Project 2
< Blackjack >

CIS 5 - Section 47165
Spring 2023

By:
Aurelisa Juan Sindhunirmala
06/05/2023

Table of Contents

Introduction

This program is created to be an online version of blackjack. The rules are exactly the same as the rules of blackjack when played in real life. Players must be able to get cards with a value close to or equal to 21 without exceeding it. This version of blackjack is created as a single player playing with a bot (dealer). By using the materials covered over 8 weeks in the CIS-5 course, I am able to make this game with some additional features, such as adding a comment section for people to rate the game after playing.

Summary

Project size : 452 lines

This project implements all the materials learned in Spring 2023. All the libraries, variables, logical operators, loops, and many more are used in this game project. More information on the implementation is listed in the spreadsheet attached. This virtual game of blackjack has similar rules as the usual blackjack. Using C++, I am able to create this simplified version of blackjack.

Game Play and Rules

These are the guidelines to play this online version of Blackjack.

1. The maximum value to bet is $500 and a minimum value of $10.
2. The dealer draws two cards from the deck to each player.
3. The dealer will let one of their cards facing up, the player will have to let two cards facing up the whole time.
4. Jacks, Queens, and Kings have a value of 10. Ace has a value of either 1 or 11, depending on the total value one has. All number cards are worth their own value.
5. The player that has a total value closest or equal to 21 wins.

My Approach to the Game

The game will start with a welcome sign and introduction of the game. Then, it will ask the player’s name and age and store that. After the player hit enter, the menu of the game will be shown; to read the rules of the game, start the game, or exit the game. The player can only comment about the game when they chose the third option: exit the game.

Contracts and Concepts Utilized

This project covers all the materials I have learned during the Spring 2023 semester. All chapters, including Introduction to C++, Expressions and Interactivity, Making Decisions, Loops and Files, Functions, Arrays, and Searching and Sorting Arrays have been implemented in this project. For more information on the implementation, please refer to the spreadsheet attached. The checklist for both Project 1 and Project 2 will be in the spreadsheet.

Development Summary

Version 1

The first version will print the introduction to the game. Furthermore, it will also print out the menu settings, which contains the rules to the game, starting the game instantly, and exiting the game.

Version 2

The second version will start the game, which includes shuffling the deck of cards and drawing the cards to the player.

Version 3

In the third version, I added a boolean expression for the player's answer to be valid.

Version 4

For the fourth version, I added an additional feature to the game, where the player can either bet by themselves or let the computer bet for them. The computer will then choose a random number from the CoinsCasinoCreateData file and declare it as the player's bet.

File: RandomNum_CreateData

This is an input file that displays the result of the random data in rows and columns and to be used when the computer wants to pick a random number.

Version 5

This version marks the first version for Project 2. I added the function prototypes and modified the switch cases by putting them as a function. I also modified some of the boolean and ternary operators.

Version 6

In this version, I added the default arguments function and more functions. I also modified the game so that the player can play again if they want to without restarting the whole process from the beginning. After the players choose whether to play again or not, the program will print out the total number of games played.

Version 7

For the seventh version, I added bubble sort, linear search, and selection sort. I added a type of suit for the player to know their specific cards. For instance, they can get 1 of hearts. Additionally, I inserted some more features on the third menu option, which is exiting the game. I added some fun facts after the player commented about the game by utilizing the selection sort.

Version 8

Version 8 is the last version of this online blackjack game. I added new features to the game. Before going to the menu section, the program will ask the user's information, such as name and age. Then, it will store the information and proceed to the menu where the users can choose the options to read the rules, start the game, or exit the game. I also changed the code on the menu function and added a display function. I made sure to check and utilize all the materials we have learned.

Sample Inputs and Outputs

This is the sample inputs and outputs of the program's latest version:

```
    ||===========================||
    ||   Welcome to Blackjack!   ||
    ||  Let's play and have fun! ||
    ||===========================||
Enter the name of player:
lisa
Enter your age:
18
Player's Information
Player's name: lisa
Player's age: 18

Menu Settings
~~~~~~~~~~~~~~~~~~~~

Choose one of the options:
    1. Rules- Rules of the Game
    2. Start- Start the Game
    3. Exit - Exit the Game

Enter your choice: 2

Before we start, let's place a bet! Do you want to let the computer place a random value of bet? (Y/N)
y
Your bet is $22.00
Player's Card 1 = 7 of Clubs
Player's Card 2 = 11 of Diamond
Player's Total = 18

Would you like to hit or stay? (H/S)
s
```

Users input the name and age, then choose the options for the menu, in this case is 2. The game starts by asking the user if they want the computer to bet for them or not. Then the game starts and asks if the user wants to hit or stay.

