Project 1

<Black Jack (21 Point)>

CSC 5 - 48102

Name:Weikang Du

Date: 10/28/16

Menu

Introduction

Title : Black Jack (21 Point)

Actually, I changed some rule of this game:
There are 52 cards, and each round has 4 times to send card, each time for 1 card.

I made the game to Player vs. Computer. There are two mode for player:

1. Normal Player: Player is able to know how many points in hand, but he/she could not know the next card number, therefore, this mode is hard one and player want to get closer 21 but not exceeds 21 as possible.

2. Super Player: Player also can know how many points in hand, and he/she has power to see the number of next card and decide to choose or deny.

Back to Menu

Summary

Project Size: about 300 lines

The number of variables: about 16

I made it around one week, and developed it one week.
It was hard to make the partial of computer, therefore,
the computer can predict whether it will exceed 21. So
the computer will never exceed 21, however, the player
can not predict that.
And also, I used some array I learned from internet and book.

Back to Menu

Flowchart

Using Cliffy

Click for Flowchart

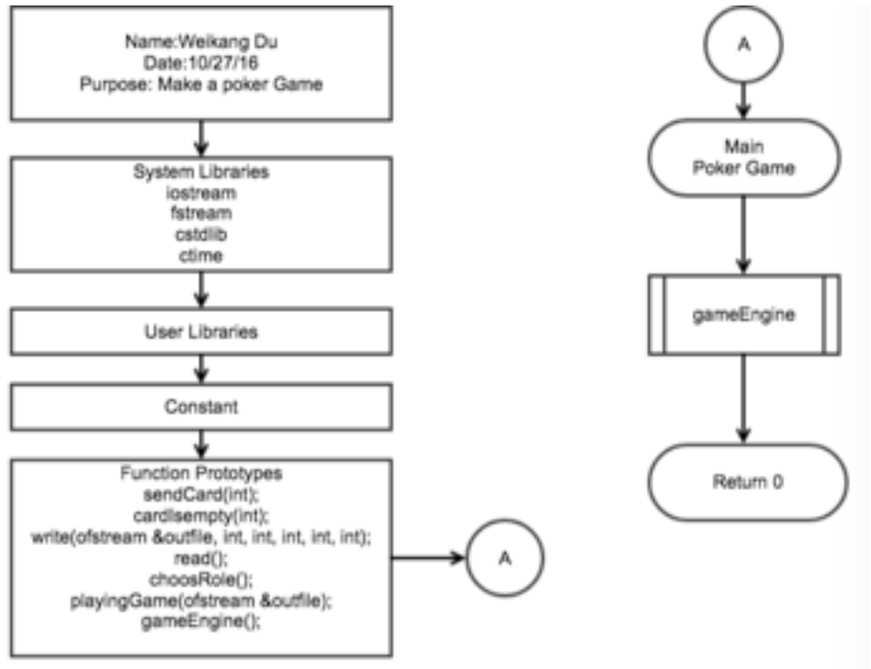
Back to Menu

SetIcon	Describtion
card[13]={4,4,4,4,4,4,4,4,4,4,4,4,4}	There are 13 kinds of card and each kind for 4, so total card number is 52.
computerCount	The number of times the computer got the card
playerCount	The number of times the player got the card
rounds	The counter for round
cardIndex	Determine the each kind of card
flag	Make sure whether card is empty
sum_computer	The total points in hand of computer
sum_player	The total points in hand of player
tmd_card	Temporary card random number
choice	Choose card or Deny
role	The game mode

