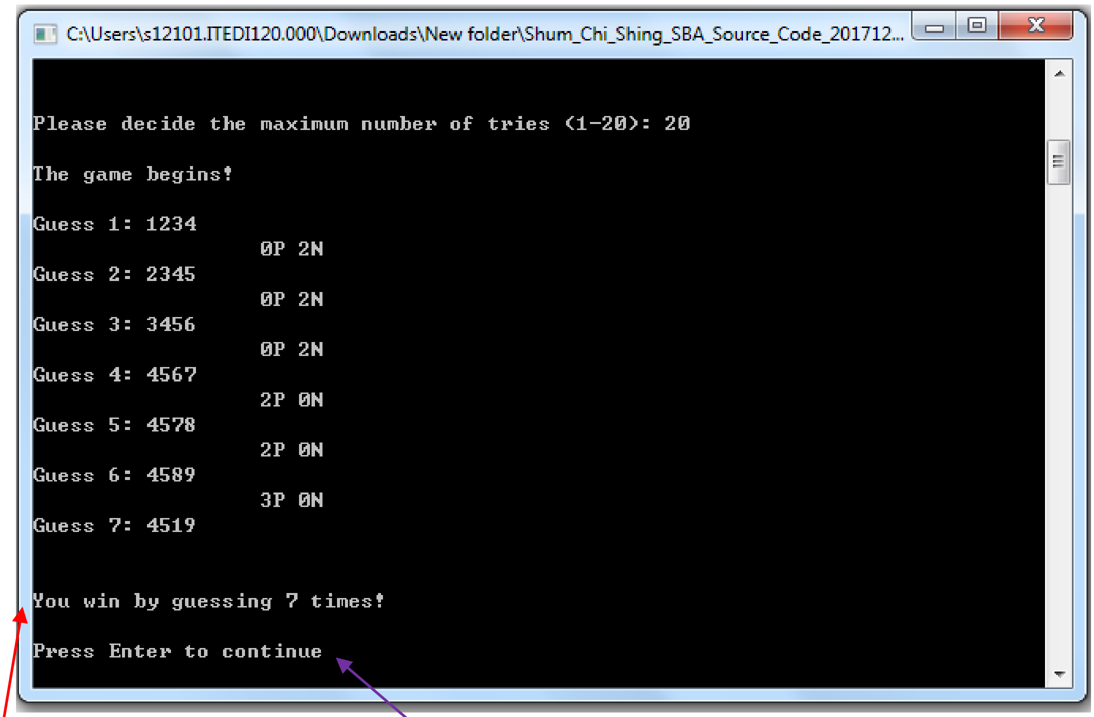
```
- - - - - - - - - - - - - -
Press Enter to continue


Please decide the variety of digits (4-9): 9


Please decide the maximum number of tries (1-20): 20

The game begins!

Guess 1: 1234
                  0P 2N
Guess 2: 2345
                  0P 2N
Guess 3: 3456
                  0P 2N
Guess 4: 4567
                  2P 0N
Guess 5: 4578
                  2P 0N
Guess 6: 4589
                  3P 0N
Guess 7:
```

Page showing the Process 4

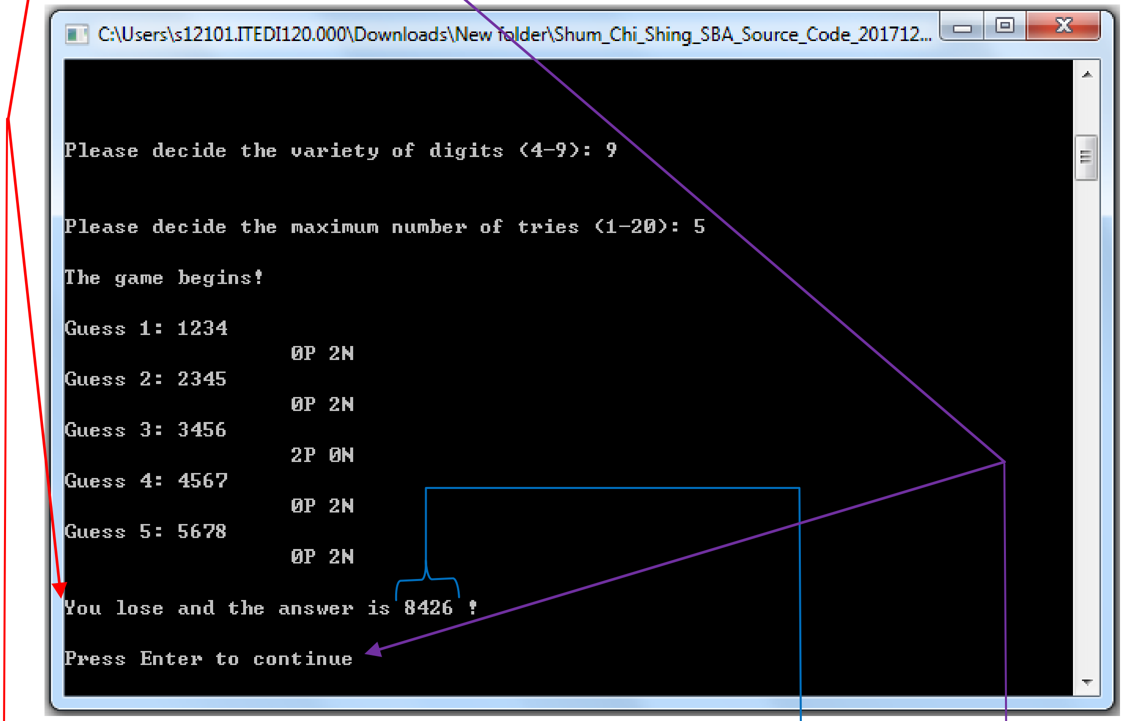
From the above page, several elements are shown.

- The statement ‘Guess [Number of Guess]:’
- The guess inputted by player
- The number of correct digit in the correct place and the letter ‘P’
- The number of correct digit but in the wrong place and the letter ‘N’

5) When player can successfully guess the correct code, the system will stop letting player guess the code anymore and the system will declare the ending of the game. The statement ‘You win by guessing [Number of Guess] times’ will be printed. On the other hand, if player fails to guess the correct code in the limited tries, the statement ‘You lose and the answer is [Correct Code] !’ will be shown. After showing the ending of the game, the system will print the statement ‘Press Enter to continue’ in the page in an attempt to instruct user to return to the menu page. The next page will show the screenshot of the system so as to make a clearer explanation.


C:\Users\s12101.ITEDI120.000\Downloads\New folder\Shum_Chi_Shing_SBA_Source_Code_201712...

```
Please decide the maximum number of tries (1-20): 20

The game begins!

Guess 1: 1234
                  0P 2N
Guess 2: 2345
                  0P 2N
Guess 3: 3456
                  0P 2N
Guess 4: 4567
                  2P 0N
Guess 5: 4578
                  2P 0N
Guess 6: 4589
                  3P 0N
Guess 7: 4519


You win by guessing 7 times!

Press Enter to continue
```


C:\Users\s12101.ITEDI120.000\Downloads\New folder\Shum_Chi_Shing_SBA_Source_Code_201712...

```
Please decide the variety of digits (4-9): 9


Please decide the maximum number of tries (1-20): 5

The game begins!

Guess 1: 1234
                  0P 2N
Guess 2: 2345
                  0P 2N
Guess 3: 3456
                  2P 0N
Guess 4: 4567
                  0P 2N
Guess 5: 5678
                  0P 2N

You lose and the answer is 8426 !

Press Enter to continue
```

Page showing the Process 4

From the above page, several elements are shown.

- The declaration of the ending of the game

- The Answer of the code guessing

- 'Press Enter to continue' statement

Algorithm Features:

1) 'Randomize' is used in the procedure to ensure that the code generated in the game will be different.

2) 'Repeat' loop is used when player is setting the rules of the game. As a result, the input of the player must be valid. The variables 'vardig' and 'tries' is used in recording the variety of digits in the code and the maximum tries in the game.

3) Initialization will be done after player finishes setting the rules of the game so as to avoid errors during the game process. 'For' loops are used in initializing the variables in several array lists.

4) 'For' loop is used in generating the code for guessing. 'Repeat' loop is used in checking whether the digit has been used in generating the code already. The flag 'useddig' is used to indicate whether the digit has been used or not.

5) 'While' loop is used in the main process of the game, which is the part for player to guess the code. The variable 'count' is used in counting how many times player guesses in the game.

6) 'Repeat' loop is used in validating the input of guessing from the player. If the input of guessing is invalid, the 'repeat' loop will not end until player input a valid guess.

7) A double of 'for' loops are used in comparing the guess of player and the code. It allows the system to compare all the digits from the guess and the code one by one. 'If' statement is used in producing the number of correct digit in correct place and the number of correct digit in incorrect place.

8) The flag 'win' is used in indicating whether player win the game or not. The flag is used in the 'while' loop in the main process of the game as the rule deciding the stopping of the game. It is also used in the ending of the game so as to show the result of the game and update the information of player.

9) The record of player in the array list will be updated immediately after deciding and declaring the results of the game. After that, the procedure 'SaveFile' is run straight after the declaration of game ending in order to update the records of players in the text file immediately.

10) At the end of the game, the statement 'Press Enter to return to Home Page' is shown to instruct user to return to menu. The function 'clrscr' is used to clear the game process in the page so as to provide a clean menu page to the user of the system.

Chapter 4 - Testing and Evaluation

4.1 Brief Description

In this chapter, testing and evaluation of different aspects in the system will be shown.