```
Dealer's Card = 9 of Clubs
Dealer's Total = 9

Dealer's new card = 11 of Clubs
Dealer's Total = 20

Dealer wins, you lose!


Do you want to play again? (Y/N)
y

Total number of games played: 1

Before we start, let's place a bet! Do you want to let the computer place a random value of bet? (Y/N)
n
Place your bet! (Remember: bet must be less than $500)
77
Your bet is $77.00
Player's Card 1 = 2 of Spade
Player's Card 2 = 5 of Heart
Player's Total = 7

Would you like to hit or stay? (H/S)
h

Player's new card = 4 of Diamond
Player's Total = 11

Player does not have a card with value of 21
I suggest to keep on going!
```

After game is finished, the program asks the user if they want to play again or not. Total number of games is printed.

```
Would you like to hit or stay? (H/S)
h

Player's new card = 4 of Diamond
Player's Total = 15

Player does not have a card with value of 21
I suggest to keep on going!

Would you like to hit or stay? (H/S)
s

Dealer's Card = 1 of Diamond
Dealer's Total = 1

Dealer's new card = 6 of Clubs
Dealer's Total = 7

Congratulations, you win!
You got your bet of $ 77.00 back

Do you want to play again? (Y/N)
n

Total number of games played: 2
```

Flowchart

To see the flowchart please refer to the file attached.

Reference

1. Dr. Mark Lehr's lectures and lab
2. Gaddis, T. (2017). Starting Out With C++ From Control Structures To Objects (9th Ed.). Pearson.

Pseudocode

```
Insert all the system libraries: iostream, iomanip, cstdlib, ctime, fstream, string, cmath, vector
Declare global const int CHOICE = 3
Function prototypes: intro, menu, display, getN, opt1, opt2, opt3, def, checkBj, isBust, bubSort,
cdType, selSort, linSch

Begin main function with parameters int argc, char** argv
Set random number seed using srand function with time(0) as argument
    Opening of the game: intro()
    Declare plyName[1] as string
    Declare plyAge[1] as integer
    Declare inp as integer
    Declare option as integer
    Get player information:
        For i from 0 to 0
            Display "Enter the name of player:"
            Read plyName[i]

            Display "Enter your age:"
            Read plyAge[i]
        End for
    Display player information:
        Display "Player's Information"
        For i from 0 to 0
            Display "Player's name:", plyName[i]
            Display "Player's age:", plyAge[i]
        End for
    Menu loop:
        Do
            menu(option)
            Switch option
                Case 1: opt1(), break
                Case 2: opt2(), break
                Case 3: opt3(), break
                Default: break
            End switch
        While inp > 0 && inp < 3
    Exit stage right
    Return 0
End function

Function intro()
    Display Output for Intro:
        Display "||==================||"
        Display "||   Welcome to Blackjack!   ||"
        Display "||  Let's play and have fun!   ||"
        Display "||==================||"
End function

Function menu(option)
    Declare vldInp as boolean
    Declare defVal as integer

    While not vldInp
        display()
        Display “Enter your choice”
        If option is not 0
            Display option
        End if
        Display “:”
        Option = getN(defVal)
        If option < 1 or option > 3
            Display “Invalid input. Please enter a number between 1 and 3.”
        Else
            vldInp = true
        End if
    End while
End function

Function display()
    Declare menuOpt as array of string with size CHOICE by 2
    menuOpt = {{“1. Rules”, “Rules of the Game”},
            {“2. Start  ”, “Start of the Game”};
            {“3. Exit   ”, “Exit the Game”}}
    Display newline
    Display “Menu Settings”
    Display “~~~~~~~~~~~~~~~~~~~~”
    Display newline
    Display “Choose one of the options:”
    For i = 0 to CHOICE - 1
        Display with width 12
        If i is equal to CHOICE - 1
            Display with width 12
        End if
        Display menuOpt[i][0] + “- ” + menuOpt[i][1]
    End for
End function

