ENG2002 Computer Programming

Mini Project

LAST CARD

Group 31

CHAN Yi Ming 13071741D

CHAU Siu Fai 13071191D

Abstract:

This project is making a game call Last Card. The game is like a UNO. The winner is the player that is the first one to discard all cards. Besides, we have added a login system to store the mark of the player. To make the game more users friendly, we added extra features. In this game, we need to handle the code of unmanaged and managed type. For unmanaged type, we need to build a static library to implement to the GUI windows form. We create two library, login.lib and lastcard.lib. After doing this project, we have a good experience on doing object-oriented programming. Also, while making the game, we face a lot of bugs, it is a good time for us to practice the logical thinking. To conclude, we build a systematic mind to handle different problems on the future.

Introduction

This mini-project aims to let students learn C++ programming from practical experience. In our project, we have asked to write a C++ object-oriented program to implement computer card game “Last Card”. “Last Card” is a popular card game played in New Zealand in schools and gaming venues. It is similar in most aspects to “Uno”, “Mau Mau” or “Crazy Eights” but several rules differentiate it, for instance the function of a particular card.

Rules

When the game starts, five cards will be distributed to each player. The following card will be placed in discard-pile and keep face up. The cards left will be used to form a stockpile. Player then requires discarding a card which rank or suit match with the exposed card in the discard-pile. The discarded card then becomes the exposed card. Player needs to draw two cards from the stockpile if he failed to do so. Player must declare “Last Card” before discarding his last card. If he didn’t, he needs to draw five cards from stockpile as penalty. Whenever the stockpile is empty, the game shuffles the discard-pile and makes a new stockpile. The first player who discards all cards in hand is the winner. Or when the time is up, the player who holds the less card will become winner.

Some special-action must act with the following cards:

- ACE: The direction of the play turns reverse. In a 2-player game, this card has no effect.
- TWO: The next player either draws two cards from the stockpile of discards another two-card(if the play can) to add up for his next player to draw, i.e. his next player needs to draw 4 cards when being unable to discard another two-card, and so on until cards are drawn from the stockpile.
- EIGHT: The current player can specify a suit for the next player to follow.
- JACK: The next player skips a turn. In a 2-player game, this acts as a "play-again" card.
- JOKER: The cards act as wild cards, which include special action cards.

Special-action card must not be the last card to discard.

Before the player start a game, he/she needs to create account or login to the account existed. Player can look at the rules by clicking rules button. When the player have not logged in and click the start game button, an error box will pop up. But the player still can click ranking to check the rank. After logged in, it will show the login name, nick name and mark of the player. Now, the player can start game, a form will pop up for the player to choose the number of computer (min=0, max=3), the number of player (min=1, max=4), the total number of player and computer cannot larger than 4, and the gaming time (5minutes or 10minutes), then the player can click the start. An initial draw card animation will appear. Then, the timer starts to count down on the top left corner. When it is the player turn, he/she can click on the card to discard, or clicking the help button, it will have a red edge when the card can discard. If the player wants to pause, he can press the pause button to stop the timer. Also, when he confuse with the rules, he can click the rule button, it will also stop the timer. There is a yellow label to show which player's turn. When the player discard "8", a form will pop up, player can change the suit by clicking the picture box.

When the time is up or any one's hand card number is 0, a form will pop up and show the winner, it will also update the mark.

Distribution of work

CHAN Yi Ming	Chau Siu Fai
Design GUI Layout	Design GUI Layout
Implement game operation	Implement game operation
Documentation	Documentation
Implement game initialization	Implement game initialization
Design system flow	Design system flow
Implement login system	Implement login system
Implement additional features	Implement additional features
Implement GUI	Implement GUI

Work schedule of implementing the project

2/3/2015-4/3/2015	Design System Flow
4/3/2015-10/3/2015	Design Header and class
10/3/2015-16/3/2015	Implement class member function
16/3/2015-20/3/2015	Design GUI layout
20/3/2015	Interim report documentation
21/3/2015-27/3/2015	Implement GUI layout
27/3/2015-31/3/2015	Debug program
1/4/2015-4/4/2015	Final report documentation

Classes

i) login Class

It is the class for the login system which stores the user information, including username, password and marks. It also include some function to read and write the text file, sorting, verify user information and update all information.

```
class login{
public:
	login();
	~login();
	void update();
	bool newac(char newun[20], char newpw[20], char newnick[20]);
	void signin(char readun[20], char readpw[20]);
	int getmark(int id);
	char *getnick(int id);
	char *getun(int id);//get username
	int logged();//check logged or get the id of the player
	void best10(int* rank);
	int getnum();//get number of total registed player
```

```
        void updatemark();//the player's mark+1,the marktxt of the player +1
private:
        int log;//-1 when not logged in,otherwise store the id of player
        char username[50][20];//maximun 50 ac
        char pw[50][20];//50 player with the pw length of 20
        char marktxt[50][10];//50 player with the maek length of 10
        int mark[50];
        char nick[50][20];//50 player with the nick name length of 20
        char num[4];//get the number of player in txt file
        int id;//store the  number of player in integer
};
```

ii) player class

It is the class to stores the cards player has. It also includes functions to modify player cards and return the value of cards.

```
class Player  //for store one player's card
{
public:
   void outCard(int cardNumber); //give out the card with the corresponding card number
```

```
    void inCard(int cardNumber);//draw the card with the corresponding card number to the hand card
    void initialCard(int initialCard[]);// the initial draw card is five, pass the card array and copy the five card
number to the hand card array
    int returnNumberOfCard() ; //return the number of hand card
    int *returncard();//retuen the hand card array which contain the card number
private:
    int playerNumberOfCard;//the number of that player's hand card
    int handCard[54];//an array for store the card number of the card,maxinum is 54 make sure is big enough.
};
```

iii) card class

It is a class to store the rank and suit of cards, includes function to set and get the rank and suit of cards.

```
class Card   //for store one card's rank and suit. total 54 card class to store a poker.
{
public:
    void set(int inputrank, int inputsuit); //define the card's rank and suit
    int getrank() ; //return the rank
```

```
    int getsuit() ;//return the suit
private:
    int rank;  //store the rank,0 is spade,1 is heart,2 is club,3 is diamond, 4 is joker
    int suit; //1-10 for 1-10,11 is J,12 is Q,13 is K, 14 is black joker,15 is red joker
};
```

iv) operationofcardgame class

It is a class to include function for normal game operations. Includes draw card, discard, shuffle and initialize the game. It include the card and player class. It also acts as role of bridge between each class.