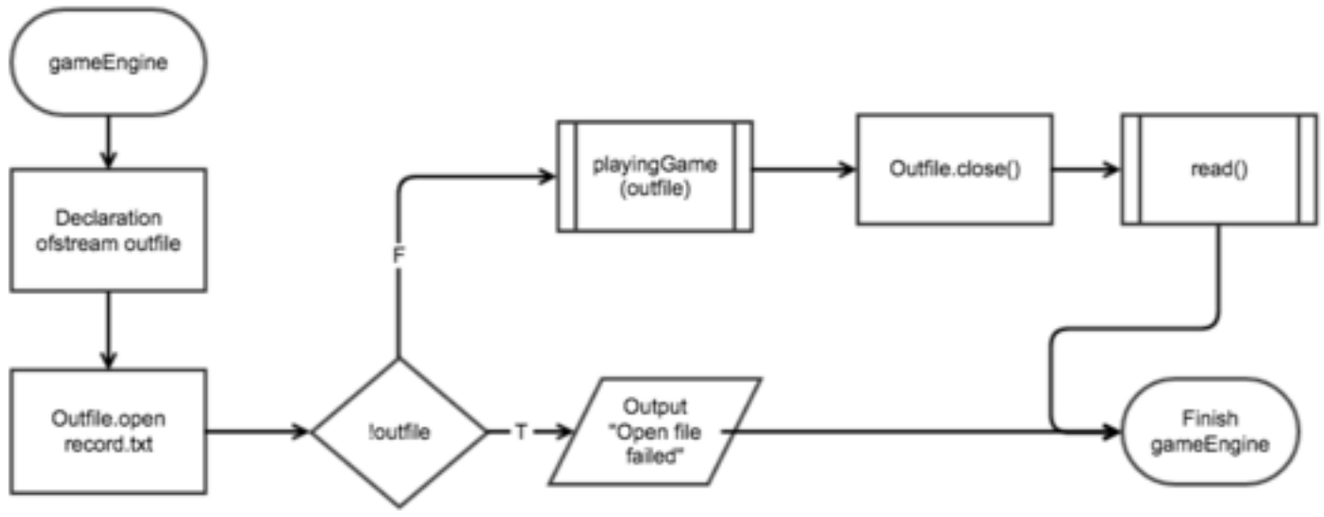
Back to Menu

Pseudo Code

Opening comments
All using system Libraries
then, set the function prototypes
enter main
function(gameEngine)
exit the program return 0



gameEngine function
Declaration the variable
open the file 'record.txt' for output
verify open success or not
if not then finish the function
otherwise, function playingGame
then close the file
and read the file again for output
then finish the function



playingGame

Declaration the variables
set the 'role' get from function choosRole
set the seed for random
use while loop check function cardIsempty true of false
use while lop limit the card counter
then send computer card
computer can cheat for determine voiding exceed maximum
if not exceed maximum then add this number to computer total
then send player card
if role is super then ask user that look the card number or not
if does, then output the card number
and then ask want the card or not
if dose, then add the card number to total
card counter increment then determine by while loop above.
then increment the round number

Back to Menu

write function
ask the player for continue play round or not
if dose, set the prior total data to 0
if not then break the while loop for check cardIsempty

cardIsempty

set the true flag
use for loop
check each element in card array is greater then 0 or not
if does let the flag false
and break the the for loop
return this flag

ChoosRole

declaration the variable choice
out put the message and choice
get the input
verify the valid input using while loop
then switch the choice
1. set choice equal 1, break the switch

Back to Menu

2. set choice equal 2, break the switch
3. display the info. and menu, then input choice again
valid choice or not using while loop, break the switch
4. exit the program
while loop the switch if choice not equal 1 or 2
return choice;

sendCard

declaration the variable
use a unlimited while loop
give a random number to card between 0-12
if cards array for this element is greater than 0
the number of this element decrement
then break the while loop
and return the card number+1

write

if not open the file exit with 0
otherwise, use for loop output to file the card computer get each time
use for loop output to file card play get each time