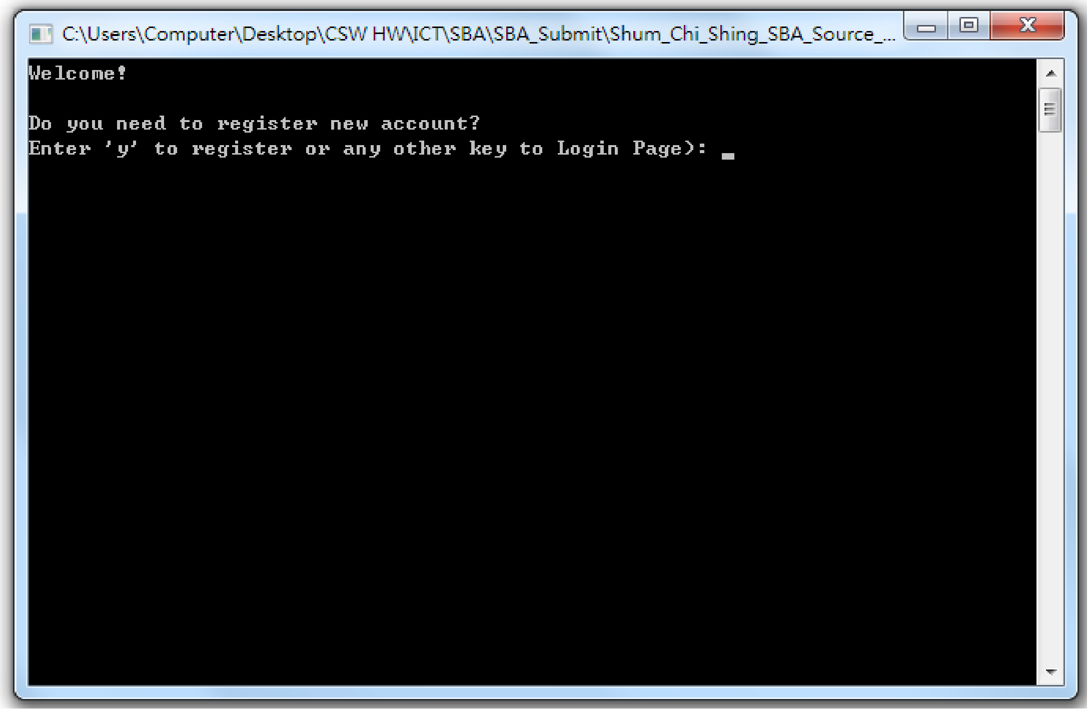
The following will be shown:

1. Test Cases Design and Test Results
2. Further Improvement of the system

4.2 Test Cases Design and Test Results

In this sub-chapter, testing is divided for each subprogram. Test cases will first be designed. The expected outcome is then described. After that, test results will be described, showing if it matches the expected outcome. If not, the problem is described. The details of the testing are as follows.

1) Starting Page of the System

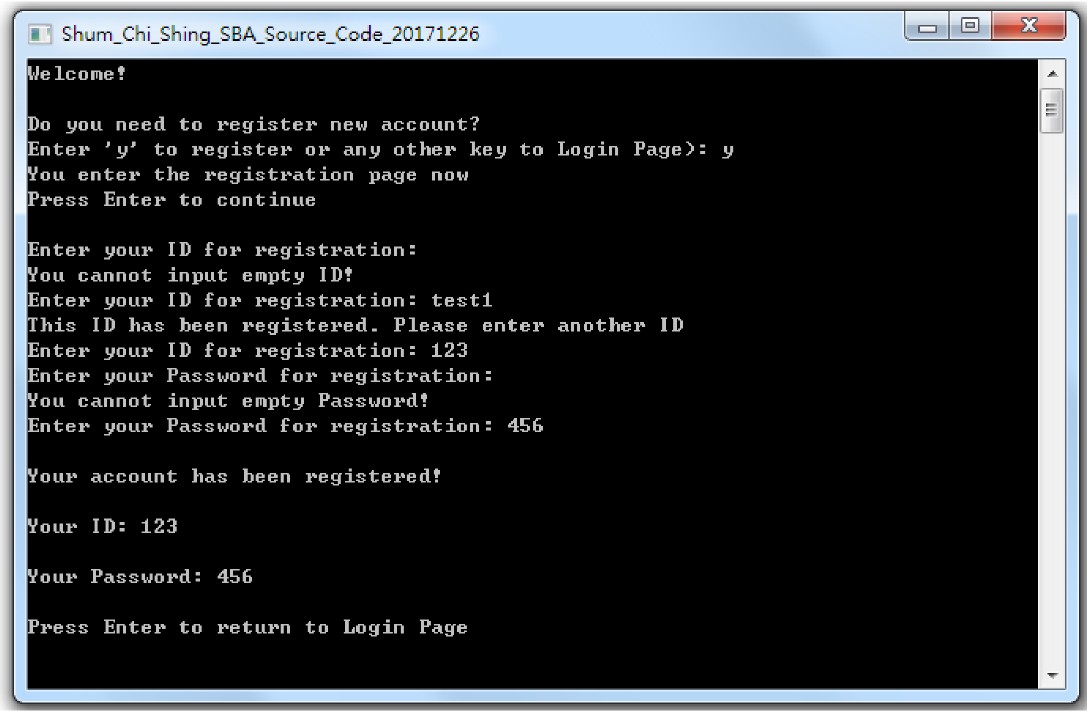

C:\Users\Computer\Desktop\CSW HW\ICT\SBA\SBA_Submit\Shum_Chi_Shing_SBA_Source_...

```
Welcome!

Do you need to register new account?
Enter 'y' to register or any other key to Login Page): _
```

Testing in Starting Page:

Test Case	Test Type	Expected Outcome	Results
'y'	Valid Input	Proceed into Registration Page	As expected
'n'	Valid Input	Proceed to Login Page	As expected
'abc'	Valid Input	Proceed to Login Page	As expected
'1'	Valid Input	Proceed to Login Page	As expected
'#'	Valid Input	Proceed to Login Page	As expected
null	Extreme or Null Input	Proceed to Login Page	As expected

2) Registration Page



```
Welcome!

Do you need to register new account?
Enter 'y' to register or any other key to Login Page): y
You enter the registration page now
Press Enter to continue

Enter your ID for registration:
You cannot input empty ID!
Enter your ID for registration: test1
This ID has been registered. Please enter another ID
Enter your ID for registration: 123
Enter your Password for registration:
You cannot input empty Password!
Enter your Password for registration: 456

Your account has been registered!

Your ID: 123

Your Password: 456

Press Enter to return to Login Page
```

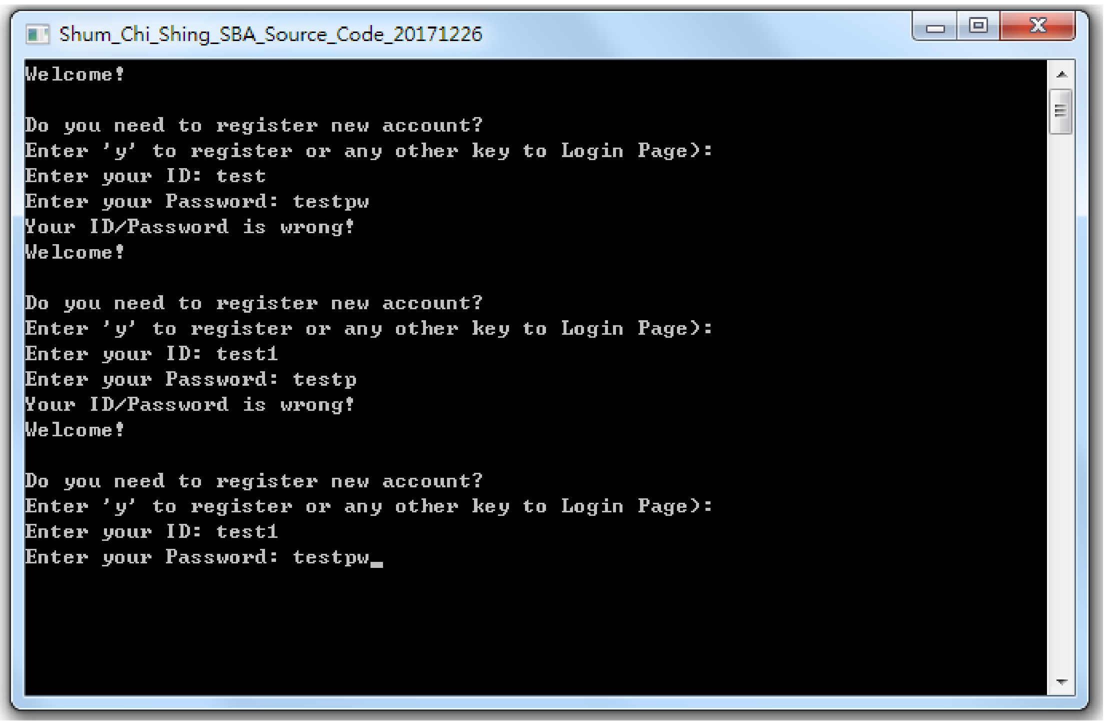
Testing in ID Input:

Test Case	Test Type	Expected Outcome	Results
'123'	Valid Input	Proceed to Password Input	As expected
'abc'	Valid Input	Proceed to Password Input	As expected
'test1'	Invalid Input (Used ID)	Error message exists	As expected
null	Extreme or Null Input	Error message exists	As expected

Testing in Password Input:

Test Case	Test Type	Expected Outcome	Results
'456'	Valid Input	Successfully Registered	As expected
'abc'(Same ID & Password)	Valid Input	Successfully Registered	As expected
'testpw' (Used Password)	Valid Input	Successfully Registered	As expected
null	Extreme or Null Input	Error message exists	As expected