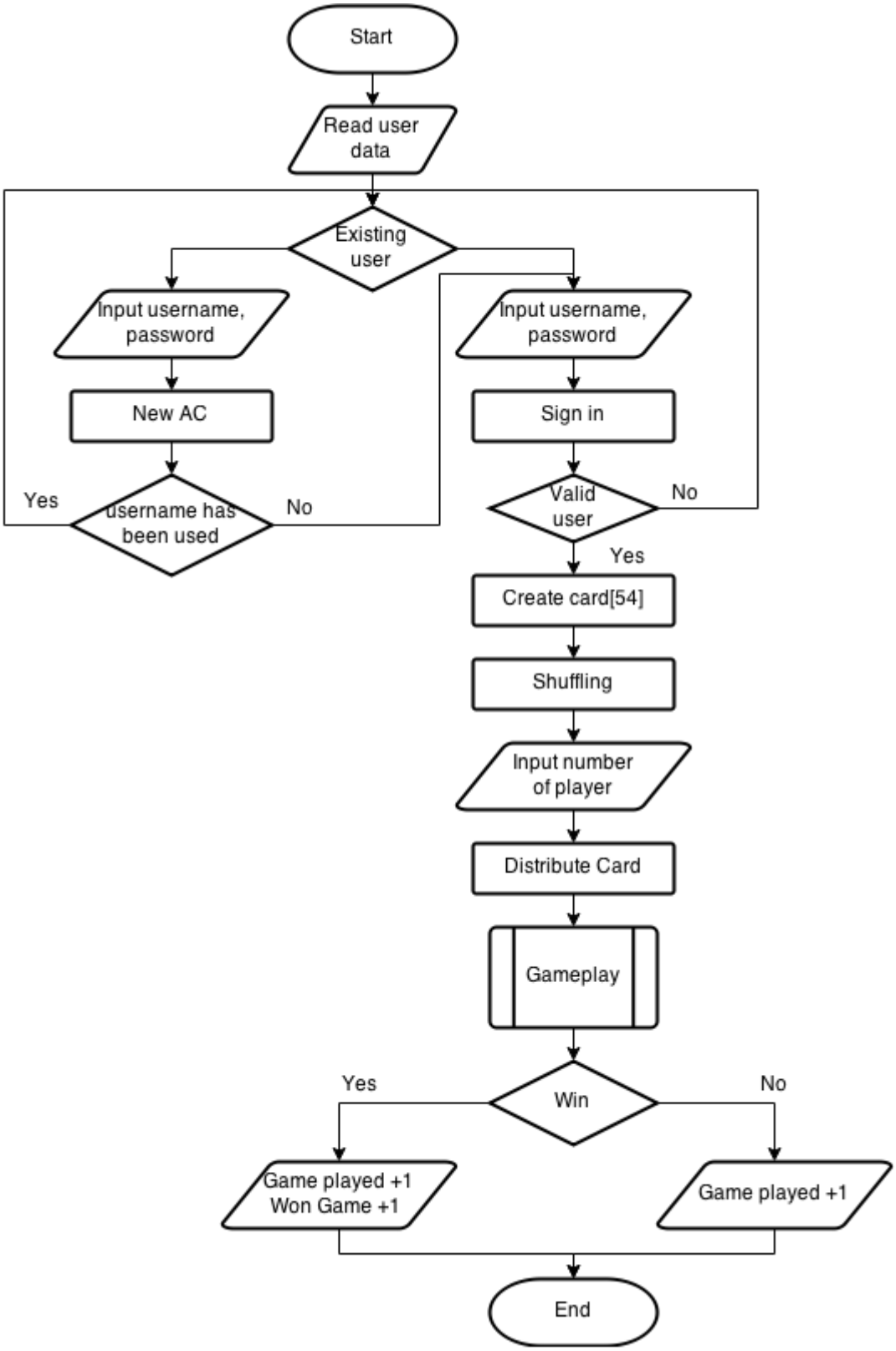
```
class OperationOfCardGame{
public:
    OperationOfCardGame(int inputNumberOfPlayer);
        int randomNum(int number);// for random a number which is 0 to number-1
    void ShufflingUsedCard();//for shuffling the used card and put the card in to the cardArray
    void Shuffling();//for shuffling all card
    void placeCardInSequence();//for start a new game to shuffling
    void createPlayer();//create the player class depends on the numberOfPlayer(computer + human)
```

```
   void initialDrawCard();//give each player five card then get the first card number, if the card is function card,
it will place to the used card array and get another card until the card is normal card store card number to the
startCard
   void drawCardToPlayer(int playerNum,int num);// draw the number of card to the player, if not enough card,it will
call ShufflingUsedCard()
   void giveOutCard(int playerNum,int cardNum);//the player give out the card of that card number
      int *getRecordCard();//return the record card for GUI showing the previous card
      int getStartCard();
      int getNumberOfPlayer();
      int getNumberOfRecordCard();
      int getNumberOfCard();
      int getNumberOfUsedCard();
      void setCardValue();//create 54 card
   ~OperationOfCardGame();    //free the memory of the heap
      Player *player;
      Card *card;
      void showcard();//for testing
private:
```

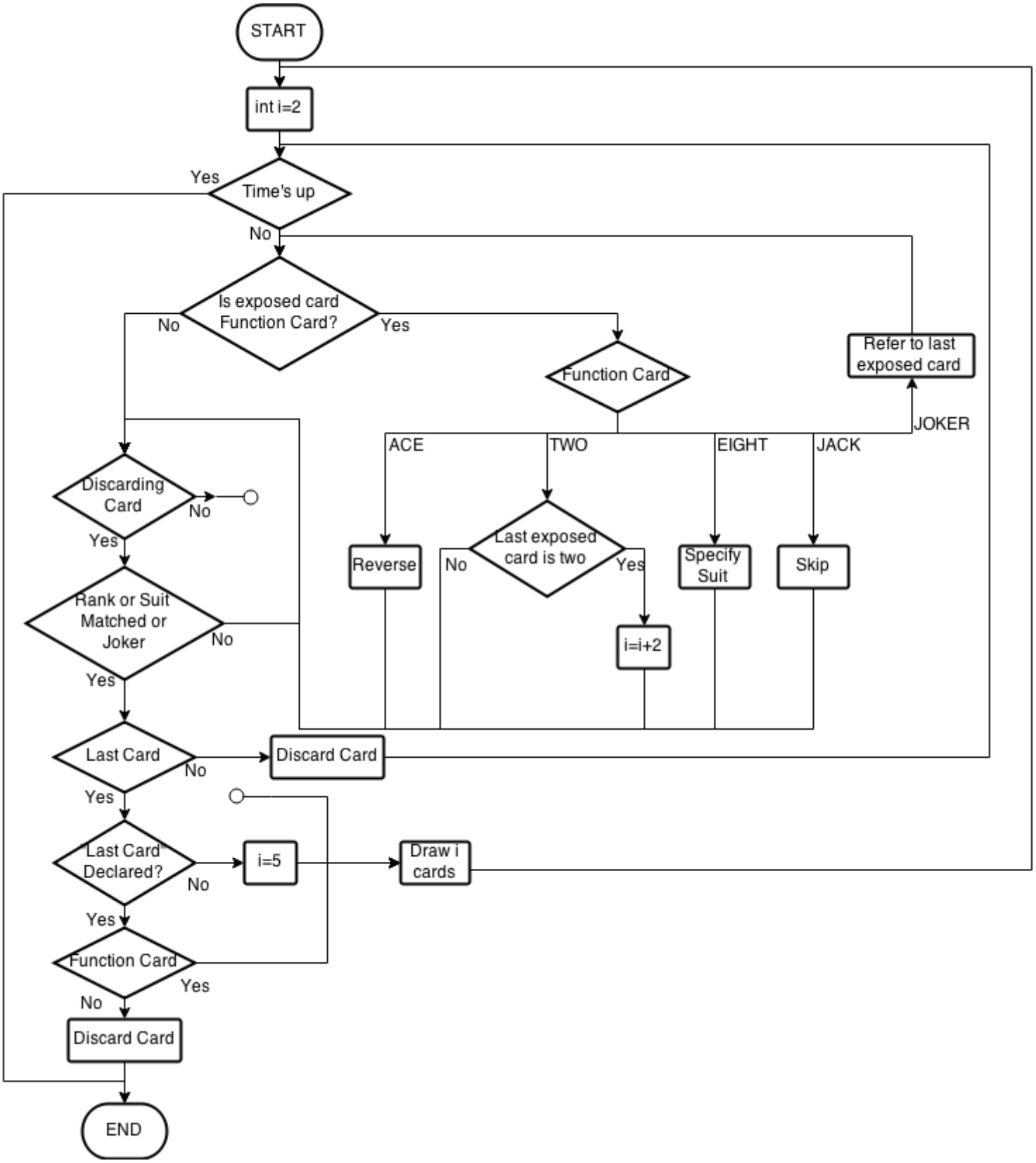
```
        int startCard;
    int cardArray[54];//like a stockpile
    int usedCard[54];//like a discard-pile
        int recordCard[54];//store the give out card history
        int numberOfRecordCard;
    int numberOfCard;
    int numberOfUsedCard;
    int numberOfPlayer;
};

class rule{
public:
        rule(OperationOfCardGame * Og);
        bool allowToDiscard(int cardNum);//check if allow to discard
        void discard(int cardNum);//update the rank suit and update the card that need to draw if rank=2.
        void redraw();
        void updatesuit(int insuit);//for "8" function card
        int getdraw();
```

```
        int getrank();
        int getsuit();
private:
        OperationOfCardGame* ruleOg;
        int draw;
        int suit;
        int rank;
        bool drawed;
};
```