Back to Menu

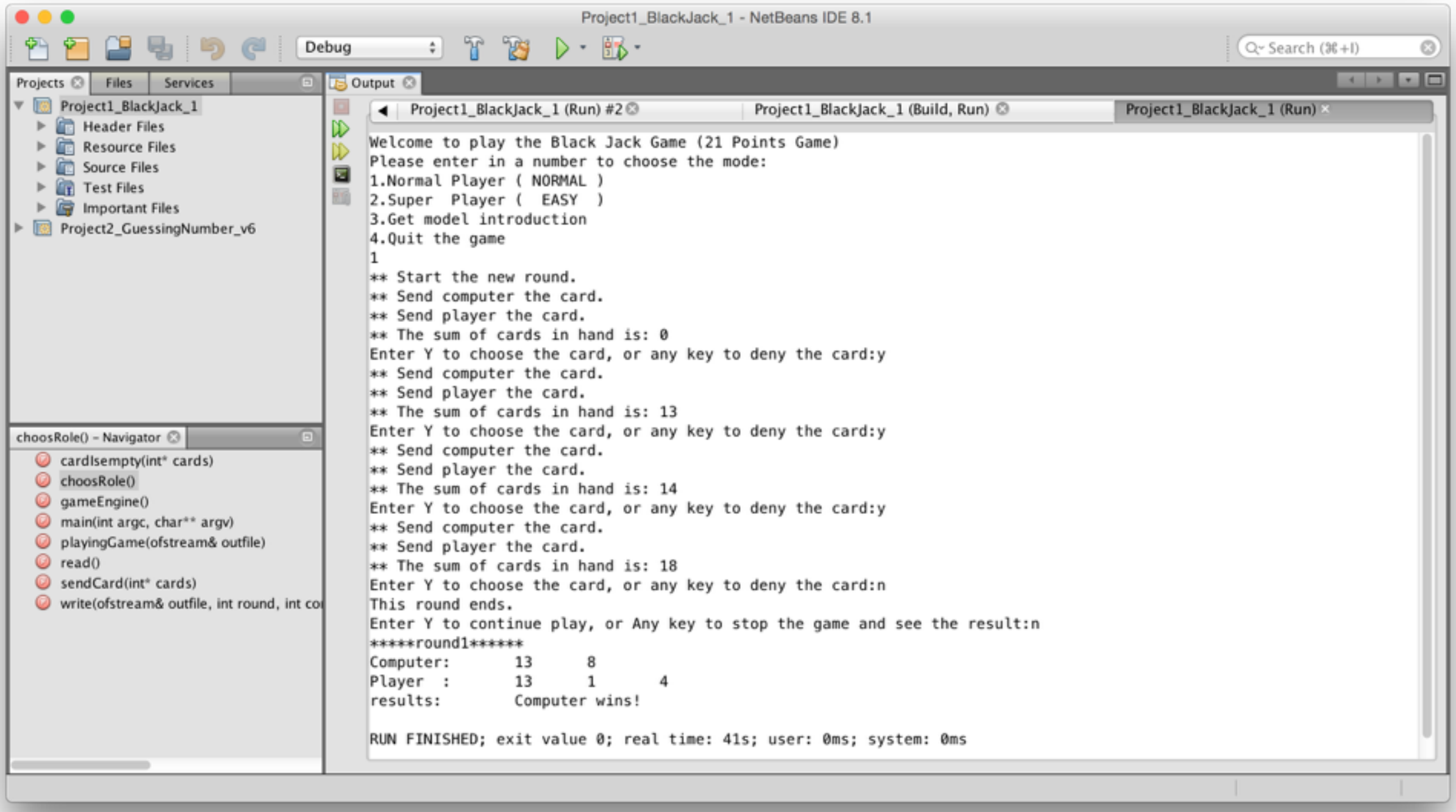
determine the result
use if and else if
then output to file the result