3) Login Page

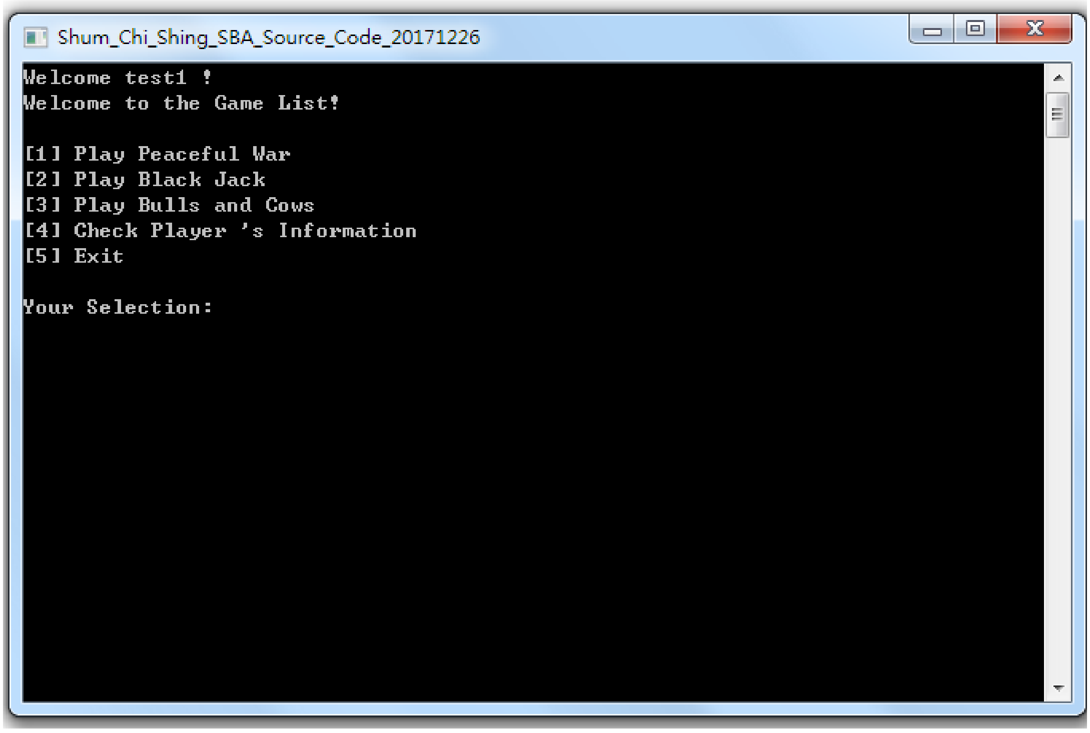


```
Welcome!

Do you need to register new account?
Enter 'y' to register or any other key to Login Page):
Enter your ID: test
Enter your Password: testpw
Your ID/Password is wrong!
Welcome!

Do you need to register new account?
Enter 'y' to register or any other key to Login Page):
Enter your ID: test1
Enter your Password: testp
Your ID/Password is wrong!
Welcome!

Do you need to register new account?
Enter 'y' to register or any other key to Login Page):
Enter your ID: test1
Enter your Password: testpw_
```

Testing in ID & Password (PW) Input:

Test Case	Test Type	Expected Outcome	Results
ID: 'test1' (Correct) PW:'testpw'(Correct)	Valid Input	Successfully login	As expected
ID: 'test' (Wrong) PW:'testpw'(Correct)	Invalid Input	Fail to login	As expected
ID: 'test1' (Correct) PW:'testp'(Wrong)	Invalid Input	Fail to login	As expected
ID: null (Invalid) PW:testpw(Correct)	Extreme or Null Input	Fail to login	As expected
ID: 'test1'(Correct) PW: null (Invalid)	Extreme or Null Input	Fail to login	As expected
ID: null (Invalid) PW: null (Invalid)	Extreme or Null Input	Fail to login	As expected

4) Menu System



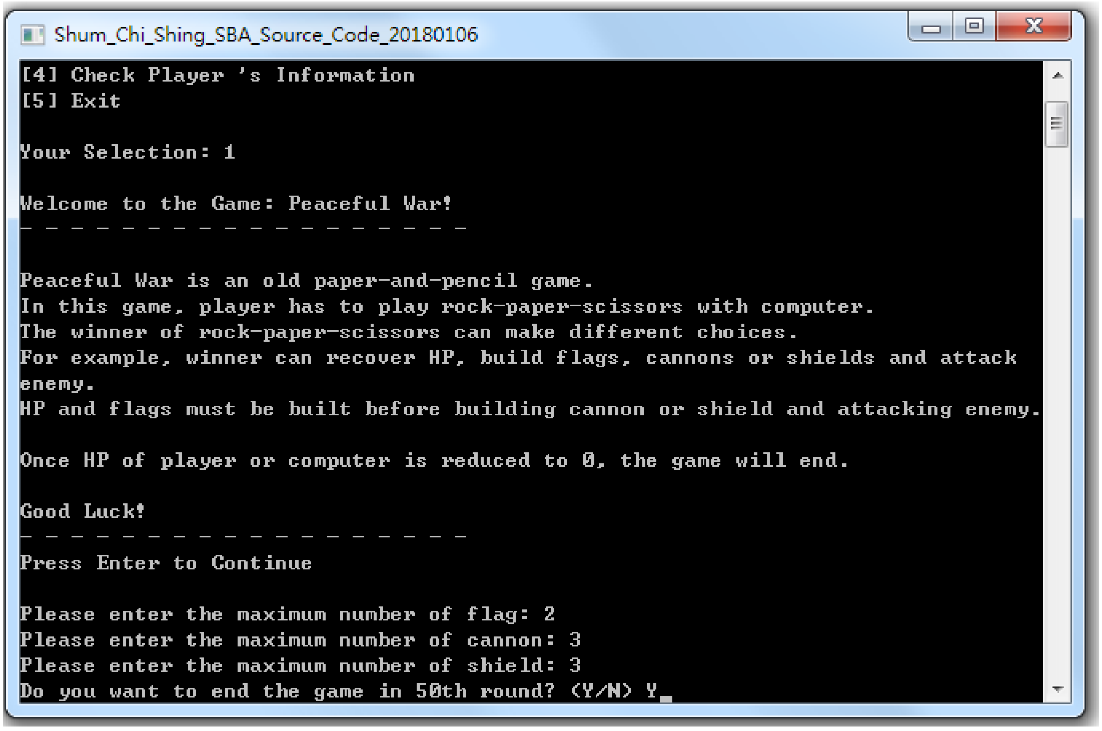
```
Welcome test1 !
Welcome to the Game List!

[1] Play Peaceful War
[2] Play Black Jack
[3] Play Bulls and Cows
[4] Check Player 's Information
[5] Exit

Your Selection:
```

Testing in Choice Input:

Test Case	Test Type	Expected Outcome	Results
1	Valid Input	Proceed to Game	As expected
0	Invalid Input	Menu is repeated to show up	As expected
-1	Invalid Input	Menu is repeated to show up	As expected
'a'	Invalid Input	Error occurs	As expected
'abc'	Invalid Input	Error occurs	As expected
12	Invalid Input	Menu is repeated to show up	As expected
0.5	Invalid Input	Error occurs	As expected
'#'	Invalid Input	Error occurs	As expected
null	Extreme or Null Input	Error occurs	Nothing happens except a new line is started

5) Peaceful War Game

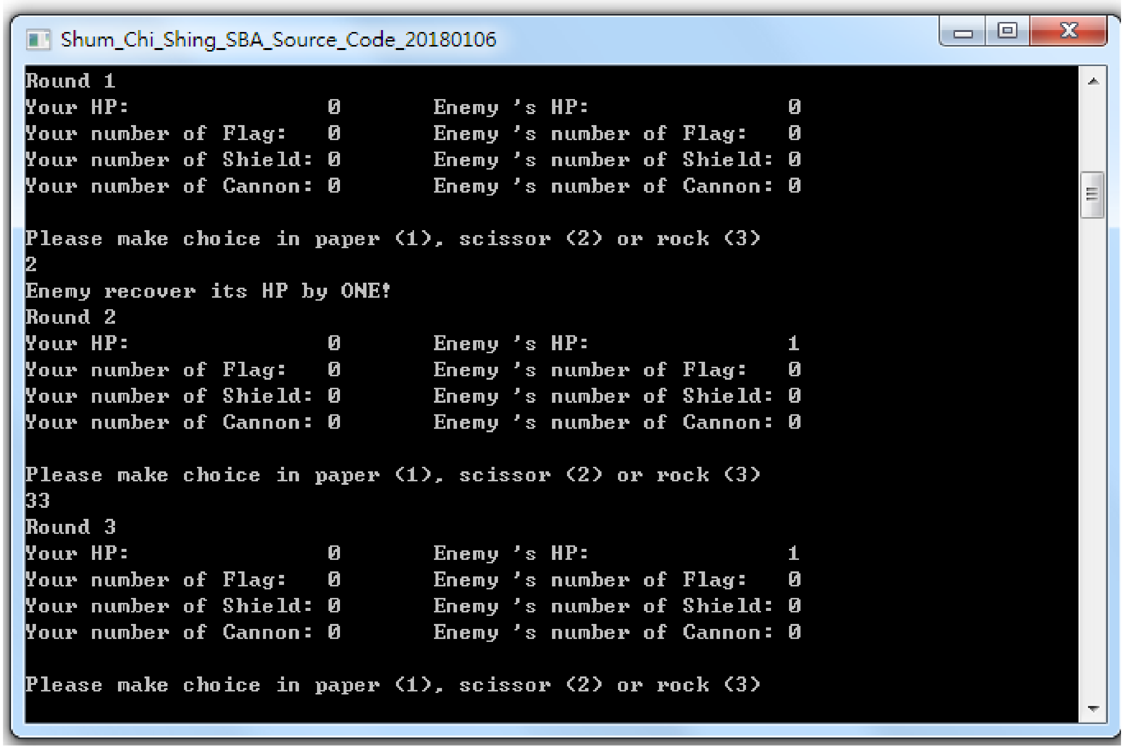
I) Testing in Setting Rules:



```
[4] Check Player 's Information
[5] Exit

Your Selection: 1

Welcome to the Game: Peaceful War!
- - - - - - - - - - - - - - - - - -

Peaceful War is an old paper-and-pencil game.
In this game, player has to play rock-paper-scissors with computer.
The winner of rock-paper-scissors can make different choices.
For example, winner can recover HP, build flags, cannons or shields and attack
enemy.
HP and flags must be built before building cannon or shield and attacking enemy.

Once HP of player or computer is reduced to 0, the game will end.

Good Luck!
- - - - - - - - - - - - - - - - - -
Press Enter to Continue

Please enter the maximum number of flag: 2
Please enter the maximum number of cannon: 3
Please enter the maximum number of shield: 3
Do you want to end the game in 50th round? (Y/N) Y_
```

