

Isaac Kang | CSC 5 - 45276| July 18, 2016

Project 1

game of black jack

Contents

[**Introduction: 1**](#_Toc457530036)

[**The Game: 2**](#_Toc457530037)

[**Rules 2**](#_Toc457530038)

[**Summary: 2**](#_Toc457530039)

[**Description: 2**](#_Toc457530040)

[**Constructs and Concepts Used 3**](#_Toc457530041)

[**Iostream Library 3**](#_Toc457530042)

[**Cstdlib Library 3**](#_Toc457530043)

[**Limits Library 3**](#_Toc457530044)

[**Fstream Library 3**](#_Toc457530045)

[**Data Types 3**](#_Toc457530046)

[**Conditional Statements 3**](#_Toc457530047)

[**Loops 3**](#_Toc457530048)

[**Flowchart: 4**](#_Toc457530049)

[**Pseudo Code: 20**](#_Toc457530050)

[**Program: 20**](#_Toc457530051)

# Introduction:

Blackjack (also called "21") is a fun and enjoyable casino card game. The basic rules of blackjack are quite easy to learn. As the name 21 implies, the object of the game is to come as close to 21 points as possible with your cards -- without going over -- and having a higher total than the dealer. The game begins with the player receiving two cards face up. The dealer also receives two cards. The player wins if the total value of his cards is closer to 21 than the dealer's cards. Another way to win is if the dealer goes over 21, while the player does not. The player loses if his card value is higher than 21.

The game of blackjack is dealt with six decks of cards (to prevent card counting), and after each hand (or a set number of hands), the cards are re-shuffled. The dealer must continue drawing cards if the value of his two cards is 17 points or less, while the player can stop at any time. The Player can split his cards when dealt cards of the same value (for instance, 6-6 or J-Q), doubling his opportunity to play that particular hand. Split aces receive only one additional card.

# The Game:

### Rules

Blackjack is a game where the goal is to hit 21 through a combination of cards or have the dealer bust (over 21). Each number has its own face value (2, 3, 4, 5, etc.…) while a Jack, Queen, King has the value of 10. An ace can be represented as a 1 (hard) or an 11 (soft). The user is dealt two cards and the dealer is dealt two cards. The user can select to hit (get another card) or stay (not get another card). The dealer must hit on a soft 17. For example, if the user has a 6 and A. The value is 17 and it is known as soft (as in the ace is represented by 11). The user can stay or get another card, in which the Ace may turn into the value 1, becoming a hard ace. One the user chooses to stay, the dealer must then continue to hit until it has a hard 17 or higher. If the dealer has a higher number than the user, the user loses their initial bet. Otherwise, the user wins.

# Summary:

Project size: 500+ lines

Number of variables: about 40

The Black Jack game took me about the entire week to and it was difficult in that I had to think of every single different scenario. While creating the program, I implemented most of the topics we learned so far (functions, if/else, loops, etc.). I wasn’t 100 percent sure how to fully use functions until I read different topics on it and got a better understanding.

# Description:

A quick description of the game: The user determines whether or not they want to play the game. If the answer is yes the user is allowed to play the game with a computer where they can make their own decisions. The objective of the game- Each participant attempts to beat the dealer by getting a count as close to 21 as possible, without going over 21.

# Constructs and Concepts Used

### Iostream Library

|  |  |
| --- | --- |
| Name | Frequency |
| Static\_cast  Cout  Cin  Cin.clear()  Cin.ignore() | 1  51  9  3  1 |

### Cstdlib Library

|  |  |
| --- | --- |
| Name | Frequency |
| Srand()  Rand() | 1  1 |

### Limits Library

|  |  |
| --- | --- |
| Name | Frequency |
| Numeric\_limits | 1 |

### Fstream Library

|  |  |
| --- | --- |
| Name | Frequency |
| Out.open()  Out.close()  Ofstream | 1  1  1 |

### Data Types

|  |  |
| --- | --- |
| Name | Frequency |
| Int  Char  Const int | 105  13  26 |