read

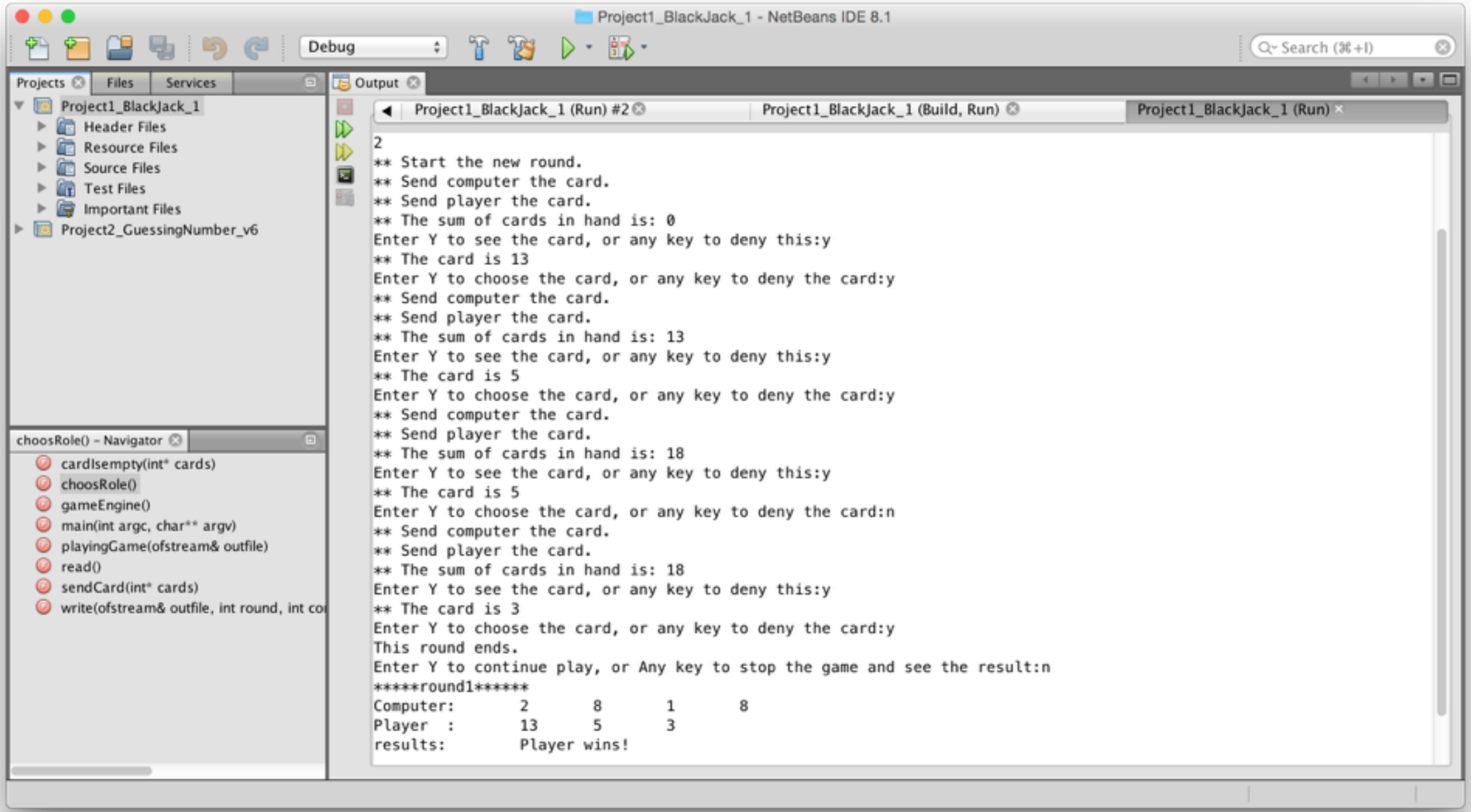
declaration the if stream infield
then open the file record.txt for get data
if not open then exit
otherwise, use while loop
getline the data
if '' then break the while loop
otherwise display the data
then close the file