Test Case	Test Type	Expected Outcome	Results
Flag: 2 Cannon: 3 Shield: 3 End in 50th: Y	Valid Input	Game begins	As expected
Flag: 0 Cannon: 3 Shield: 0 End in 50th: N	Extreme or Null Input	Game begins	As expected
Flag: 0 Cannon: 0 Shield: 0 End in 50th: N	Invalid Input	Repeated input is required	As expected
Flag: 0.1 Cannon: 0.2 Shield: 0.3 End in 50th: 0.4	Invalid Input	Error occurs	As expected
Flag: a Cannon: b Shield: c End in 50th: d	Invalid Input	Error occurs	As expected
Flag: null Cannon: null Shield: null End in 50th: null	Extreme or Null Input	Repeated input is required	Nothing happens except a new line is started

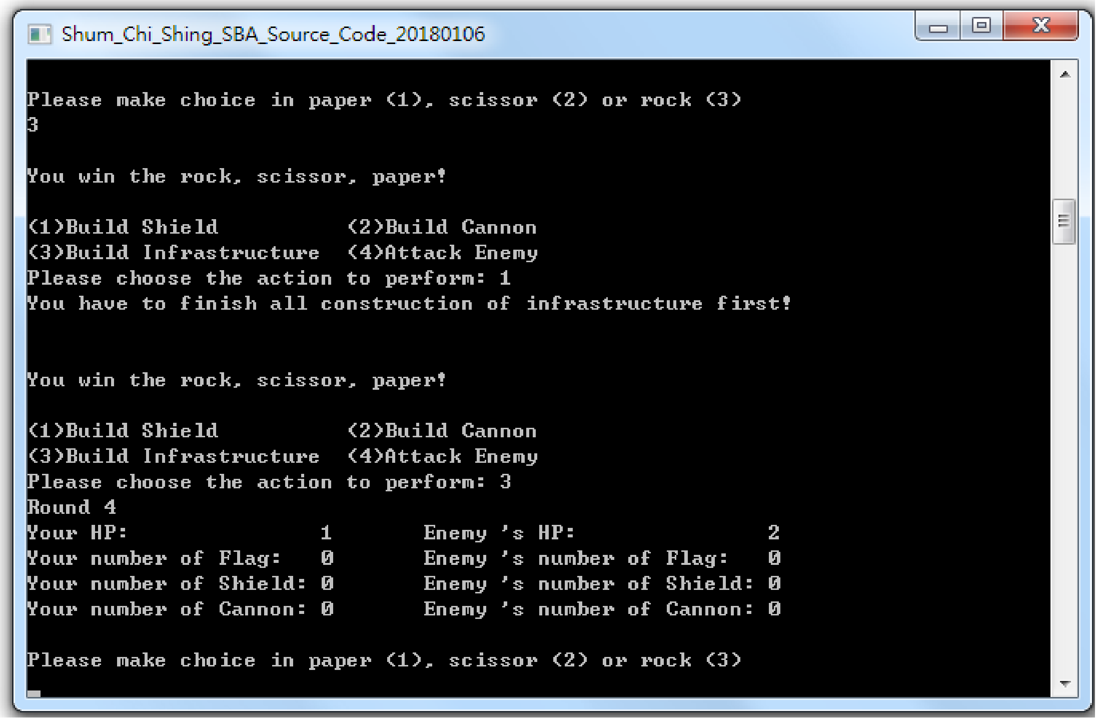
II) Testing in Choices in Rock-Paper-Scissors



```
Round 1
Your HP:              0       Enemy 's HP:              0
Your number of Flag:  0       Enemy 's number of Flag:  0
Your number of Shield: 0      Enemy 's number of Shield: 0
Your number of Cannon: 0      Enemy 's number of Cannon: 0

Please make choice in paper (1), scissor (2) or rock (3)
2
Enemy recover its HP by ONE!
Round 2
Your HP:              0       Enemy 's HP:              1
Your number of Flag:  0       Enemy 's number of Flag:  0
Your number of Shield: 0      Enemy 's number of Shield: 0
Your number of Cannon: 0      Enemy 's number of Cannon: 0

Please make choice in paper (1), scissor (2) or rock (3)
33
Round 3
Your HP:              0       Enemy 's HP:              1
Your number of Flag:  0       Enemy 's number of Flag:  0
Your number of Shield: 0      Enemy 's number of Shield: 0
Your number of Cannon: 0      Enemy 's number of Cannon: 0

Please make choice in paper (1), scissor (2) or rock (3)
```

Test Case	Test Type	Expected Outcome	Results
1	Valid Input	Game runs smoothly	As expected
0	Invalid Input	Repeated input is required	As expected
-1	Invalid Input	Repeated input is required	As expected
4	Invalid Input	Repeated input is required	As expected
0.5	Invalid Input	Error occurs	As expected
'a'	Invalid Input	Error occurs	As expected
'abc'	Invalid Input	Error occurs	As expected
null	Extreme or Null Input	Repeated input is required	Nothing happens except a new line is started

III) Testing in Action Taking

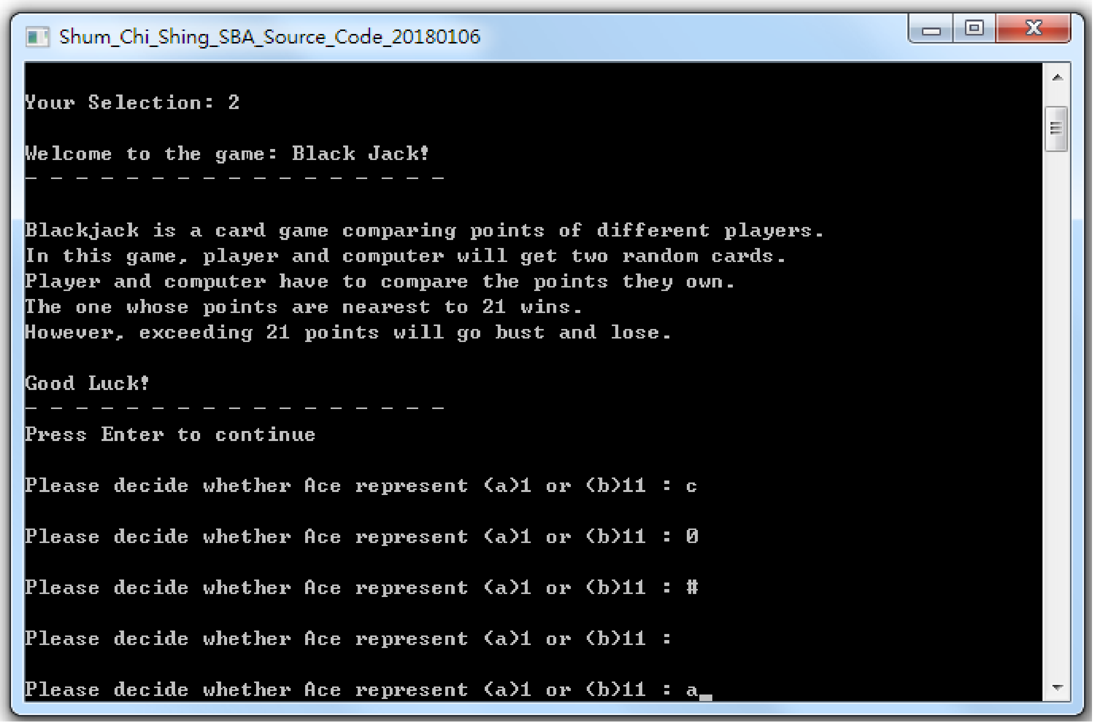


```
Please make choice in paper (1), scissor (2) or rock (3)
3

You win the rock, scissor, paper!

(1)Build Shield          (2)Build Cannon
(3)Build Infrastructure  (4)Attack Enemy
Please choose the action to perform: 1
You have to finish all construction of infrastructure first!


You win the rock, scissor, paper!

(1)Build Shield          (2)Build Cannon
(3)Build Infrastructure  (4)Attack Enemy
Please choose the action to perform: 3
Round 4
Your HP:                1       Enemy 's HP:                2
Your number of Flag:    0       Enemy 's number of Flag:    0
Your number of Shield: 0        Enemy 's number of Shield: 0
Your number of Cannon: 0        Enemy 's number of Cannon: 0

Please make choice in paper (1), scissor (2) or rock (3)
```

Test Case	Test Type	Expected Outcome	Results
1	Valid Input	Error message shows as it does not suit game rules	As expected
3	Valid Input	Game runs smoothly	As expected
4	Invalid Input	Repeated input is required	As expected
-1	Invalid Input	Repeated input is required	As expected
0.5	Invalid Input	Error occurs	As expected
'a'	Invalid Input	Error occurs	As expected
'abc'	Invalid Input	Error occurs	As expected
null	Extreme or Null Input	Repeated input is required	Nothing happens except a new line is started