Main Program Flow



Gameplay


START
int i=2
Yes
Time's up
No
No
Is exposed card Function Card?
Yes
Function Card
Refer to last exposed card
JOKER
ACE
TWO
EIGHT
JACK
Discarding Card
No
Yes
Reverse
No
Last exposed card is two
Yes
Specify Suit
Skip
i=i+2
Rank or Suit Matched or Joker
No
Yes
Last Card
No
Discard Card
Yes
Last Card Declared?
No
i=5
Draw i cards
Yes
Function Card
Yes
No
Discard Card
END

Problem

Shuffling

In a card game, shuffling is very important. It is difficult to find a good method to shuffling the card completely. We try many methods which can only shuffle the card in a specific pattern, which is not shuffle completely. To solve this problem, we use a random function which generate number base on system time, which will not repeated. It made our shuffling succeed.

When we programming the class function for shuffling the used card, we find a bug that some card will disappear and some card is repeated after calling the function, but the total number of card is still 54. We do not know why this is happened. Then, we run the debugger, and find out that “int random = rand()%numberOfUsedCard-1;” will return value of -1 to numberOfUsedCard-1, it has chance to create a negative value. However, the random number is used to shuffling the card by change the array with the array of that random number. The array name with negative number will lead to unexpected result. That is why some card will disappear. Some card is fine, when the card disappear, it means “int random =rand()%numberOfUsedCard-1;” random a negative value. To solve it, we change it to “int random = rand()%(numberOfUsedCard-1);”, it will now generate the number from 0 to numberOfUsedCard-2, which make the Shuffling functional.

PictureBox

Since picturebox array has not been taught in class, we have to use different picturebox to deal with 54 cards. Which make our program very complicated and difficult to implement. To solve this, we made a set of image which show difference number of card to replace lot of picturebox. Which make our program more simple and clear.

For showing the hand card, we use the feature of switch by the number of the hand card. Since we do not add break, the statement stated will be call in descending order. For example, when the player has 24 hand card, it will run all the code below

the case 24. Then, all 24 pictureboxs will display the hand card of that player.

```
switch ((pGame->player+gamer)->returnNumberOfCard()){
    case 24: cardName(pGame->card[*(pGame->player[gamer].returncard()+23)].getrank(),pGame->card[*(pGame->player[gamer].returncard()+23)].getsuit());/
             pictureBox24->Image = Image::FromFile("card\\"+s+r+".gif");
             pictureBox24->Enabled = true;
             pictureBox24->Visible = true;
    case 23: cardName(pGame->card[*(pGame->player[gamer].returncard()+22)].getrank(),pGame->card[*(pGame->player[gamer].returncard()+22)].getsuit());
             pictureBox23->Image = Image::FromFile("card\\"+s+r+".gif");
             pictureBox23->Enabled = true;
             pictureBox23->Visible = true;
    case 22: cardName(pGame->card[*(pGame->player[gamer].returncard()+21)].getrank(),pGame->card[*(pGame->player[gamer].returncard()+21)].getsuit());
             pictureBox22->Image = Image::FromFile("card\\"+s+r+".gif");
             pictureBox22->Enabled = true;
             pictureBox22->Visible = true;
    case 21: cardName(pGame->card[*(pGame->player[gamer].returncard()+20)].getrank(),pGame->card[*(pGame->player[gamer].returncard()+20)].getsuit());
             pictureBox21->Image = Image::FromFile("card\\"+s+r+".gif");
             pictureBox21->Enabled = true;
             pictureBox21->Visible = true;
    case 20: cardName(pGame->card[*(pGame->player[gamer].returncard()+19)].getrank(),pGame->card[*(pGame->player[gamer].returncard()+19)].getsuit());
             pictureBox20->Image = Image::FromFile("card\\"+s+r+".gif");
             pictureBox20->Enabled = true;
             pictureBox20->Visible = true;
    case 19: cardName(pGame->card[*(pGame->player[gamer].returncard()+18)].getrank(),pGame->card[*(pGame->player[gamer].returncard()+18)].getsuit());
             pictureBox19->Image = Image::FromFile("card\\"+s+r+".gif");
             pictureBox19->Enabled = true;
             pictureBox19->Visible = true;
    case 18: cardName(pGame->card[*(pGame->player[gamer].returncard()+17)].getrank(),pGame->card[*(pGame->player[gamer].returncard()+17)].getsuit());
             pictureBox18->Image = Image::FromFile("card\\"+s+r+".gif");
             pictureBox18->Enabled = true;
             pictureBox18->Visible = true;
```

GUI Problems

Since the code in GUI is different from Command line interface, we finds difficulties to make and GUI interface. We also finds difficulties to turn the card value to image. Also, since there are many code in GUI header, it is difficult to debug when error occurred. Which take us many time to implement the GUI. Beside, the pointer of GUI also make us confused. Sometime our library can't work with the GUI well since there are different type of information.

GDI+ Problems

We want to play some animation when there is draw card or discard, but actually, we need 8 animation, draw card to left draw card to right, draw card to up, draw card to down, up discard, down discard, left discard, right discard. But there is only one paint event handler. We create 8 Boolean variable, to indicate each animation, and a bool flag to check if any other called the paint to run animation, to make sure only one animation will be displayed at the same time.

Text File

There are many information which have to store in the text file. To modify the text file, we have to use difference function to read and write the text file every time. It is also difficult to manage the user information. We have to make sure the information will not be repeated. Also, we have to make the check of username case insensitive, which also takes us time to find ways to check every character ascii code.

Sorting

In ranking, we have to sort the player by their score. We use insertion sort to sort the score and place them in descending order to show the best 10 player in game.