Back to Menu

Sample I/O



Back to Menu


Project1_BlackJack_1 - NetBeans IDE 8.1
Debug
Search (⌘+I)
Projects
Files
Services
Project1_BlackJack_1
Header Files
Resource Files
Source Files
Test Files
Important Files
Project2_GuessingNumber_v6
choosRole() - Navigator
cardIsempty(int* cards)
choosRole()
gameEngine()
main(int argc, char** argv)
playingGame(ofstream& outfile)
read()
sendCard(int* cards)
write(ofstream& outfile, int round, int co
Output
Project1_BlackJack_1 (Run) #2
Project1_BlackJack_1 (Build, Run)
Project1_BlackJack_1 (Run)
2
** Start the new round.
** Send computer the card.
** Send player the card.
** The sum of cards in hand is: 0
Enter Y to see the card, or any key to deny this:y
** The card is 13
Enter Y to choose the card, or any key to deny the card:y
** Send computer the card.
** Send player the card.
** The sum of cards in hand is: 13
Enter Y to see the card, or any key to deny this:y
** The card is 5
Enter Y to choose the card, or any key to deny the card:y
** Send computer the card.
** Send player the card.
** The sum of cards in hand is: 18
Enter Y to see the card, or any key to deny this:y
** The card is 5
Enter Y to choose the card, or any key to deny the card:n
** Send computer the card.
** Send player the card.
** The sum of cards in hand is: 18
Enter Y to see the card, or any key to deny this:y
** The card is 3
Enter Y to choose the card, or any key to deny the card:y
This round ends.
Enter Y to continue play, or Any key to stop the game and see the result:n
*****round1******
Computer: 2 8 1 8
Player : 13 5 3
results: Player wins!

Back to Menu

Project1_BlackJack_1 - NetBeans IDE 8.1

Debug

Search (⌘+I)

Projects | Files | Services

- Project1_BlackJack_1
  - Header Files
  - Resource Files
  - Source Files
  - Test Files
  - Important Files
- Project2_GuessingNumber_v6

choosRole() - Navigator

- cardIsempty(int* cards)
- choosRole()
- gameEngine()
- main(int argc, char** argv)
- playingGame(ofstream& outfile)
- read()
- sendCard(int* cards)
- write(ofstream& outfile, int round, int co

Output

Project1_BlackJack_1 (Run) #2 | Project1_BlackJack_1 (Build, Run) | Project1_BlackJack_1 (Run)

```
Welcome to play the Black Jack Game (21 Points Game)
Please enter in a number to choose the mode:
1.Normal Player ( NORMAL )
2.Super  Player (  EASY  )
3.Get model introduction
4.Quit the game
3
Normal mode: You can not know the card number when you decidingchoose or not.
Super mode: You can see the card number before you deciding

Please enter in a number to choose the mode:
1.Normal Player ( NORMAL )
2.Super  Player (  EASY  )
3.Get model introduction
4.Quit the game
1
** Start the new round.
** Send computer the card.
** Send player the card.
** The sum of cards in hand is: 0
Enter Y to choose the card, or any key to deny the card:
```

Project1_BlackJack_1 (Run)

Back to Menu

Project1_BlackJack_1 - NetBeans IDE 8.1

Debug

Search (⌘+I)

Projects | Files | Services

- Project1_BlackJack_1
  - Header Files
  - Resource Files
  - Source Files
  - Test Files
  - Important Files
- Project2_GuessingNumber_v6

choosRole() - Navigator

- cardIsempty(int* cards)
- choosRole()
- gameEngine()
- main(int argc, char** argv)
- playingGame(ofstream& outfile)
- read()
- sendCard(int* cards)
- write(ofstream& outfile, int round, int co

Output

Project1_BlackJack_1 (Run) #2 | Project1_BlackJack_1 (Build, Run) | Project1_BlackJack_1 (Run)

```
Welcome to play the Black Jack Game (21 Points Game)
Please enter in a number to choose the mode:
1.Normal Player ( NORMAL )
2.Super  Player (  EASY  )
3.Get model introduction
4.Quit the game
4

RUN FINISHED; exit value 0; real time: 1s; user: 0ms; system: 0ms
```