6) Blackjack Game

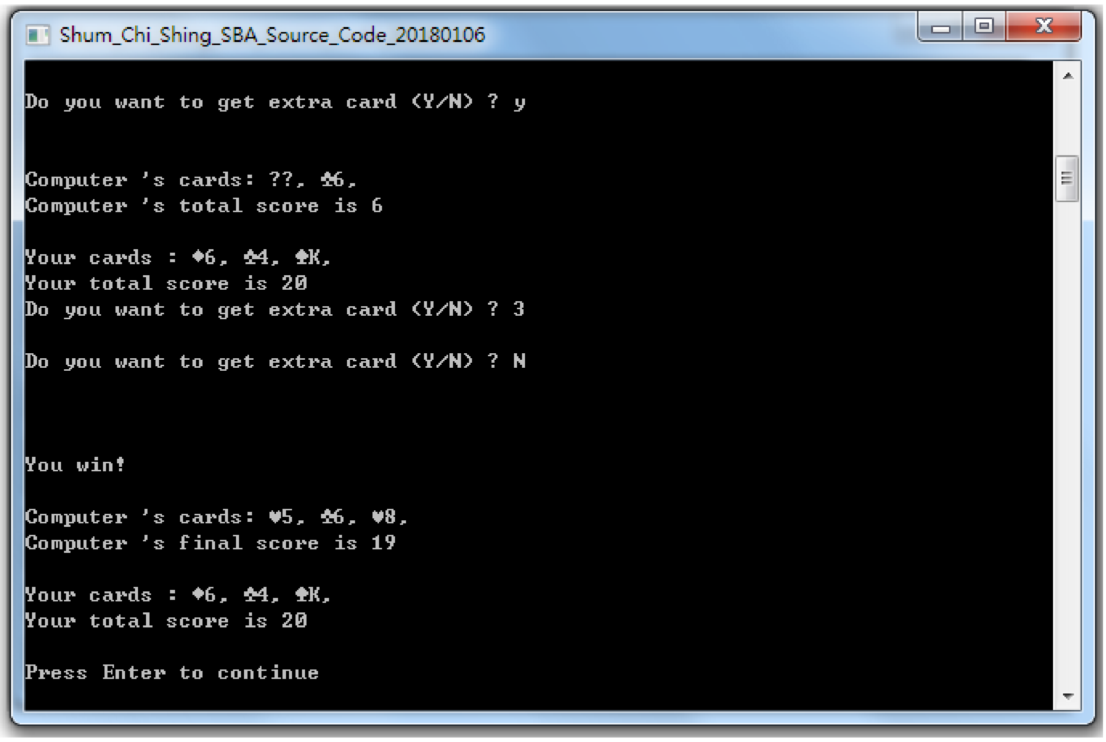
I) Testing in Game Rule Setting



```
Your Selection: 2

Welcome to the game: Black Jack!
- - - - - - - - - - - - - - - - -

Blackjack is a card game comparing points of different players.
In this game, player and computer will get two random cards.
Player and computer have to compare the points they own.
The one whose points are nearest to 21 wins.
However, exceeding 21 points will go bust and lose.

Good Luck!
- - - - - - - - - - - - - - - - -
Press Enter to continue

Please decide whether Ace represent (a)1 or (b)11 : c

Please decide whether Ace represent (a)1 or (b)11 : 0

Please decide whether Ace represent (a)1 or (b)11 : #

Please decide whether Ace represent (a)1 or (b)11 :

Please decide whether Ace represent (a)1 or (b)11 : a_
```

Test Case	Test Type	Expected Outcome	Results
'a'	Valid Input	Game runs smoothly	As expected
'B'	Valid Input	Game runs smoothly	As expected
'ab'	Invalid Input	Repeated input is required	The first character is selected and Game continues
'ca'	Invalid Input	Repeated input is required	As expected
'c'	Invalid Input	Repeated input is required	As expected
0	Invalid Input	Repeated input is required	As expected
1	Invalid Input	Error occurs	Repeated input is required
0.5	Invalid Input	Error occurs	Repeated input is required
'#'	Invalid Input	Repeated input is required	As expected
null	Extreme or Null Input	Repeated input is required	Nothing happens except a new line is started

II) Testing in Obtaining Extra Card

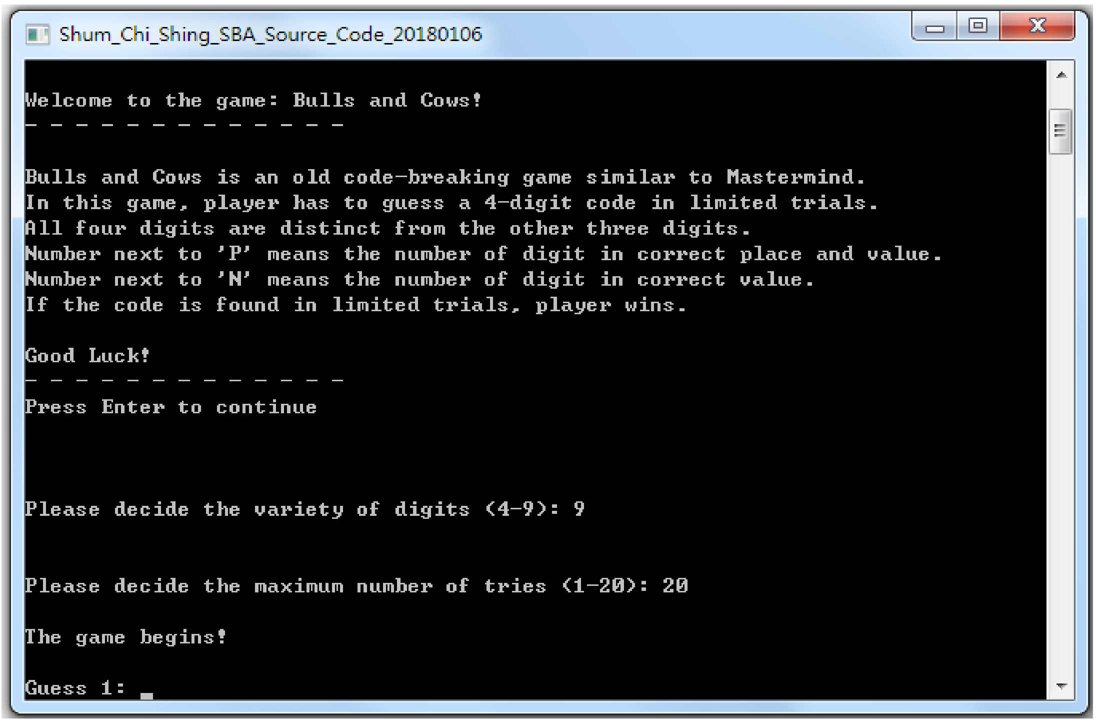


```
Do you want to get extra card (Y/N) ? y


Computer 's cards: ??, ♣6,
Computer 's total score is 6

Your cards : ♦6, ♣4, ♠K,
Your total score is 20
Do you want to get extra card (Y/N) ? 3

Do you want to get extra card (Y/N) ? N



You win!

Computer 's cards: ♥5, ♣6, ♥8,
Computer 's final score is 19

Your cards : ♦6, ♣4, ♠K,
Your total score is 20

Press Enter to continue
```

Test Case	Test Type	Expected Outcome	Results
'y'	Valid Input	Game runs smoothly	As expected
'N'	Valid Input	Game runs smoothly	As expected
'yn'	Invalid Input	Repeated input is required	The first character is selected and Game continues
'zx'	Invalid Input	Repeated input is required	As expected
'z'	Invalid Input	Repeated input is required	As expected
1	Invalid Input	Repeated input is required	As expected
0.5	Invalid Input	Error occurs	Repeated input is required
'#'	Invalid Input	Repeated input is required	As expected
null	Extreme or Null Input	Repeated input is required	Nothing happens except a new line is started

7) Bulls and Cows Game

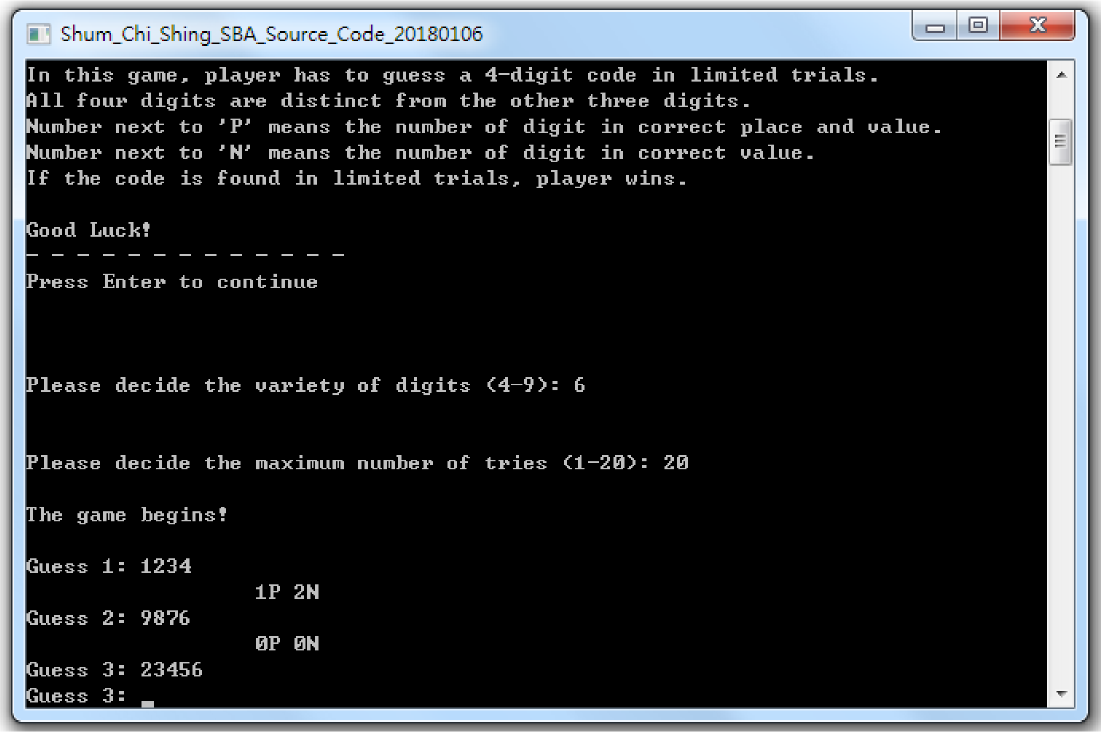
I) Testing in Game Rule Setting



```
Welcome to the game: Bulls and Cows!
- - - - - - - - - - - - - -

Bulls and Cows is an old code-breaking game similar to Mastermind.
In this game, player has to guess a 4-digit code in limited trials.
All four digits are distinct from the other three digits.
Number next to 'P' means the number of digit in correct place and value.
Number next to 'N' means the number of digit in correct value.
If the code is found in limited trials, player wins.

Good Luck!
- - - - - - - - - - - - - -
Press Enter to continue



Please decide the variety of digits (4-9): 9


Please decide the maximum number of tries (1-20): 20

The game begins!

Guess 1: _
```