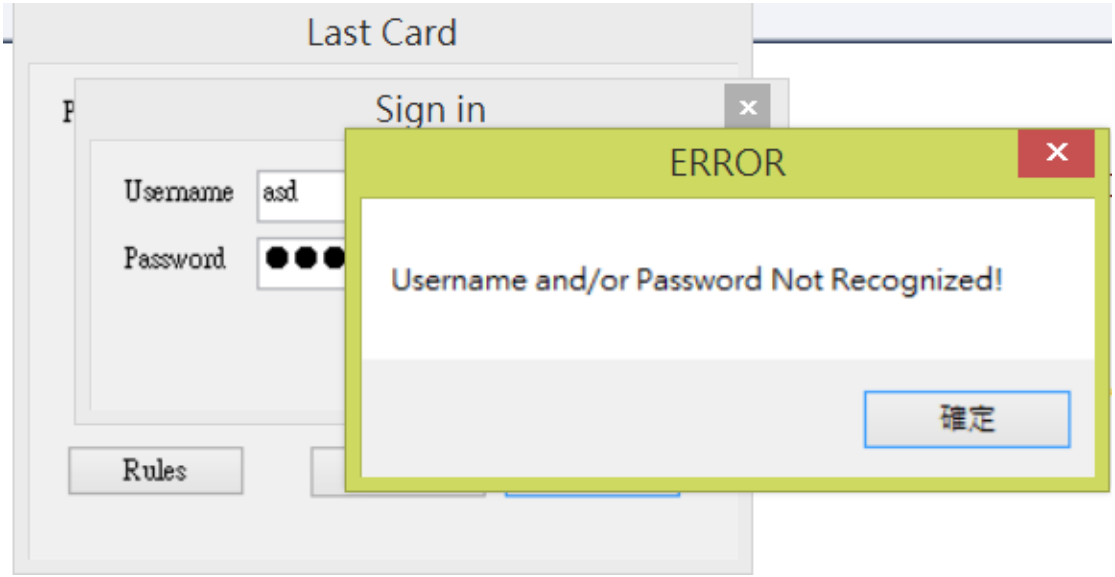
Testing

Testing the login system

When we have not log in and try to start game. It will pop up an error box.

It is okay to click the ranking when no player log in.

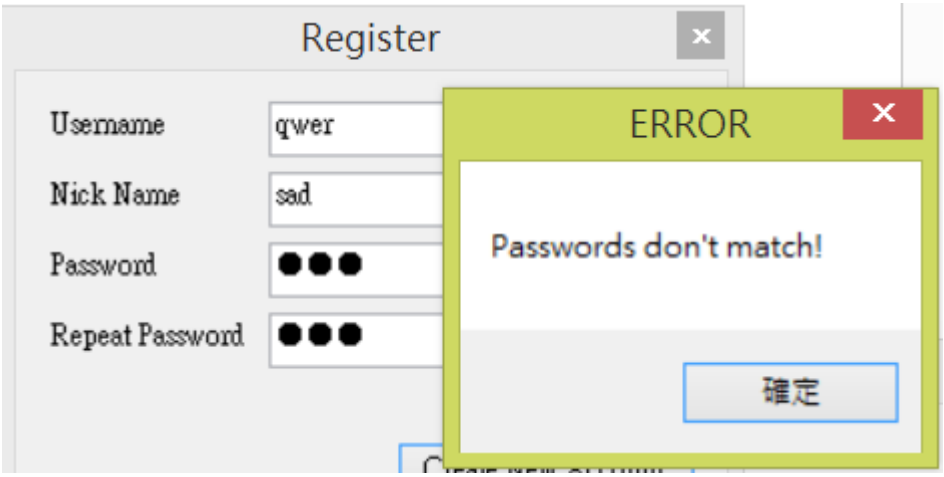
When log in with unregister username or wrong password, it will pop up error box



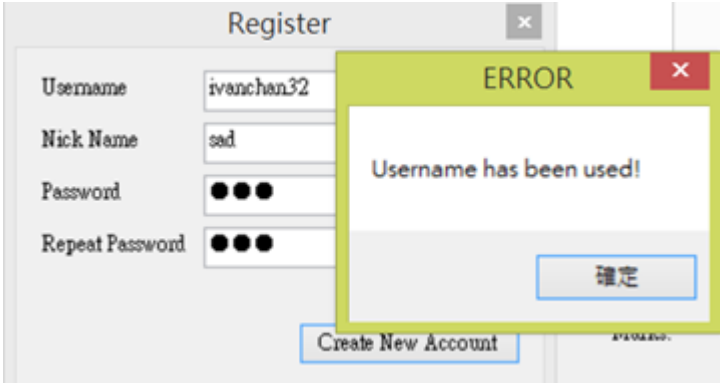
When log in with correct AC, it will show correct username, nick name, mark.

The rule shows correctly without log in or logged in.

When register the new ac, it will check password is match or not.



If the username has been used, it will pop up a error



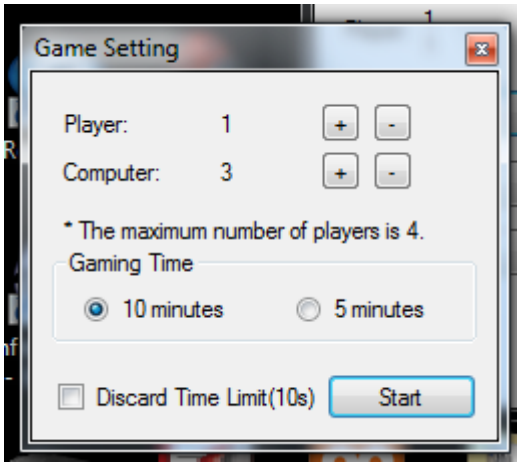
The maximum number of player is 4 and minimum is 1.

The maximum number of computer is 3 and minimum is 0.

But the number of computer cannot be 0 when player is 1.

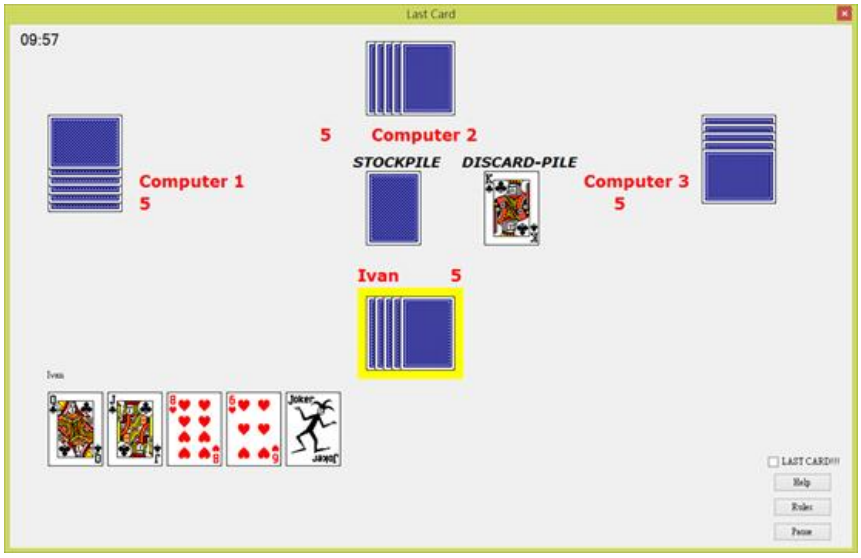
We test each combination of player number and computer number.

There are also option in Gaming Time and if there are discarding time limit.