Back to Menu

records.txt

```
******round1******
Computer:       12      5
Player  :       11      7
results:        Player wins!
```

Program

```
/*
 * File:   main.cpp
 * Author: WeikangDu
 * Created on October 27, 2016, 7:07 PM
 * Purpose: Make a Poker Game (Black Jack)
 */

#include<iostream> //Input and output
#include<fstream>  //I/O file
#include<string>
#include<cstdlib>  //For the random number
#include<ctime>    //Time

using namespace std;
//User Libraries

//Global Constants
```

Back to Menu

```
//Function Prototypes
int sendCard(int);
bool cardIsempty(int);
void write(ofstream &outfile, int, int, int, int, int);
void read();
int choosRole();
void playingGame(ofstream &outfile);
void gameEngine();

//*********************************************************
// Main function here!!                                   *
//*********************************************************
int main(int argc, char** argv)
{
    gameEngine();

    return 0;
}
```

Back to Menu

```
//*************************************************************
// Definition of function sendCard                            *
// This function limit the number and kind of card            *
//*************************************************************
int sendCard(int *cards)
{
    int cardIndex;
    while (true)
    {
        //Produce a random number
        cardIndex = rand() % 13;
        //Determine whether the same point card exits
        if (cards[cardIndex]>0)
        {
            cards[cardIndex]--;
            break;
        }
    }
    return (cardIndex + 1);
}
```

Back to Menu

```
//*************************************************************
// Definition of function cardIsempty                         *
// This function determine whether here is no card for total *
// and the each kind of card                                  *
//*************************************************************
bool cardIsempty(int *cards)
{
    bool flag=true;
    for (int i=0; i<13; i++)
    {
        if (cards[i]>0)
        {
            flag=false;
            break;
        }
    }
    return flag;
}
```

Back to Menu

```
//*************************************************************
// Definition of function write                               *
// This function Output the result and data to a file,        *
// and determine the winner                                   *
//*************************************************************
void write(ofstream &outfile, int round,int computerCount,int *computer,int playerCount,int
*player)
{
    if (!outfile)
    {
        cout<<"Open file failed!"<<endl;
    }
    else
    {
        int sum_computer=0;
        int sum_player=0;
       //Output the round number
        outfile<<"*****round"<<round<<"******\n";
        outfile<<"Computer:\t";
```

Back to Menu

```
//Output the card number each time for computer
 for (int i=0; i<computerCount; i++)
 {
     sum_computer+=computer[i];
     outfile<<computer[i]<<"\t";
 }
 outfile<<"\n";
 outfile<<"Player  :\t";
//Output the card number each time for player
 for (int i=0; i<playerCount; i++)
 {
     sum_player+=player[i];
     outfile<<player[i]<<"\t";
}
outfile<<"\n";
outfile<<"results:\t";
```

Back to Menu

```
//Determine the result and display it
 if (sum_computer>21 && sum_player>21)
 {
    outfile<<"Computer and Player are both explode!\n";
 }
 else if (sum_computer<=21 && sum_player>21)
 {
    outfile<<"Computer wins!\n";
 }
 else if (sum_computer>21 && sum_player<=21)
 {
    outfile<<"Player wins!\n";
 }
    else if (sum_computer<sum_player)
    {
       outfile<<"Player wins!\n";
    }
    else if (sum_computer>sum_player)
    {
       outfile<<"Computer wins!\n";
    }
    else if (sum_computer==sum_player){
       outfile<<"No one wins!\n";
    }
  }
}
```

Back to Menu

```
//*********************************************************
// Definition of function read                            *
// This function read the data from the file              *
//*********************************************************
void read()
{
    ifstream infile;
    infile.open("records.txt");
    if (!infile)
    {
        cout<<"Open file failed!"<<endl;
    }
else
{
    string str;
    while(!infile.eof())
    {
        getline(infile, str);
        if(str=="")
        {
            break;
        }
```