Function getN(defVal = 0)
    defVal = 0
    Declare inp as integer
    Read inp from input
    Return (inp is not equal to 0) ? inp : defVal
End function

Function opt1()
    Display newline
    Display “Let’s get to know the game better!”
    Display “1) The maximum value to bet is $500.”
    Display “2) The dealer draws two cards from the deck to each player.”
    Display “3) The dealer will let one of their cards facing up, the player will have to let two
    cards facing up the whole time.”
    Display “4) This game will not have any Jacks, Queens, and Kings. All number cards are
    worth its own value.”
    Display “5) The player that has a total value closest or equal to 21 wins.”
End function

Function isBust(handTot, limit)
    If handTot > limit then
        Display “Bust! You lose, better luck next time!”
        Return true
    End if
```

```
    Return false
End function

Function isBust(plyTot)
    Return isBust(plyTot, 21)
End function

Function linSch(arr, size, key)
    For i = 0 to size - 1
        If arr[i] is equal to key then
            Return true
        End if
    End for
    Return false
End function

Function bubSort(arr)
    n = size of arr
    For i = 0 to n - 2
        For j = 0 to n - i - 2
            If arr[j] > arr[j + 1] then
            Temp = arr[j]
            arr[j] = arr[j + 1]
            Arr[j + 1] = temp
            End if
        End for
    End for
End function

Function cdType(type)
    Switch type
        Case 1: display “Spade”, break
        Case 2: display “Heart”, break
        Case 3: display “Clubs”, break
        Case 4: display “Diamond”, break
        Default: break
    End switch
End function

Function opt2()
```

```
Declare ans, cards, getCard, plyTot, dealTot as integers
Declare plyBet as float
Declare choice, ansBet, answer as characters
Declare cont as boolean
Declare static gameCnt as integer
Cont = true
gameCnt = 0
Do
    plyTot = 0
    dealTot = 0
    Declare plyHand as empty array of integers
    Display “Before we start, let’s place a bet! Do you want to let the computer place
    a random value of bet? (Y/N)”
    Read ansBet
    If ansBet is ‘Y’ or ansBet is ‘y’ then
        Open input file “out.dat” for reading
        Read rows, cols from the file
        Set randRow to random integer between 1 and rows
        Set randCol to random integer between 1 and cols
        For i = 1 to rows
            For j = 1 to cols
                Read plyBet from the file
                If i is equal to randRow and j is equal to randCol then
                    Display “Your bet is $” + plyBet
                End if
            End for
        End for
        Close the file
    Else if ansBet is ‘N’ or ansBet is ‘n’ then
        Display “Place your bet! (Remember: bet must be less than $500)”
        Read plyBet
        If plyBet > 500 then
            Display “Don’t bet too much! You’ve exceeded the limit”
        Else
            Set plyBet to absolute value of plyBet
        End if
        Display “Your bet is $” + plyBet
    End if
    For i = 1 to 2
        Set cards to random integer between 1 and NCARD
        Set getCard to (cards % 11) + 1
        Append getCard to plyHand
        Display “Player’s Card “ + i + “ = “ + getCard + “ of “
        Call cdType with random integer between 1 and 4
End for
Call bubSort with plyHand
For each card in plyHand
        Increment plyTot by card
End for
Display “Player’s Total = “ + plyTot
While cont and (dealTot <= 21 or plyTot <= 21 or dealTot is not 21 or plyTot is
not 21)
        Display “Would you like to hit or stay? (H/S)“
        Read choice
        If choice is ‘H’ or choice is ‘h’ then
                Set cards to random integer between 1 and NCARD
                Set getCard to (cards % 11) + 1
                Append getCrd to plyHand
                Display “Player’s new card = “ + getCard + “ of ”
                Call cdType with random integer between 1 and 4
                Increment plyTot by getCard
                Display “Player’s Total = “ + plyTot
                If isBust(plyTot) then
                        Return
                End if
                Declare schVal, pCard as integers
                Set schVal to 21
                Set pCard to empty array
                Append 2 to pCard
                Set numPC to 2
                Set found to linSch(pCard, numPC, schVal)
                If found then
                        Display “Player has a card with value of “ + schVal
                Else
                        Display “Player does not have a card with value of “ +
                        schVal
                        Display “I suggest to keep on going!”
                End if
        Else if choice is ‘S’ or choice is ‘s’ then
                Set cards to random integer between 1 and NCARD
```