### Conditional Statements

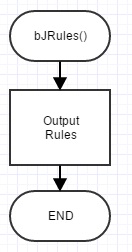
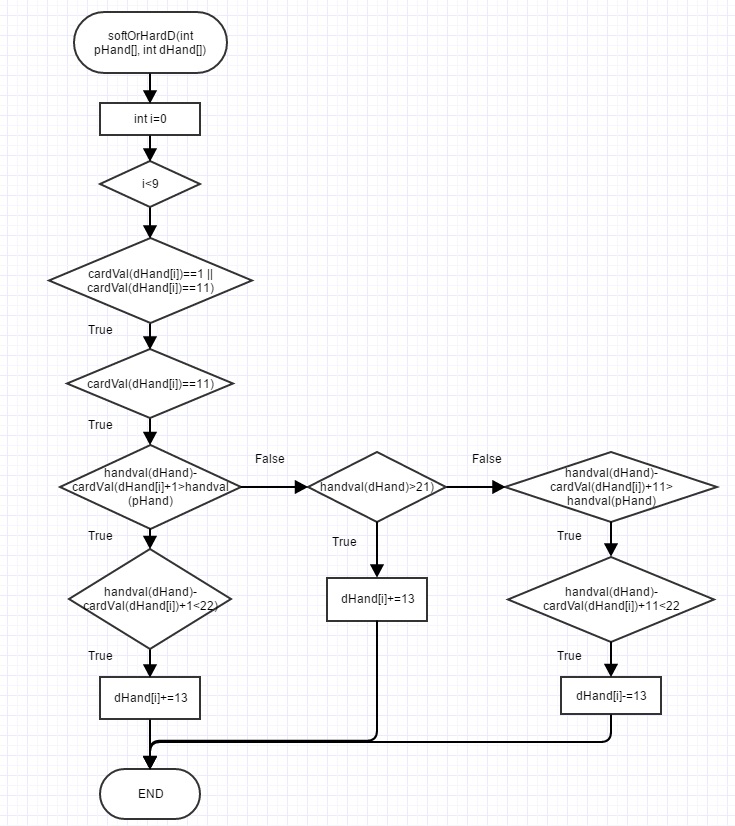
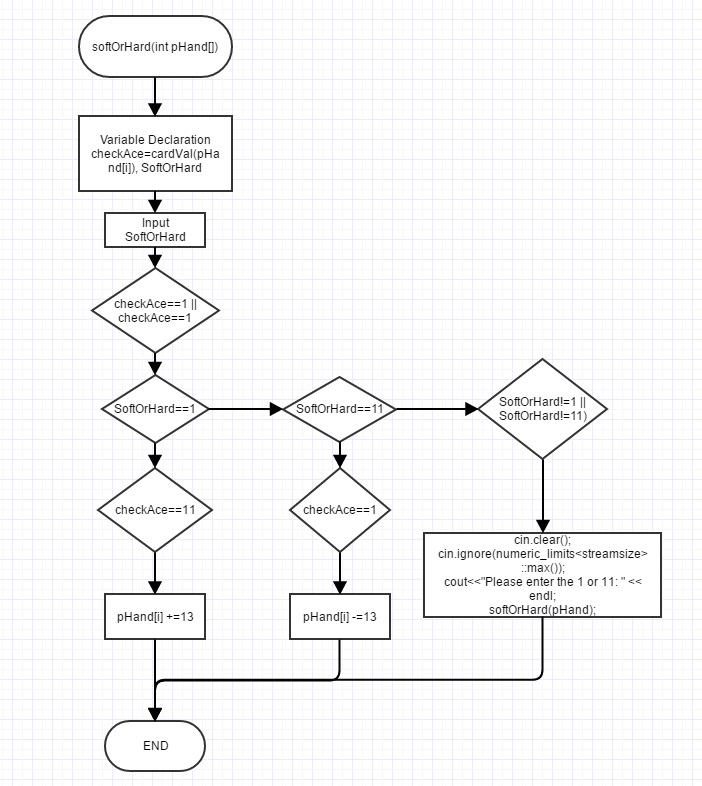
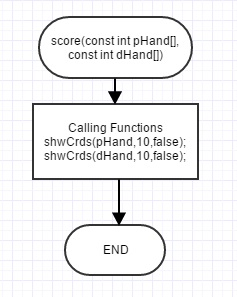