Back to Menu

```
        else
                {
                    cout<<str<<endl;
                }
            }
            infile.close();
        }
    }
//*********************************************************
// Definition of function choosRole                       *
// This function show the menu and return a mode          *
//*********************************************************
int choosRole()
{
    int choice;
    //Display the menu
    cout<<"Welcome to play the Black Jack Game (21 Points Game)"<<endl;
    cout<<"Please enter in a number to choose the mode:"<<endl;
    cout<<"1.Normal Player ( NORMAL )"<<endl;
    cout<<"2.Super  Player (  EASY  )"<<endl;
    cin>>choice;
    return choice;
}
```

Back to Menu

```
//*************************************************************
// Definition of function playingGame                         *
// This function send the card to computer and player,        *
// and determine whether choose or not the card for each time  *
//*************************************************************
void playingGame(ofstream &outfile)
{
    int tmp_card; //Temporary card number
    char choice;
    int card_count=0; //The times of send card
    int sum_computer,sum_player; //The total number in hand for both
    int role; //chose the mode

//Declaration the Variables
int cards[13]={ 4,4,4,4,4,4,4,4,4,4,4,4,4}; //The 13 kinds of poker and each for 4, no joker
int computer[24];
int computerCount=0; //The number of computer get card
int player[24];
int playerCount=0;   //The number of player get card
int rounds=0;      //Set the round
```

Back to Menu

```
role=choosRole();
cout<<"** Start the new round."<<endl;

srand((unsigned)time(NULL)); //Set the seed for the random number

while (!cardIsempty(cards)) //Call the cardIsempty function
{
    card_count=0;
    sum_computer=0;
    sum_player=0;

    while (card_count<4)
    {
cout << "** Send computer the card.\n";
tmp_card=sendCard(cards); //Call the sendCard function
if (sum_computer+tmp_card<=21)
{
    sum_computer+=tmp_card; //Call the sendCard function
    computer[computerCount]=tmp_card;
    computerCount++;
}
```

Back to Menu

```
cout<<"** Send player the card.\n";
tmp_card=sendCard(cards); //Call the sendCard function

cout<<"** The sum of cards in hand is: "<<sum_player<<endl;
if(role==2) //mode 2 super player
{
    cout<<"Enter Y to see the card, or any key to deny this:";
    cin>>choice;
    if(choice=='Y'||choice=='y')
    {
        cout<<"** The card is "<<tmp_card<<endl;
    }
}
```

```
    cout << "Enter Y to choose the card, or any key to deny the card:";
    cin >> choice;
    if (choice=='Y'||choice=='y')
    {
        sum_player+=tmp_card;
        player[playerCount]=tmp_card;
        playerCount++;
    }
    card_count++;
}
```

Back to Menu

```
rounds++;
write(outfile,rounds,computerCount,computer,playerCount,player);
cout << "This round ends."<<endl;
cout << "Enter Y to continue play, or Any key to stop the game and see the result:";
cin >> choice;

if (choice=='Y'||choice=='y')
{
    computerCount=0;
    playerCount=0;

    for(int j=0; j<24; j++)
    {
      computer[j]=0;
      player[j]=0;
    }
    for(int j=0; j<13; j++)
    {
      cards[j]=4;
      card_count=0;
        continue;
    }
}
```

Back to Menu

```
            {
                break;
            }
        }
    }

//*********************************************************
// Definition of function gameEngine                      *
//*********************************************************
void gameEngine()
{
    ofstream outfile;
    outfile.open("records.txt");
    if (!outfile)
    {
        cout<<"Open file failed!"<<endl;
    }
    else
    {
        playingGame(outfile);
        outfile.close();
        read();
    }
}
```

Back to Menu