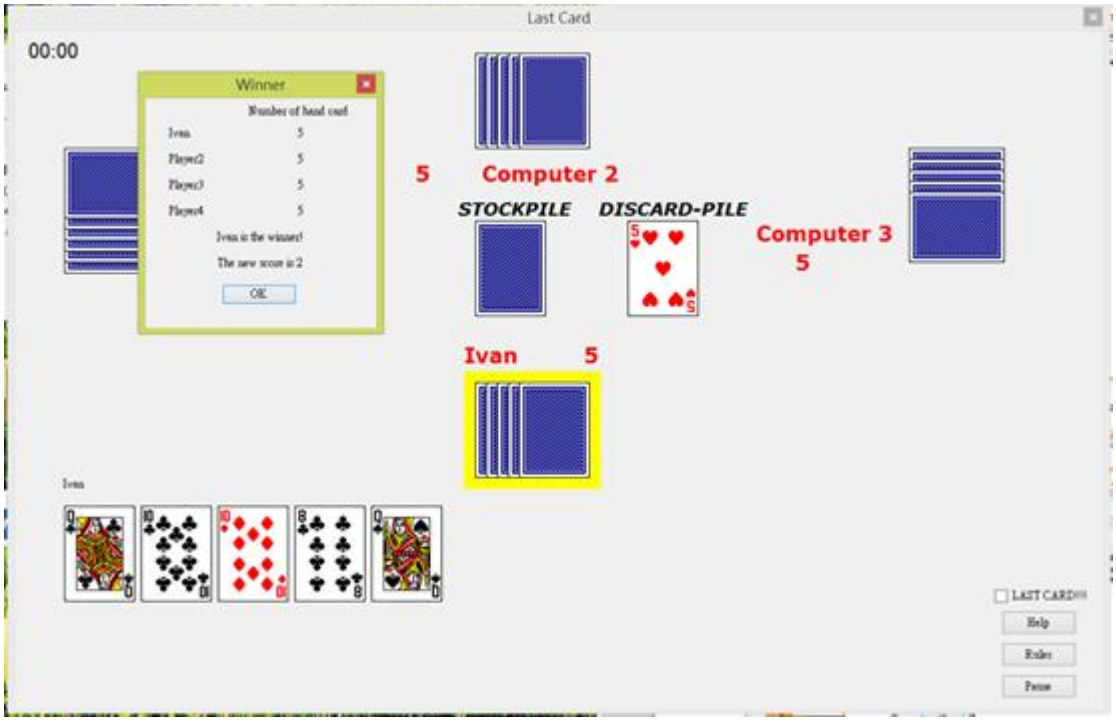


When rerun the program, it can still logged in. it means the data has been success write in to the txt file.

The timer runs correctly.

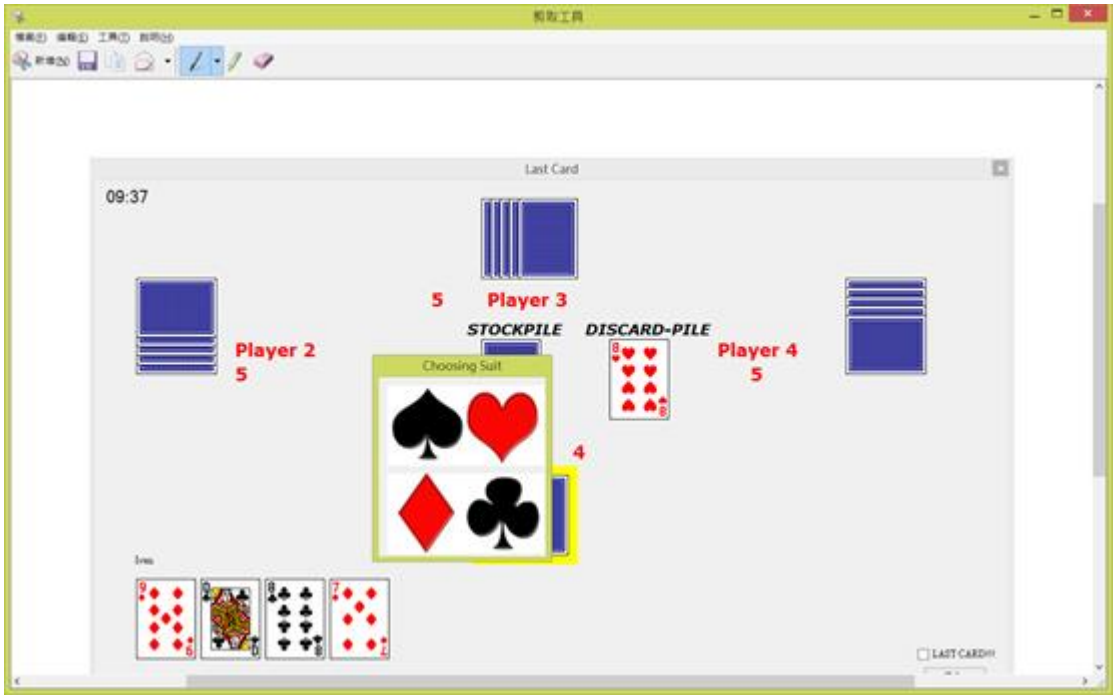


When time is up, a form will pop up and count the gamer card (if they have same card, the first gamer of the same card will win), if player win, the mark will update. Now, the score of ivanchan32 is 2.

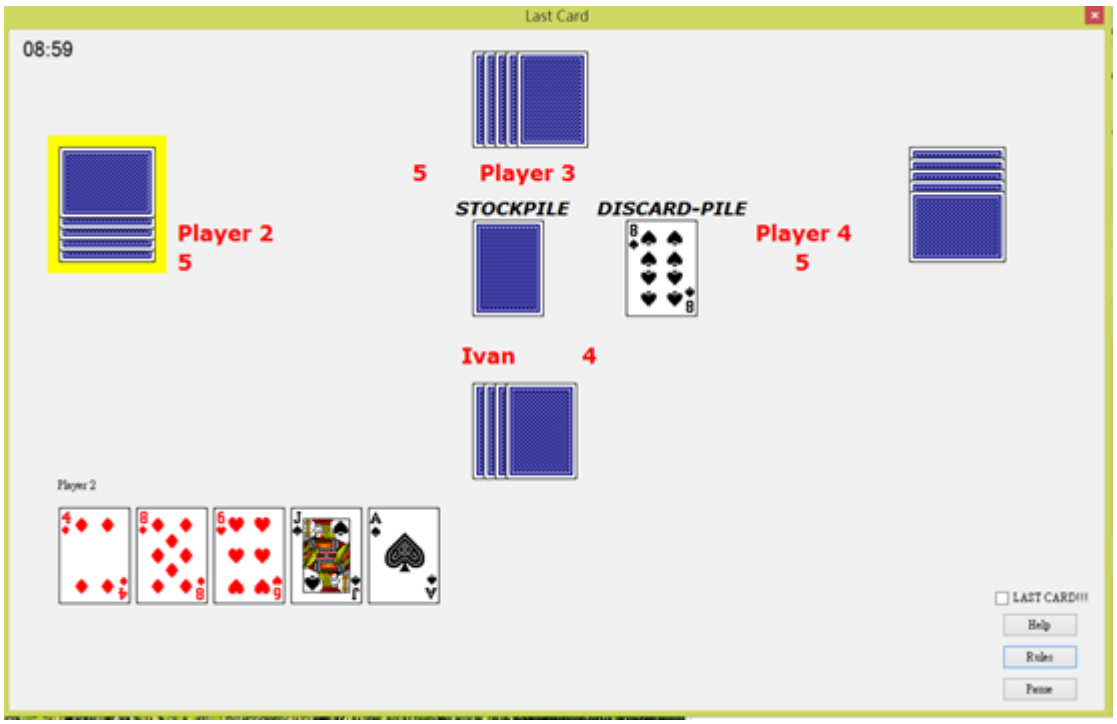


When clicking OK, It will back to menu, the mark updated.