```
                    Set getCard to (cards % 11) + 1
                    Increment dealTot by getCard
                    Display “Dealer’s Card = “ + getCard + “ of “
                    Call cdType with random integer between 1 and 4
                    Display “Dealer’s Total = “ + dealTot
                    If dealTot < 17 and dealTot is not 21 then
                         Set cards to 0
                         Repeat
                              Set cards to random integer between 1 and NCARD
                              Set getCard to (cards % 11) + 1
                         Until cards < dealTot
                         Increment dealTot by getCard
                         Display “Dealer’s new card = “ + getCard + “ of ”
                         Call cdType with random integer between 1 and 4
                         Display “Dealer’s Total = “ + dealTot
                    Else if dealTot > 21 then
                         Display “Dealer busts, you win!”
                         Display “You got your bet of $” + plyBet + “ back”
                    End if
                    If dealTot <= 21 and plyTot <= 21 then
                         If dealTot > plyTot then
                              Display “Dealer wins, you lose!”
                              Cont = false
                         Else if dealTot < plyTot then
                              Display “Congratulations, you win!”
                              Display “You got your bet of $” + plyBet + “ back”
                              Cont = false
                         Else
                              Display “It’s a tie!”
                              Cont = false
                         End if
                    End if
          End if
End while
If checkBj(plyTot) then
          Clear plyHand
          Continue
End if
Display “Do you want to play again? (Y/N)”
Read answr
```

```
        If answr is ‘N’ or answr is ‘n’ then
            Cont = false;
        Else if answr is ‘Y’ or answr is ‘y’ then
            Cont = true;
        End if
        Increment gameCnt by 1
        Display “Total number of games played: “ + gameCnt
    End do
    Return
End function

Function selSort(arr, size)
    For i = 0 to size - 1
        minIndx = i
        For j = i + 1 to size
            If arr[j] < arr[minIndx] then
                minIndx = j
            End if
        End for
        If minIndx is not equal to i then
            Temp = arr[i]
            Arr[i] = arr[minIndx]
            arr[minIndx] = temp
        End if
    End for
End function

Function opt3()
    Declare cmnt as string
    Display “Thanks for playing, see you next time!”
    Display “Please leave a comment for us to improve in the future!”
    Read cmnt
    cmnSize = length of cmnt
    Declare cmnChar as array of integers with size cmnSize
    For i = 0 to cmnSize - 1
        cmnChar[i] = ASCII value of cmnt[i]
    End for
    Call selSort with cmnChar and cmnSize
    Display “Thank you for your feedback!”
    Display “Here is some fun fact we can give you!”
    Display “The sorted comment (in characters) for your first word is ”
    For i = 0 to cmnSize - 1
        Display character representation of cmnChar[i]
    End for
    Return
End function

Function def(ans)
    Display “Exiting game.”
    Terminate the program

Function checkBj(plyTot)
    If plyTot is equal to 21 then
        Display “Blackjack! You win!”
        Return true
    Else
        Return false
    End if
End function

Exit the program
```

Code

```
/*
 * File:   main.cpp
 * Author: Aurelisa J. Sindhunirmala
 * Created on June 2, 2023, 10:18 AM
 * Purpose: Project 2; Create a blackjack game Version 8
 */

//System Libraries
#include <iostream>     //Input-Output Library
#include <iomanip>      //Format Library
#include <cstdlib>      //Random functions
#include <ctime>        //Set Random Library
#include <fstream>      //File Stream Library
#include <string>       //String Library
#include <vector>       //Vector Library
#include <cmath>        //Math Library
using namespace std;
```

```
//User Libraries

//Global Constants - Math/Physics/Chemistry/Conversions ONLY!!!!
const int CHOICE = 3;


//Function Prototypes
void intro();                  //Opening of the game
void menu(int&);               //Menu settings of the game
void display();                //2 dimensioned arrays
int  getN(int);                //Getting the player's answers
void opt1();                   //Menu Option 1
void opt2();                   //Menu Option 2
void opt3();                   //Menu Option 3
void def(int);                 //Default switch case for Menu
bool checkBj(int);             //Check Blackjack
bool isBust(int&);             //Check if player's hand exceeds 21 (bust)
bool isBust(int&, int);        //Check if a specific hand exceeds a given limit
void bubSort(vector<int>&);    //Sort the player's hand with bubble sort
void cdType(int);              //Get the card type: Suits in Deck
void selSort(int[], int);      //Sort the player's comment
bool linSch(int[], int, int);  //Linear search to see if sum of player's card = 21