Test Case	Test Type	Expected Outcome	Results
Variety: 9 Tries: 20	Valid Input	Game runs smoothly	As expected
Variety: 4 Tries: 1	Valid Input	Game runs smoothly	As expected
Variety: 0 Tries: 0	Invalid Input	Repeated input is required	As expected
Variety: 20 Tries: 30	Invalid Input	Repeated input is required	As expected
Variety: -6 Tries: -15	Invalid Input	Repeated input is required	As expected
Variety: 0.1 Tries: 0.9	Invalid Input	Error occurs	As expected
Variety: 'a' Tries: 'b'	Invalid Input	Error occurs	As expected
Variety: '#' Tries: '&'	Invalid Input	Error occurs	As expected
Variety: 'abc' Tries: 'def'	Invalid Input	Error occurs	As expected
Variety: null Tries: null	Extreme or Null Input	Repeated input is required	Nothing happens except a new line is started

II) Testing in Code Guessing



```
In this game, player has to guess a 4-digit code in limited trials.
All four digits are distinct from the other three digits.
Number next to 'P' means the number of digit in correct place and value.
Number next to 'N' means the number of digit in correct value.
If the code is found in limited trials, player wins.

Good Luck!
- - - - - - - - - - - - - -
Press Enter to continue



Please decide the variety of digits (4-9): 6


Please decide the maximum number of tries (1-20): 20

The game begins!

Guess 1: 1234
                  1P 2N
Guess 2: 9876
                  0P 0N
Guess 3: 23456
Guess 3: _
```

Test Case	Test Type	Expected Outcome	Results
1234	Valid Input	Game runs smoothly	As expected
4321	Valid Input	Game runs smoothly	As expected
9876	Invalid Input (Invalid Variety)	Repeated input is required	Game runs smoothly
23456	Invalid Input	Repeated input is required	As expected
11111	Invalid Input	Repeated input is required	As expected
1111	Extreme Input (All same digit)	Game runs smoothly but only '0P0N','1P0N' or '0P1N' shows	'0P0N' or '1P3N' is shown
123	Invalid Input	Repeated input is required	As expected
0.12	Invalid Input	Error occurs	Game runs smoothly but '0P0N' is shown
'abcd'	Invalid Input	Error occurs	Game runs smoothly but '0P0N' is shown
'@#%&'	Invalid Input	Error occurs	Game runs smoothly but '0P0N' is shown
null	Null Input	Repeated input is required	As expected

4.3 Further Improvement of the system

Based on the results of the testing in the sub-chapter 4.2, there are several problems are found. In this sub-chapter, the problem found and the possible solution will be discussed. It is sincerely hoped that these solution can help address the problems in the future. The following are the known problem and related possible solutions.

Problem 1: There are several inputs of user using variable type 'integer'. It causes the existence the run-time error when non-integer input like real number, character or string is entered.

Possible Solution: Changing the variable holding user's input to 'string', using 'repeat' loop and setting a fixed range to limit user from entering valid input.

Problem 2: There are several inputs of user using variable type 'char'. When the string which first letter suits the requirement of that input is entered, the system will adopt the first letter of string to continue running but it is designed to show error message to instruct user to input again.

Possible Solution: Changing the variable holding user's input to 'string'. Then, it should use 'length[input]' to check whether the input contains one character or more than one. If the input contains more than one character, user will be instructed to enter the input again. There will also be rules to check whether user enters the correct type and range of input.

Problem 3: In the game, Bulls and Cows, even if the player set the variety of the game is not including the large digits like 5,6,7,8 or 9, player can still input such numbers while it wastes user a guessing try.

Possible Solution: When user sets the variety of the code, the system may capture the maximum digit can user input to guess. When user input the guess every try, the system can thus check whether there are any digit larger than the maximum digit captured by system.

Problem 4: In the game, Bulls and Cows, if user input the guess that not all four digits are different from others, the same digit will still do the comparison and will be counted in the tips. However, the same digit should not be counted in the tips more than once.

Possible Solution: Flags can be declared so as to indicate whether the same digit has already been counted in the tips or not.

Chapter 5 - Conclusion and Discussion

5.1 Brief Description

This is the last chapter of the report. In this chapter, the strength and weakness of the system will be discussed. After that, the function hoped to be updated in the future will be stated. After all, there is a part of the reflection of the programmer.

5.2 Strength & Weakness of System

Strength:

1) Clear guidelines and instructions are given to users.

2) Not much demanding on hardware as only keyboard and screen are used as input and output device in the system.

Weakness:

1) The user interface is a bit user-unfriendly as the layout is a bit complicated and the interface is monotonous.

2) There are some bugs that may cause run-time error if user enters some invalid inputs.

5.3 Future Improvement on System

1) Bugs can be cleared to prevent all errors from happening.

2) Data encryption can be done in editing text file so as to protect privacy of users.

3) User interface can be improved by changing the layout and colour of texts.

4) The function for users to edit their personal information can be provided.

5) The function for users to view their ranking among different users can be provided.

6) The games can be divided into two parts, playing with computer or playing with friends.

5.4 Reflection

Through this program, I have learnt a lot about system developing. I am glad that I have this opportunity to design such system and I hope that I can have more opportunities to do similar work in the future.

Reference and Acknowledgement

Website:

1) Peaceful War

https://zh.wikipedia.org/wiki/%E5%A4%A9%E4%B8%8B%E5%A4%AA%E5%B9%B3

2) Blackjack

https://en.wikipedia.org/wiki/Blackjack

3) Bulls and Cows

https://en.wikipedia.org/wiki/Bulls_and_Cows

4) Mastermind

https://en.wikipedia.org/wiki/Mastermind_(board_game)

Book:

1) NSS Information and Communication Technology Elective D1 (Pearson)

2) NSS Information and Communication Technology Elective D2 (Pearson)