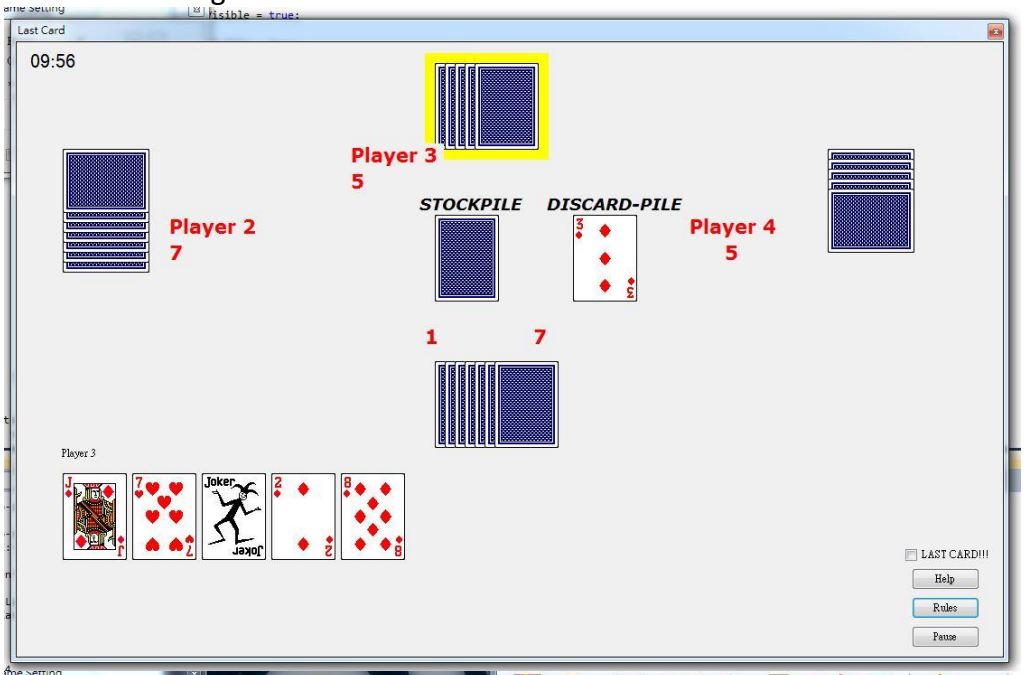
When discard “8” it will pop up a form to choose the suit, for example, choose spade.

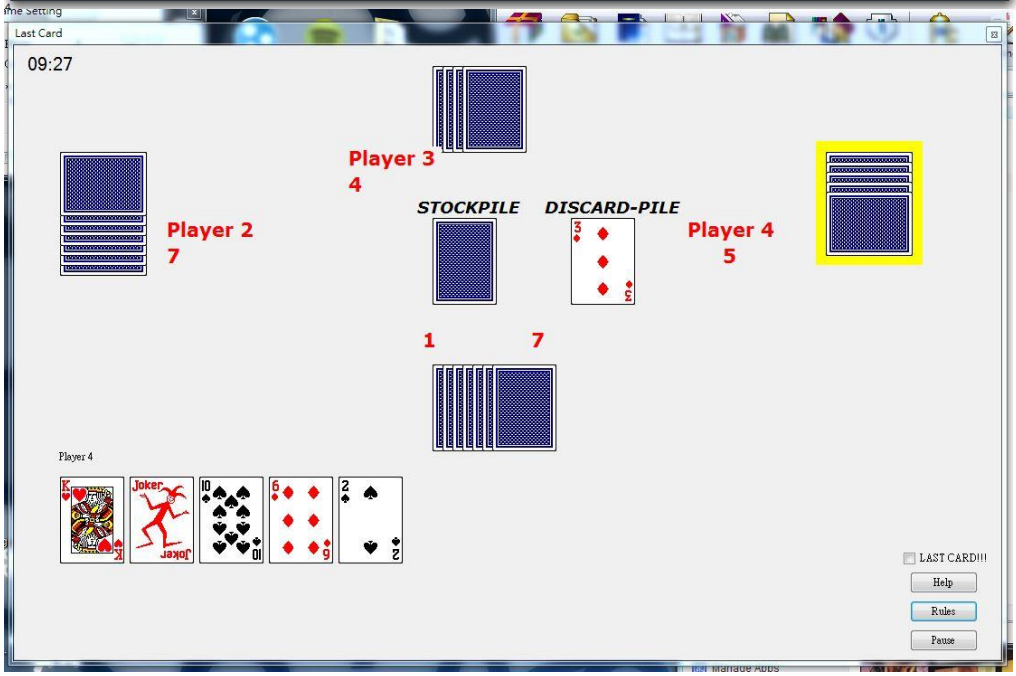


The discard-pile card will become spade for player easier to follow.



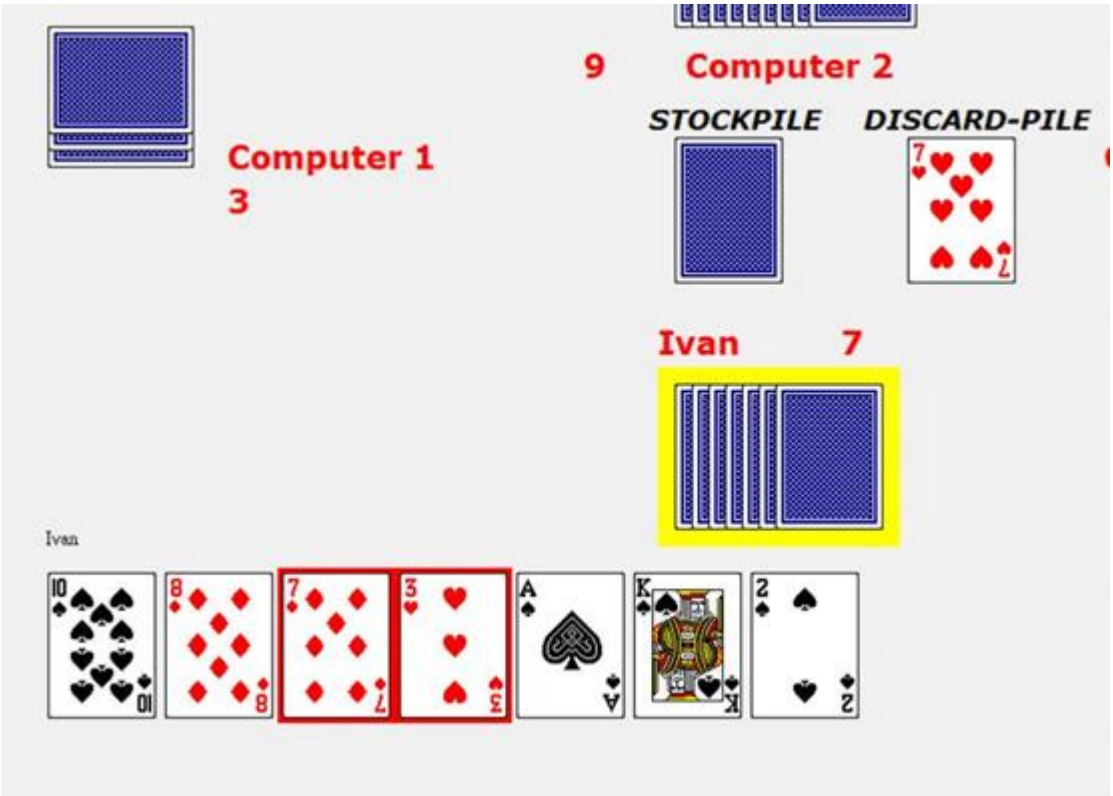
As Joker is a wild card, when player discard joker, the image shown in discard-pile will not be changed.