//Execution Begins HERE!!!
int main(int argc, char** argv) {
   //Set Random Number Seed
   srand(static_cast<unsigned int>(time(0)));

   //Map Input to Output - Process
   //Opening of the game
   intro();

   //Declare an array to store player names
   string plyName[1];
   int plyAge[1],
       inp,
       option;

   //Get player information
```

```
    for (int i = 0; i < 1; i++){
        cout << "Enter the name of player:" << endl;
        cin >> plyName[i];

        cout << "Enter your age: " << endl;
        cin >> plyAge[i];
    }


    //Display player information
    cout << "Player's Information " << endl;
    for (int i = 0; i < 1; i++){
        cout << "Player's name: " << plyName[i] << endl;
        cout << "Player's age: " << plyAge[i] << endl;
    }

    do{ //switch case for rules, start, or exit game
        menu(option);
        switch(option){
            case 1: opt1(); break;
            case 2: opt2(); break;
            case 3: opt3(); break;
            default: break;
        }
    } while (inp > 0 && inp < 3);

    //Exit stage right
    return 0;
}

void intro(){  //Opening of the game
    //Display Output for Intro
    cout << setw(35) << "||===========================||" << endl;
    cout << setw(35) << "||   Welcome to Blackjack!   ||" << endl;
    cout << setw(35) << "||  Let's play and have fun! ||" << endl;
    cout << setw(35) << "||===========================||" << endl;
}

void menu(int& option){ //Menu settings of the game
    //Declare variables
    bool vldInp = false;
    int defVal;

    while (!vldInp){ //User enter choice and program reads
        display();
        cout << endl << "Enter your choice";
        if (option != 0){
            cout << option;
        }
        cout << ": ";

        option = getN(defVal);

        if (option < 1 || option > 3){
            cout << "Invalid input. Please enter a number between 1 and 3." << endl << endl;
        } else{
            vldInp = true;
        }
    }
}

void display(){ //2 dimensioned arrays

    string menuOpt[CHOICE][2] = {
        {"1. Rules", "Rules of the Game"},
        {"2. Start", "Start the Game"},
        {"3. Exit ", "Exit the Game"}
    };
    cout << endl << "Menu Settings" << endl;
    cout << "~~~~~~~~~~~~~~~~~~~~" << endl;
    cout <<endl << "Choose one of the options:" << endl;

    for (int i = 0; i < CHOICE; i++){
        cout << setw(12);
        if (i == CHOICE - 1){
            cout << setw(12);
        }
        cout << menuOpt[i][0] << "- " << menuOpt[i][1] << endl;
    }
}
```

```
int getN(int defVal = 0){ //Getting the player's answers

    int inp;
    cin >> inp;
    return (inp != 0) ? inp : defVal;
}

void opt1(){ //Print out rules of the game
    //Rules
    cout << "\nLet's get to know the game better!\n";
    cout << "1) The maximum value to bet is $500." << endl;
    cout << "2) The dealer draws two cards from the deck to each player." << endl;
    cout << "3) The dealer will let one of their card facing up,"
                " player will have to let two cards facing up the whole time." << endl;
    cout << "4) This game will not have any Jacks, Queens, and Kings."
            " All number cards are worth its own value." << endl;
    cout << "5) The player that has a total value closest or equal to 21 wins." << endl;
}

bool isBust(int& handTot, int limit){ //Read if the player busts = 21
    if(handTot > limit){
        cout << "Bust! You lose, better luck next time!" << endl << endl;
        return true;
    }
    return false;
}

bool isBust(int& plyTot){
    return isBust(plyTot, 21);
}

bool linSch(int arr[], int size, int key){ //linear search to see if player's sum card = 21
    for (int i = 0; i < size; i++){
        if (arr[i] == key){
            return true;
        }
    }
    return false;
}
```