Appendix

Program Source Code:

```
program SBA;

uses crt;
type
  userinfo = record
                        ID:string;
                        PW:string;
                        winPWar:integer;
                        losePWar:integer;
                        drawPWar:integer;
                        winBJ:integer;
                        loseBJ:integer;
                        drawBJ:integer;
                        winMM:integer;
                        loseMM:integer
                      end;

var
  act, code,pter,
  q,w,e:integer;
  user: array[1..100] of userinfo;
  userdata:text;
  eid,epw,rid,rpw:string;
  sameid,equal:boolean;
  reg:char;

procedure ReadFile;
begin
  code:=0;
  assign(userdata, 'data.txt');
  reset(userdata);

  while not EOF(userdata) do
    begin
      code:=code+1;
      readln(userdata, user[code].ID);
      readln(userdata, user[code].PW);
      readln(userdata, user[code].winPWar, user[code].losePWar,
user[code].drawPWar);
      readln(userdata, user[code].winBJ, user[code].loseBJ, user[code].drawBJ);
      readln(userdata, user[code].winMM, user[code].loseMM);
    end;

  close(userdata)
```

```
end;

procedure SaveFile;
begin
  assign(userdata, 'data.txt');
  rewrite(userdata);

  for e:=1 to code do
    with user[e] do
      begin
        writeln(userdata, ID);
        writeln(userdata, PW);
        writeln(userdata, winPWar, ' ',losePWar, ' ', drawPWar);
        writeln(userdata, winBJ, ' ',loseBJ, ' ', drawBJ);
        writeln(userdata, winMM, ' ',loseMM)
      end;

  close(userdata)
end;

procedure Peaceful_War;

var
  maxfg,mfg,efg,
  maxcn,mcn,ecn,
  maxsd,msd,esd,
  mHP,eHP,
  mhit,ehit,
  mhd,ehd,
  mchoice,echoice,
  step,target:integer;
  mhdwin,ehdwin,hddraw,
  mvalidmv,evalidmv:boolean;
  limround:char;

begin
  writeln;
  randomize;
  writeln('Welcome to the Game: Peaceful War!');
  writeln('- - - - - - - - - - - - - - - - - -');
  writeln;
  writeln('Peaceful War is an old paper-and-pencil game. ');
  writeln('In this game, player has to play rock-paper-scissors with computer. ');
  writeln('The winner of rock-paper-scissors can make different choices.');
  writeln('For example, winner can recover HP, build flags, cannons or shields and
attack');
  writeln('enemy.');
  writeln('HP and flags must be built before building cannon or shield and attacking
enemy.');
  writeln('Once HP of player or computer is reduced to 0, the game will end. ');
  writeln;
  writeln('Good Luck!');
  writeln('- - - - - - - - - - - - - - - - - -');
  writeln('Press Enter to Continue');
  readln;

  maxfg:=0; mfg:=0; efg:=0;
  maxcn:=0; mcn:=0; ecn:=0;
  maxsd:=0; msd:=0; esd:=0;
  mHP:=0; eHP:=0;
  mhit:=0; ehit:=0;
  mhd:=0; ehd:=0;
  mchoice:=0; echoice:=0;
  step:=0;
  target:=0;
  limround:='n';
  mhdwin:=FALSE; ehdwin:=FALSE; hddraw:=FALSE;
  mvalidmv:=FALSE; evalidmv:=FALSE;

  repeat
    write('Please enter the maximum number of flag: ');
    readln(maxfg);
  until maxfg>=0;
  repeat
  write('Please enter the maximum number of cannon: ');
  readln(maxcn);
  until maxcn>0;
  repeat
  write('Please enter the maximum number of shield: ');
  readln(maxsd);
  until maxsd>=0;
  repeat
  write('Do you want to end the game in 50th round? (Y/N) ');
  readln(limround);
  until (limround='Y') or (limround='y') or (limround='N') or (limround='n');
  writeln;

  repeat
    step:=step+1;
    writeln('Round ',step);
    writeln('Your HP:                          ',mHP,'            Enemy ''s HP:
',eHP);
    writeln('Your number of Flag:      ',mfg,'            Enemy ''s number of Flag:
',efg);
    writeln('Your number of Shield: ',msd,'            Enemy ''s number of Shield: ',esd);
    writeln('Your number of Cannon: ',mcn,'            Enemy ''s number of Cannon:
',ecn);
    mhdwin:=FALSE; ehdwin:=FALSE; hddraw:=FALSE;
    mvalidmv:=FALSE; evalidmv:=FALSE;
    mchoice:=0; echoice:=0;
```

```
mhd:=0; ehd:=0;
writeln;
repeat
  writeln('Please make choice in paper (1), scissor (2) or rock (3)');
  readln(mhd);
until (mhd>=1) and (mhd<=3);

ehd:=random(3)+1;

if ((mhd=1) and (ehd=1)) or
      ((mhd=2) and (ehd=2)) or
      ((mhd=3) and (ehd=3))
then hddraw:=TRUE;

if ((mhd=1) and (ehd=3)) or
      ((mhd=2) and (ehd=1)) or
      ((mhd=3) and (ehd=2))
then mhdwin:=TRUE;

if ((mhd=1) and (ehd=2)) or
      ((mhd=2) and (ehd=3)) or
      ((mhd=3) and (ehd=1))
then ehdwin:=TRUE;

if mhdwin
then begin
            repeat
              writeln;
              writeln('You win the rock, scissor, paper!');
              writeln;
              writeln('(1)Build Shield                    (2)Build Cannon');
              writeln('(3)Build Infrastructure    (4)Attack Enemy');
              write('Please choose the action to perform: ');
              readln(mchoice);
              if (not(mchoice=3)) and (not(mHP=4))
              then begin
                          writeln('You have to finish all construction of infrastructure first!');
                          writeln
                        end
              else if (not(mchoice=3)) and (not(mfg=maxfg))
                        then begin
                          writeln('You have to finish all construction of infrastructure first!');
                          writeln
                        end
                        else if (mchoice=1) and (not(msd<maxsd))
                                  then begin
                                    writeln('You cannot build shield anymore!');
                                    writeln
                                  end
                                  else if (mchoice=2) and (not(mcn<maxcn))
                        then begin
                         writeln('You cannot build cannon anymore!');
                         writeln
                        end
                        else if (mchoice=3) and (mHP=4) and (mfg=maxfg)
                             then begin
                              writeln('You cannot recover your HP and build
flag anymore!');
                              writeln
                             end
                             else if (mchoice=4) and (mcn=0)
                                  then begin
                                   writeln('You have no cannon to attack the
enemy!');
                                   writeln
                                  end
                                  else if (mchoice<1) or (mchoice>4)
                                       then begin
                                        writeln('Invalid Input!');
                                        writeln
                                       end
                                       else mvalidmv:=TRUE
        until mvalidmv;

        if mvalidmv
        then case mchoice of
              1 : msd:=msd+1;
              2 : mcn:=mcn+1;
              3 : if mHP<>4
                  then mHP:=mHP+1
                  else mfg:=mfg+1;
              4 : begin
                   target:=random(100)+1;
                   case target of
                    1..45 : if ecn>0
                            then begin
                                  ecn:=ecn-1;
                                  mhit:=mhit+1;
                                  writeln('You destroy one of the Enemy ''s
cannons')
                                 end
                            else target:=random(55)+46;
                    46..90 : if esd>0
                             then begin
                                   esd:=esd-1;
                                   mhit:=mhit+1;
                                   writeln('You destroy one of the Enemy ''s
shields')
                                  end
                             else if efg>0
```

```
                                  then begin
                                        efg:=efg-1;
                                        mhit:=mhit+1;
                                        writeln('You destroy one of the
Enemy "s flags')
                                       end
                                  else begin
                                        ehp:=ehp-1;
                                        mhit:=mhit+1;
                                        writeln('You reduce Enemy "s HP
by 1')
                                       end;
                    91..100 : writeln('Your shot missed!')
                   end
                  end
             end
       end;

  if ehdwin
  then begin
        repeat
         echoice:=random(4)+1;
         if (not(echoice=3)) and (not(eHP=4))
         then evalidmv:=FALSE
         else if (not(echoice=3)) and (not(efg=maxfg))
              then evalidmv:=FALSE
              else if (echoice=1) and (not(esd<maxsd))
                   then evalidmv:=FALSE
                   else if (echoice=2) and (not(ecn<maxcn))
                        then evalidmv:=FALSE
                        else if (echoice=3) and (eHP=4) and (efg=maxfg)
                             then evalidmv:=FALSE
                             else if (echoice=4) and (ecn=0)
                                  then evalidmv:=FALSE
                                  else evalidmv:=TRUE
        until evalidmv;

        case echoice of
         1 : begin
              esd:=esd+1;
              writeln('Enemy built a shield!')
             end;
         2 : begin
              ecn:=ecn+1;
              writeln('Enemy built a cannon!')
             end;
         3 : if eHP<>4
             then begin
                   eHP:=eHP+1;
                   writeln('Enemy recover its HP by ONE!')
                  end
             else begin
                    efg:=efg+1;
                    writeln('Enemy built a flag!')
                  end;
         4 : begin
              target:=random(100)+1;
              case target of
               1..45 : if mcn>0
                        then begin
                              mcn:=mcn-1;
                              ehit:=ehit+1;
                              writeln('Enemy destroyed one of your cannons!')
                             end
                        else target:=random(55)+46;
               46..90 : if msd>0
                         then begin
                               msd:=msd-1;
                               ehit:=ehit+1;
                               writeln('Enemy destroyed one of your shields!')
                              end
                         else if mfg>0
                              then begin
                                    mfg:=mfg-1;
                                    ehit:=ehit+1;
                                    writeln('Enemy destroyed one of your
flags!')
                                   end
                              else begin
                                    mhp:=mhp-1;
                                    ehit:=ehit+1;
                                    writeln('Enemy reduced your HP by
ONE!')
                                   end;
               91..100 : writeln('Enemy ''s shot missed!')
              end
             end
        end
       end;

  if hddraw
  then begin
        writeln('It is a draw...');
        writeln
       end

 until (((limround='y') or (limround='Y')) and (step=50))
       or ((mhit>0) and (eHP=-1))
       or ((ehit>0) and (mHP=-1));
  if (((limround='y') or (limround='Y')) and (step=50))
  then begin
            writeln('The game is ended and nobody win!');
            user[pter].drawPWar:=user[pter].drawPWar+1
          end;

  if ((mhit>0) and (eHP=-1))
  then begin
            writeln('You win the game!');