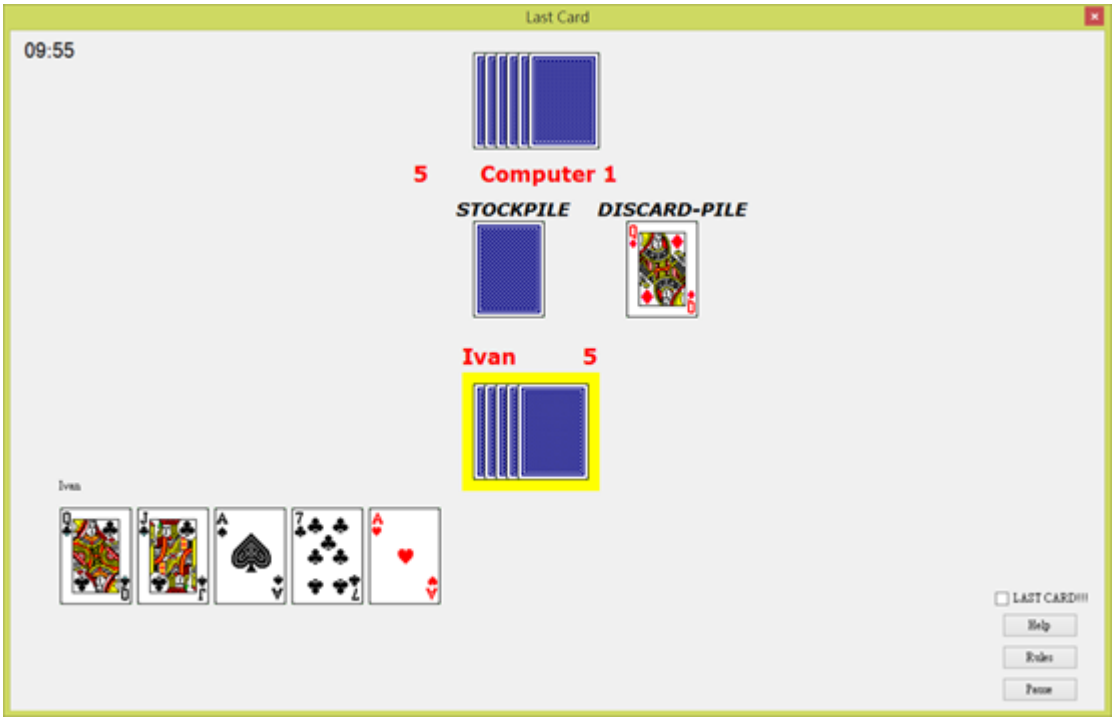


However, the discarding record will also show the Joker.

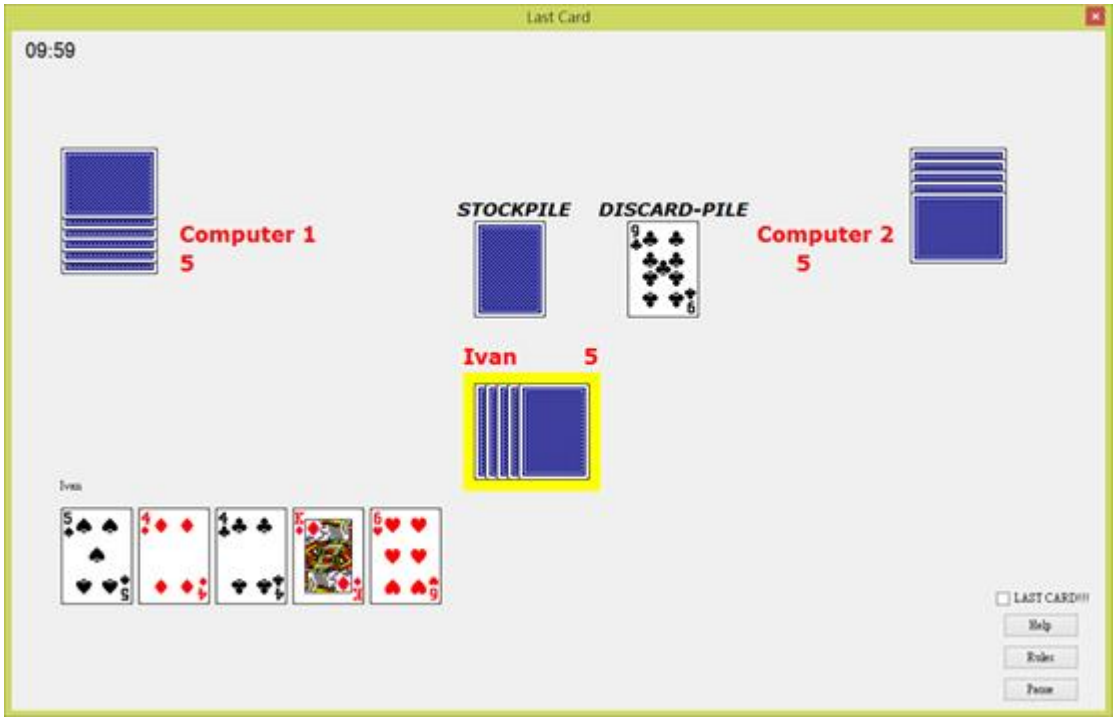
The help function work properly. The cards highlighted in red are able to discard.



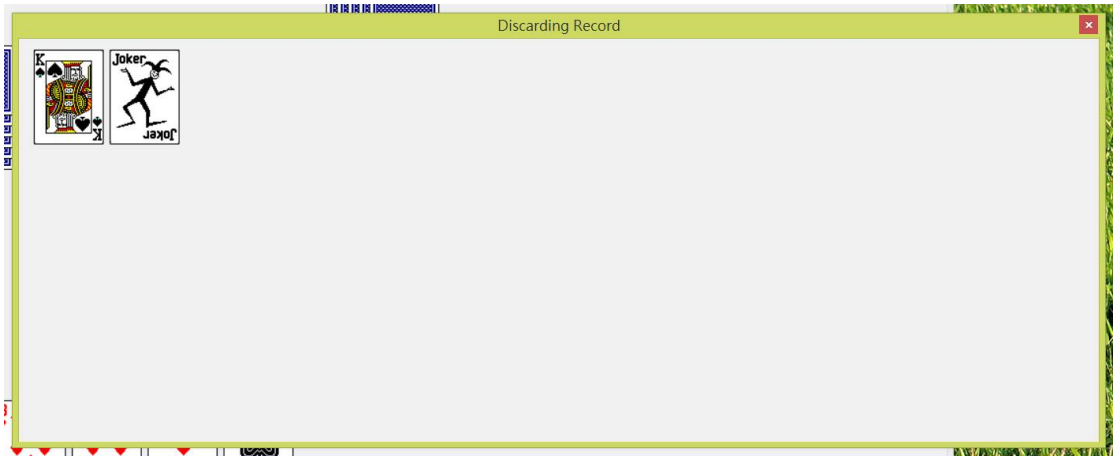
When total gamer is 2, the position will be like this.



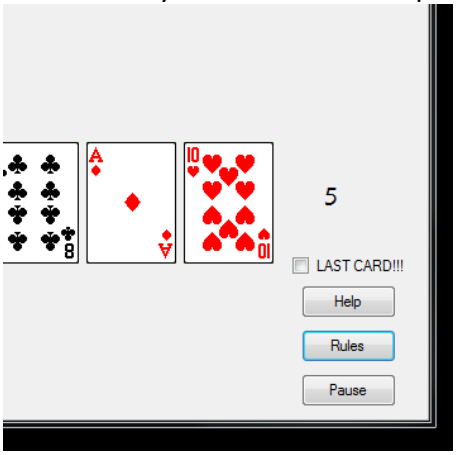
When total gamer is 3, the position will be like this.