```
void bubSort(vector<int>& arr){
    int n = arr.size();
    for (int i = 0; i < n - 1; i++){
        for (int j = 0; j < n - i - 1; j++){
            if (arr[j] > arr[j + 1]){
                //Swap the elements
                int temp = arr[j];
                arr[j] = arr[j + 1];
                arr[j + 1] = temp;
            }
        }
    }
}

void cdType(int type){ //suits in the card deck
    switch(type){
        case 1: cout << "Spade"; break;
        case 2: cout << "Heart"; break;
        case 3: cout << "Clubs"; break;
        case 4: cout << "Diamond"; break;
        default: break;
    }
}

void opt2(){
    //Declare variables
    int ans,                //player's input toward the menu settings
        cards,              //range of the total cards in a deck
        getCard,              //get card
        plyTot,             //Sum of player's cards worth
        dealTot;             //Sum of dealer's cards worth

    float plyBet;           //Player's bet

    char choice,            //Player's choice, H/S
         ansBet,           //Player's choice, Y/N to bet
         answr;            //Player's choice if want to continue game or exit game

    bool cont;              //Continuation of game
    static int gameCnt;     //Count the number of games played

    //Initialize variable
    cont = true; // Control the game loop
    gameCnt = 0; // Game count

    do{
      //Reset variables for a new game
      plyTot = 0;
      dealTot = 0;
      vector<int> plyHand; //Player's hand

      //Player's option to bet alone or let the computer bet for the player
      cout << endl << "Before we start, let's place a bet! Do you want to let the computer place a
random value of bet? (Y/N)" << endl;
      cin >> ansBet;

      if (ansBet == 'Y' || ansBet == 'y'){
        //computer randomly pick number from file input
        //Open input file
        fstream in;
        int rows, cols;

        in.open("out.dat",ios::in);

        //Read the number of rows and columns from the input file
        in >> rows >> cols;

        //Generate random row and column numbers within the bounds of the input file
        int randRow = rand() % rows + 1;
        int randCol = rand() % cols + 1;

        //Loop through each number in the input file and find the one corresponding to the
randomly selected row and column
        for (int i = 1; i <= rows; i++){
          for (int j = 1; j <= cols; j++){
            in >> plyBet;
            if (i == randRow && j == randCol){
              //Display the randomly selected number to the user as their bet
```

```
                cout << fixed << setprecision(2) << showpoint;
                cout << "Your bet is $" << plyBet << endl;
            }
        }
    }
    in.close();

//When the user choose to bet on their own
} else if (ansBet == 'N' || ansBet == 'n'){
    cout << "Place your bet! (Remember: bet must be less than $500)" << endl;
    cin >> plyBet;

    //Player's bet cannot be more than 500
    if (plyBet > 500){
        cout << "Don't bet too much! You've exceeded the limit" << endl;
    } else { //Cannot be negative
        plyBet = abs(plyBet);
    }
    //Print the player's bet
    cout << fixed << setprecision(2) << showpoint;
    cout << "Your bet is $" << plyBet << endl;
}

//Cards for player
for (int i = 1; i <= 2; i++){
    cards = rand() % 52 + 1;     //set the range to [1,52]
    getCard = cards % 11 + 1;       //set the range of value each card is worth [1,11]
    plyHand.push_back(getCard);
    cout << "Player's Card " << i <<" = " << getCard << " of ";
    cdType(rand() % 4 + 1);
    cout << endl;
}

//Sort the player's hand using bubble sort
bubSort(plyHand);

//Calculate the player's total
for (int i = 0; i < plyHand.size(); i++){
    plyTot += plyHand[i];
}
```

```
cout << "Player's Total = " << plyTot << endl << endl;

//Player make choice
while (cont && (dealTot <= 21 || plyTot <= 21 || dealTot != 21 || plyTot != 21)){
    cout << "Would you like to hit or stay? (H/S)" << endl;
    cin >> choice;
    cout << endl;

    //If player choose hit
    if (choice == 'H' || choice == 'h'){
        cards = rand() % 52 + 1;
        getCard = cards % 11 + 1;
        plyHand.push_back(getCard);
        cout << "Player's new card = " << getCard << " of ";
        cdType(rand() % 4 + 1);
        cout << endl;
        plyTot += getCard;
        cout << "Player's Total = " << plyTot << endl << endl;