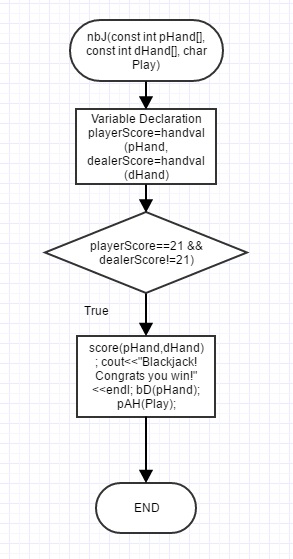
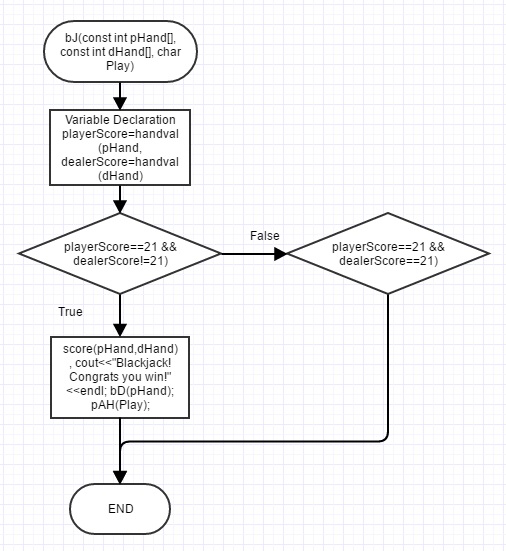
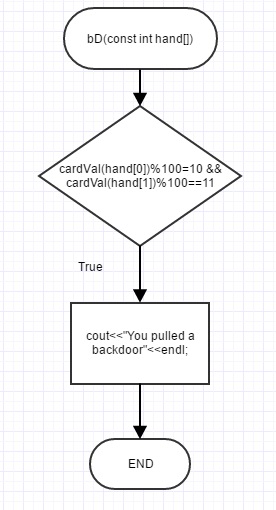
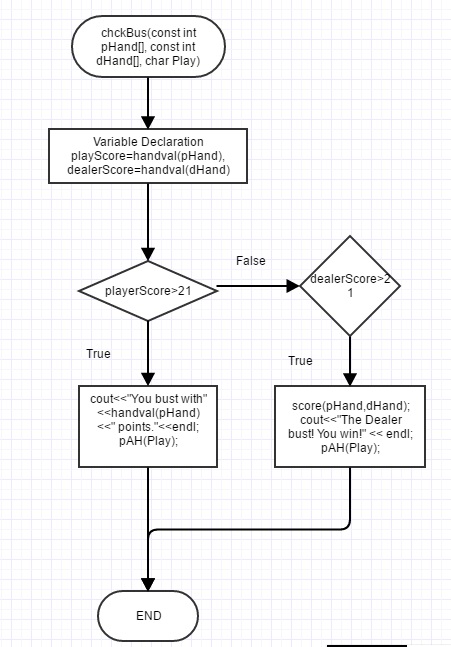