User can watch the recorded card by clicking DISCARD-PILE.



There will be a timer counting discarding time as option, when time's up, system will automatically draw cards to the player.



Additional features

In order to make our game more attractive to player and more entertaining, we add several additional features in game.

Set Gaming time

In the setting of game, the game time can be decided. User can choose 5 minutes game or a 10 minutes game in order they want to play longer

Password Verification

In registration process, double entry of password is needed in order to prevent typo. Account will not be created in case there are two different passwords entered.

Pause the game

If the player needs to be excused in game, he can press the pause buttons to pause the game. If the game has been pause, no one can discard or draw card to make the game fairer.

Record discarded card

Some of the player will memorize the cards to make game more challenging. In the game, player can press in the discard-pile to check the previous record of discarded card.

Ranking Sort

In ranking, player will know their score and the rank by sorting. The will know their friends score and rank in order to make the game more competitive.

Discard Time Limit

To increase the difficulty, we set a timer to count down 10 seconds, when the player do not discard or draw card until time is up, it will automatically draw card to the player.

User friendly design

Helping mode

If the player finds difficulties in game, he can press the help button, the game will provide several options which can be discarding, and help the player to choose the correct card.

Yellow label & Animation

There is a yellow label to indicate it is which gamer's turn, it is easier for player to follow. Also, we slow the computer discard, if we do not slow it, computer can finish action within a second, we add some animation, when the computer discard or draw card, it will show an animation, to let the player know the action of computer.

Conclusion

Through this project, we both found our technique to implement program with c++ improved. We really think it is a great achievement that we can build a game by ourselves. We also get more familiarize the tool in GUI. We explode many ways the deal with the last card game. In this project, we also learn the way to shuffle and sorting which have not learnt from lesson. We think the project improve the ways we think and problem solving. It also makes us more familiarize with program logic. We enjoy doing this project when we solve problems we faced.

Future Development

If more time is allowed, our game will be more challenging and more attractive. We can make the AI more smarts. If our skills are more advanced, we can also make the interface more attractive. The animation will be more attractive and smoother. We can also want to make the game interactive, which can communicate with other computer via network, which will make the multiplayer mode more fun.

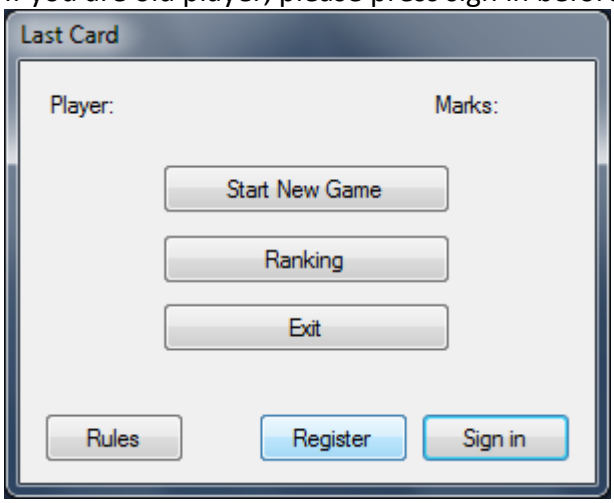
Appendix

User Manual

User Manual

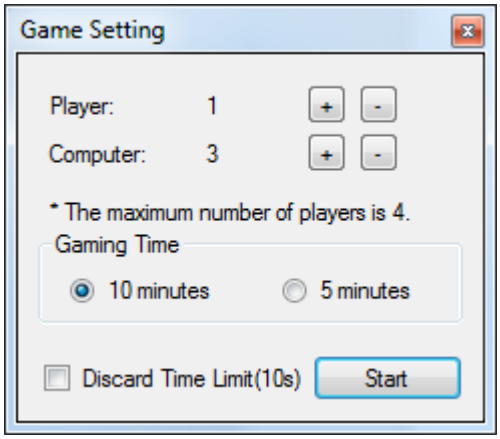
Registration

If you are new to the game, please press the register button to create a new account. If you are old player, please press sign in before you start the game.



Enter Game

When you are really to play, press “Start New Game”, a setting interface will let you choose the number of player you want. If you want to play with your friend, you can enter more number in player. If you play alone, you can enter computer player to play with you. The minimum number of player is two and maximum number of player is four.



Game playing

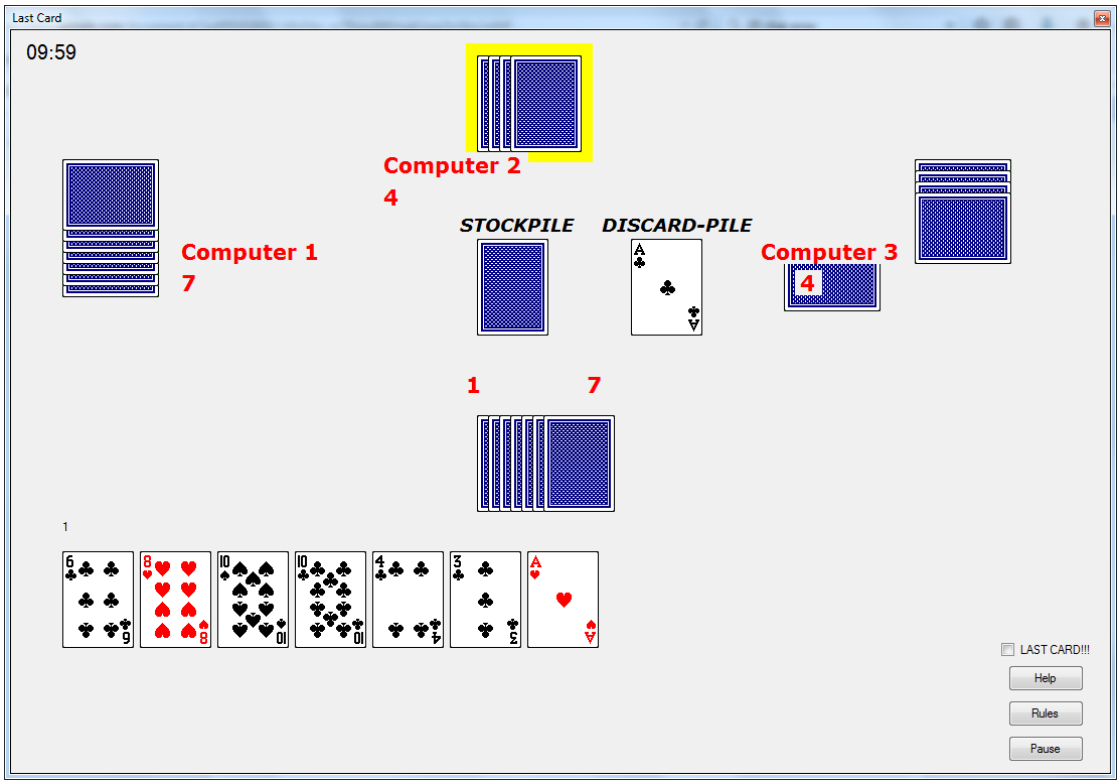
All the card you have will be show in the bottom of window. You should press the card you have to discard it. If you don't know which card to discard, you can press help button. All cards can be discard will be highlighted in red.

If you want to check to record of card, press on discard-pile, a window will be showed.

Before discard your second last card, you should check the last card checkbox.

If you find difficulty in game, you can press "rule" any time to see the rule of game.

When you press pause, the game will be paused and resume when you press OK.



-END-