        //if plyTot > 21
        if (isBust (plyTot)){
            return;
        }

        //Linear search for a specific card value
        int schVal = 21;
        int pCard[2],
            numPC = 2;
        bool found = linSch(pCard, numPC, schVal);
        if (found){
            cout << endl << "Player has a card with value of " << schVal << endl;
        } else{
            cout << "Player does not have a card with value of " << schVal << endl;
            cout << "I suggest to keep on going!" << endl << endl;
        }
    } else if (choice == 'S' || choice == 's'){
        //Dealer's turn to getCard
        cards = rand() % 52 + 1;
        getCard = cards % 11 + 1;
        dealTot += getCard;
        cout << "Dealer's Card = " << getCard << " of ";
        cdType(rand() % 4 + 1);
        cout << endl;
        cout << "Dealer's Total = " << dealTot << endl << endl;

        //If dealer's total value is more than 17, dealer will not take another card
        if (dealTot < 17 && dealTot != 21){
            for (int cards = 0; cards < dealTot; cards++){
                cards = rand() % 52 + 1;
                getCard = cards % 11 + 1;
            }
            dealTot += getCard;
            cout << "Dealer's new card = " << getCard << " of ";
            cdType(rand() % 4 + 1);
            cout << endl;
            cout << "Dealer's Total = " << dealTot << endl << endl;
        } else if (dealTot > 21){
            cout << "Dealer busts, you win!" << endl;
            cout << "You got your bet of $ " << plyBet << " back" << endl;
        }

        //If dealer's and user's total value is less than or equal to 21
        if (dealTot <= 21 && plyTot <= 21){
            if (dealTot > plyTot){
                cout << "Dealer wins, you lose!" << endl << endl;
                cont = false;
            } else if (dealTot < plyTot){
                cout << "Congratulations, you win!" << endl;
                cout << "You got your bet of $ " << plyBet << " back" << endl;
                cont = false;
            } else{
            cout << "It's a tie!" << endl << endl;
            cont = false;
            }
        }
    }
}

//Check for blackjack for Player
if (checkBj(plyTot)){
```

```
            plyHand.clear();    //Reset to player's hand
            return;             //Continue to the next iteration of the do-while loop
        }

        //Check if the player wants to continue playing
        cout << "\nDo you want to play again? (Y/N)" << endl;
        cin >> answr;

        if (answr == 'N' || answr == 'n'){
            cont = false; //Exit the game loop if player chooses N
        } else if (answr == 'Y' || answr == 'y'){
            cont = true;  //Continue the game loop if player chooses Y
        }
        //Increment the game count after each game
        gameCnt++;
        cout << endl << "Total number of games played: " << gameCnt << endl;
    } while (cont);

    //Return to the main menu
    return;
}

void selSort(int arr[], int size){ //selection sort to rearrange the comments alphabetically
    for (int i = 0; i < size - 1; i++){
        int minIndx = i;
        for (int j = i + 1; j < size; j++){
            if (arr[j] < arr[minIndx]){
                minIndx = j;
            }
        }
        if (minIndx != i){
            //Swap the elements
            int temp = arr[i];
            arr[i] = arr[minIndx];
            arr[minIndx] = temp;
        }
    }
}

void opt3(){ //Print out the outro
```

```
    //Declare variable
    string cmnt;    //Player's comment about the game

    //Exit
    cout << endl << "Thanks for playing, see you next time!" << endl;

    //User's comment about the game
    cout << "Please leave a comment for us to improve in the future!" << endl;
    cin >> cmnt;

    //Sort the comment characters in ascending order
    int cmnSize = cmnt.size();
    int cmnChar[cmnSize];
    for (int i = 0; i < cmnSize; i++){
        cmnChar[i] = cmnt[i];
    }
    selSort(cmnChar, cmnSize);

    //Display the sorted comment characters
    cout << "\nThank you for your feedback!" << endl;
    cout << "Here is some fun fact we can give you!" << endl;
    cout << "The sorted comment (in characters) for your first word is ";
    for (int i = 0; i < cmnSize; i++){
        cout << static_cast<char>(cmnChar[i]);
    }
    cout << endl;

    return;
}

void def(int ans){ //Default switch case for menu
    cout << endl << "Exiting game." << endl;
    exit(0); //Terminate the program and return control to the operating system
}

bool checkBj(int plyTot){ //Check blackjack
    return (plyTot == 21) ? (cout << "Blackjack! You win!\n", true) : false;
}
```