            user[pter].winPWar:=user[pter].winPWar+1
          end;

  if ((ehit>0) and (mHP=-1))
  then begin
            writeln('You lose the game...');
            user[pter].losePWar:=user[pter].losePWar+1
          end;

  writeln('- - - - - - - - - - - - - - - - - - - - - - -');
  SaveFile;
  write('Press Enter to return to Home Page');
  readln;
  clrscr
end;

procedure Black_Jack;
const maxpt=21;
var
  ab,maddkey:char;
  temptxt:string;
  ace11,draw,
  mwin,ewin,
  mex,eex:boolean;
  usedcd:array [1..52] of boolean;
  cdnum:array [1..52] of integer;
  cdtxt:array [1..52] of string[3];
  mcd,ecd:array [1..5] of string[3];
  mvalue,evalue:array [1..5] of integer;
  x,y,z,
  msum,esum,eshowsum,
  mcount,ecount:integer;

begin
  writeln;
  randomize;
  writeln('Welcome to the game: Black Jack!');
  writeln('- - - - - - - - - - - - - - - - -');
  writeln;
```

```
writeln('Blackjack is a card game comparing points of different players.');
writeln('In this game, player and computer will get two random cards.');
writeln('Player and computer have to compare the points they own.');
writeln('The one whose points are nearest to 21 wins.');
writeln('However, exceeding 21 points will go bust and lose.');
writeln;
writeln('Good Luck!');
writeln('- - - - - - - - - - - - - - - - -');
writeln('Press Enter to continue');
readln;

ab:=' ';
ace11:=FALSE;
mwin:=FALSE;
ewin:=FALSE;
mex:=FALSE;
eex:=FALSE;
draw:=FALSE;
x:=0;
y:=0;

while (ab<>'a') and (ab<>'b') and (ab<>'A') and (ab<>'B')do
begin
  write('Please decide whether Ace represent (a)1 or (b)11 : ');
  readln(ab);
  writeln
end;

if (ab='a') or (ab='A')
then ace11:=FALSE;
if (ab='b') or (ab='B')
then ace11:=TRUE;
writeln;
writeln('The game begins!');

for x:=1 to 52 do
begin
  usedcd[x]:=FALSE;
  y:=y+1;
  cdnum[x]:=y;
  if (cdnum[x]=11) or (cdnum[x]=12) or (cdnum[x]=13)
  then cdnum[x]:=10;
  if (cdnum[x]=1) and Ace11
  then cdnum[x]:=11;
  if y=13
  then y:=0
end;

x:=0; y:=0; z:=1;
```

```
for x:=3 to 6 do
  for y:=1 to 13 do
    begin
      str(y,temptxt);
      if temptxt='1'
      then temptxt:='A';
      if temptxt='11'
      then temptxt:='J';
      if temptxt='12'
      then temptxt:='Q';
      if temptxt='13'
      then temptxt:='K';
      cdtxt[z]:=chr(x)+temptxt;
      z:=z+1
    end;

for x:=1 to 5 do
  repeat
    y:=random(52)+1;
    z:=random(52)+1;
    if (not(usedcd[y])) and (not(usedcd[z])) and (y<>z)
    then begin
                mcd[x]:=cdtxt[y];
                mvalue[x]:=cdnum[y];
                usedcd[y]:=TRUE;
                ecd[x]:=cdtxt[z];
                evalue[x]:=cdnum[z];
                usedcd[z]:=TRUE
              end
  until (mcd[x]<>") and (ecd[x]<>");

msum:=mvalue[1]+mvalue[2];
esum:=evalue[1]+evalue[2];
eshowsum:=evalue[2];
writeln;
writeln('Computer "s cards: ??, ',ecd[2]);
writeln('Computer "s total score is ', eshowsum);
writeln;
writeln('Your cards : ',mcd[1],', ',mcd[2]);
writeln('Your total score is ', msum);
writeln;
mcount:=3;
ecount:=2;

repeat
  maddkey:='N';
  if (mcount<=5) and (msum<maxpt)
  then    repeat
```

```
                write('Do you want to get extra card (Y/N) ? ');
                readln(maddkey);
                writeln
              until (maddkey='Y') or (maddkey='y') or (maddkey='N') or (maddkey='n');

  if ((maddkey='Y') or (maddkey='y')) and (mcount<=5) and (msum<=maxpt)
  then begin
              writeln;
              msum:=msum+mvalue[mcount];
              write('Computer ''s cards: ??, ');
              for x:=2 to ecount do
                write(ecd[x],', ');
              writeln;
              writeln('Computer ''s total score is ', eshowsum);
              writeln;
              write('Your cards : ');
              for y:=1 to mcount do
                write(mcd[y],', ');
              writeln;
              writeln('Your total score is ', msum);
              mcount:=mcount+1
            end;
  if msum>21
  then mex:=TRUE
until (maddkey='N') or (maddkey='n') or (mex);

while (esum<=17) and (ecount<5) do
begin
  writeln;
  ecount:=ecount+1;
  eshowsum:=eshowsum+evalue[ecount];
  esum:=esum+evalue[ecount]
end;

writeln;
if msum=21
then mwin:=TRUE;
if esum=21
then ewin:=TRUE;
if msum>21
then mex:=TRUE;
if esum>21
then eex:=TRUE;
if (not(mex)) and (not(eex)) and (msum>esum)
then mwin:=TRUE;
if (not(mex)) and (not(eex)) and (esum>msum)
then ewin:=TRUE;
if (mex) and (eex)
then draw:=TRUE;
if (msum=esum)
```

```
then draw:=TRUE;

if (mwin) and (not(draw))
then begin
            writeln('You win!');
            user[pter].winBJ:=user[pter].winBJ+1
          end;

if (ewin) and (not(draw))
then begin
            writeln('You lose!');
            user[pter].loseBJ:=user[pter].loseBJ+1
          end;

if (eex) and (not(draw))
then begin
            writeln('Computer goes bust! You win!');
            user[pter].winBJ:=user[pter].winBJ+1
          end;

if (mex) and (not(draw))
then begin
            writeln('You go bust! You lose!');
            user[pter].loseBJ:=user[pter].loseBJ+1
          end;

if draw
then begin
            writeln('It is a draw!');
            user[pter].drawBJ:=user[pter].drawBJ+1
          end;

writeln;

write('Computer ''s cards: ');
for x:=1 to ecount do
  write(ecd[x],', ');
writeln;
writeln('Computer ''s final score is ', esum);

writeln;

write('Your cards : ');
for y:=1 to mcount-1 do
  write(mcd[y],', ');
writeln;
writeln('Your total score is ', msum);

writeln;
```

```
  SaveFile;
  writeln('Press Enter to continue');
  readln;
  clrscr
end;

procedure Bulls_and_Cows;
var
  vardig,tries,tempcode,
  count,x,y,z,i,j,k,
  P,N:integer;
  useddig:array [1..9] of boolean;
  win,valid:boolean;
  guess:array [1..20] of string;
  ans,tempdig:string;

begin
  writeln;
  randomize;
  writeln('Welcome to the game: Bulls and Cows!');
  writeln('- - - - - - - - - - - - -');
  writeln;
  writeln('Bulls and Cows is an old code-breaking game similar to Mastermind.');
  writeln('In this game, player has to guess a 4-digit code in limited trials.');
  writeln('All four digits are distinct from the other three digits.');
  writeln('Number next to "P" means the number of digit in correct place and value.');
  writeln('Number next to "N" means the number of digit in correct value.');
  writeln('If the code is found in limited trials, player wins.');
  writeln;
  writeln('Good Luck!');
  writeln('- - - - - - - - - - - - -');
  writeln('Press Enter to continue');
  readln;
  writeln;

  repeat
    writeln;
    write('Please decide the variety of digits (4-9): ');
    readln(vardig)
  until (vardig>=4) and (vardig<=9);
    writeln;

  repeat
    writeln;
    write('Please decide the maximum number of tries (1-20): ');
    readln(tries);
  until (tries>=1) and (tries<=20);
  writeln;
```

```
 writeln('The game begins!');
 writeln;

 win:=FALSE;
 ans:='';
 tempdig:='';
 tempcode:=0;
 valid:=FALSE;
 count:=0;
 for x:=1 to 9 do
  useddig[x]:=FALSE;
 for y:=1 to 20 do
   guess[y]:='';

 for z:=1 to 4 do
  begin
   repeat
    tempcode:=random(vardig)+1
   until not(useddig[tempcode]);
   useddig[tempcode]:=TRUE;
   str(tempcode,tempdig);
   ans:=ans+tempdig
  end;

 while (not(win)) and (not(count>tries-1)) do
  begin
   count:=count+1;

   repeat
    write('Guess ',count,': ');
    readln(guess[count]);
    valid:=TRUE;
    if (length(guess[count])<>4)
    then valid:=FALSE
    else for i:=1 to 4 do
          if (copy(guess[count],i,1)<chr(49)) and
(copy(guess[count],i,1)>chr(vardig+48))
          then valid:=FALSE
   until valid;

  write('                              ');
  P:=0; N:=0;
  for j:=1 to 4 do
   for k:=1 to 4 do
    if copy(guess[count],j,1)=ans[k]
    then if j=k
         then P:=P+1
         else N:=N+1;
    if P=4
    then win:=TRUE
    else write(P,'P ',N,'N');
    writeln
  end;

  writeln;

  if win
  then begin
            writeln('You win by guessing ',count,' times!');
            user[pter].winMM:=user[pter].winMM+1
          end
  else begin
            writeln('You lose and the answer is ',ans,' !');
            user[pter].winMM:=user[pter].winMM+1
          end;

  writeln;
  SaveFile;
  writeln('Press Enter to continue');
  readln;
  clrscr
end;

procedure Info;
begin
  writeln('Your Information:');
  writeln;
  writeln('Name of Player: ',user[pter].ID);
  writeln;
  writeln('                              Win              Lose              Draw');
  writeln('Peaceful War:        ', user[pter].winPWar, '                 ',
user[pter].losePWar, '                 ', user[pter].drawPWar);
  writeln('Black Jack:          ', user[pter].winBJ, '                 ', user[pter].loseBJ, '
', user[pter].drawBJ);
  writeln('Bulls and Cows:      ', user[pter].winMM, '                 ', user[pter].loseMM, '
NIL');
  writeln;
  writeln('Press Enter to return to Main Page');
  readln;
  clrscr
end;

begin
  ReadFile;
```

```
repeat
writeln('Welcome!');
writeln;
writeln('Do you need to register new account?');
write('Enter "y" to register or any other key to Login Page): ');
readln(reg);

if reg='y'
then begin
      writeln('You enter the registration page now');
      writeln('Press Enter to continue');
      readln;
      repeat
        sameid:=FALSE;
        repeat
          write('Enter your ID for registration: ');
          readln(rid);
          if length(rid)=0
          then writeln('You cannot input empty ID!');
        until length(rid)<>0;
        for q:=1 to code do
          if rid=user[q].ID
          then begin
                  sameid:=TRUE;
                  writeln('This ID has been registered. Please enter another ID')
                end
      until sameid=FALSE;
      repeat
        write('Enter your Password for registration: ');
        readln(rpw);
        if length(rpw)=0
        then writeln('You cannot input empty Password!');
      until length(rpw)<>0;
      writeln;
      code:=code+1;
      with user[code] do
      begin
        ID:=rid;
        PW:=rpw;
        winPWar:=0;
        losePWar:=0;
        drawPWar:=0;
        winBJ:=0;
        loseBJ:=0;
        drawBJ:=0;
        winMM:=0;
        loseMM:=0
      end;
```

```
              writeln('Your account has been registered!');
              writeln;
              writeln('Your ID: ',user[code].ID);
              writeln;
              writeln('Your Password: ',user[code].PW);
              writeln;
              SaveFile;
              writeln('Press Enter to return to Login Page');
              readln;
              clrscr
             end;

   repeat
     write('Enter your ID: ');
     readln(eid);
   until length(eid)<>0;

   repeat
   write('Enter your Password: ');
   readln(epw);
   until length(epw)<>0;

   equal:=FALSE;

   for w:=1 to code do
     if (eid=user[w].ID) and (epw=user[w].PW)
     then begin
               equal:=TRUE;
               pter:=w
              end;
   if equal=FALSE
   then writeln('Your ID/Password is wrong!');
 until equal=TRUE;
 clrscr;
 writeln('Welcome ',user[pter].ID,' !');

 repeat
   writeln('Welcome to the Game List!');
   writeln;
   writeln('[1] Play Peaceful War');
   writeln('[2] Play Black Jack');
   writeln('[3] Play Bulls and Cows');
   writeln('[4] Check Player ''s Information');
   writeln('[5] Exit');
   writeln;
   write('Your Selection: ');
   readln(act);
   case act of
     1 : Peaceful_War;
     2 : Black_Jack;
```

```
      3 : Bulls_and_Cows;
      4 : Info
    end
  until act=5;
  clrscr;
  SaveFile;
  writeln('Press Enter to leave');
  readln
end.
```