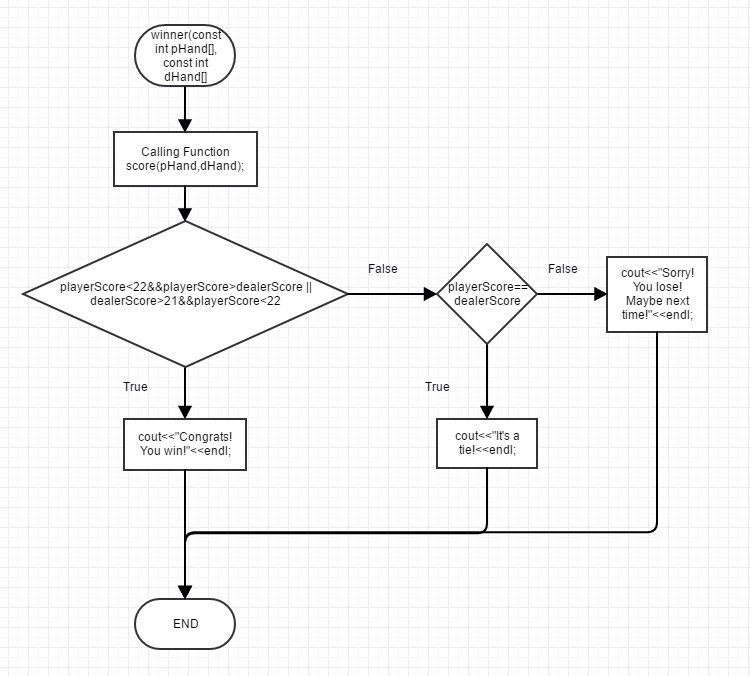
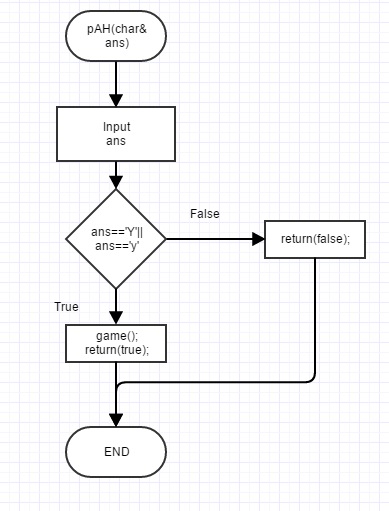
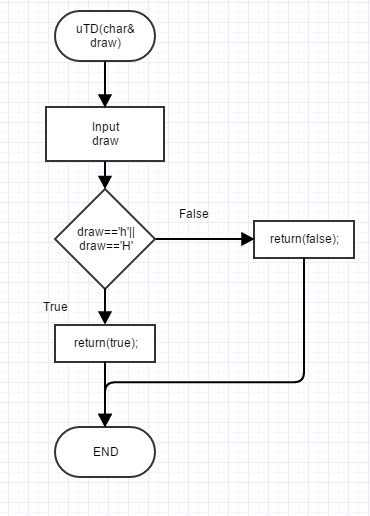
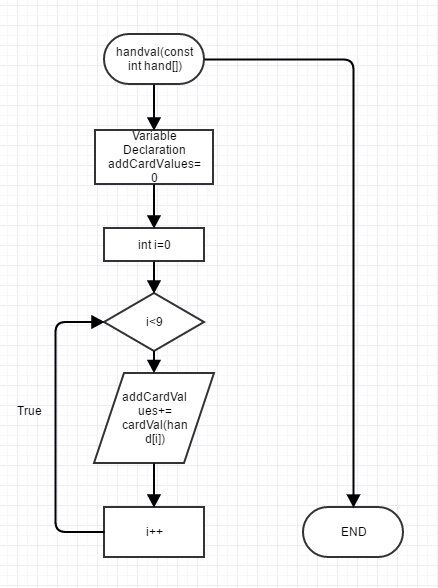
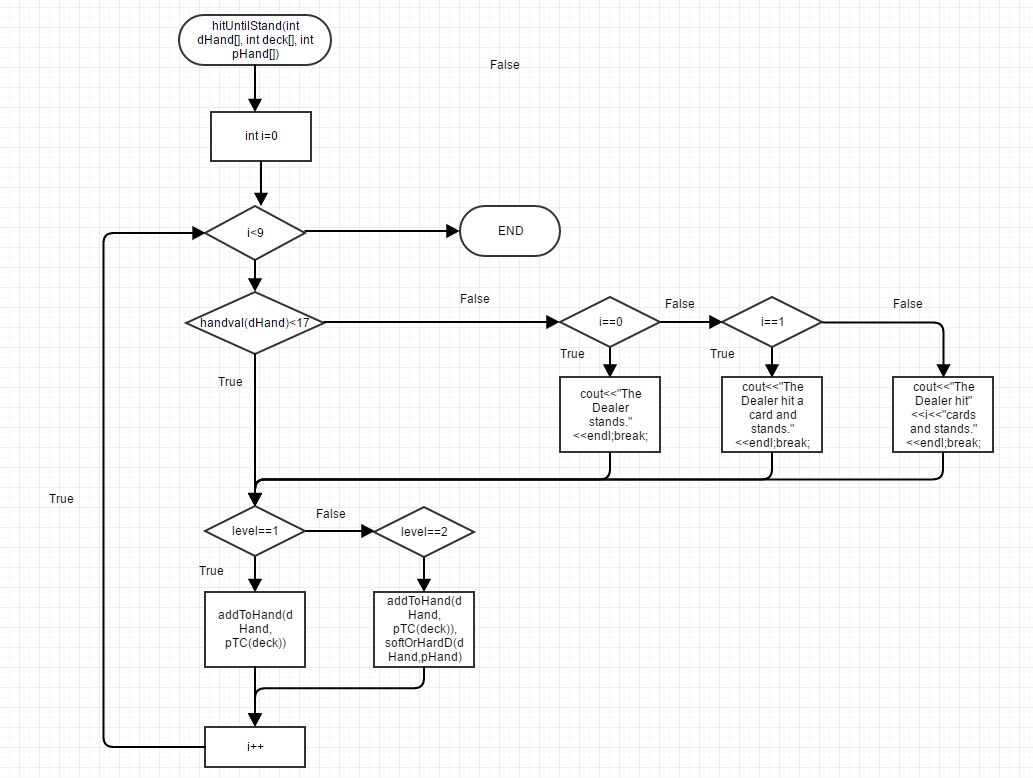
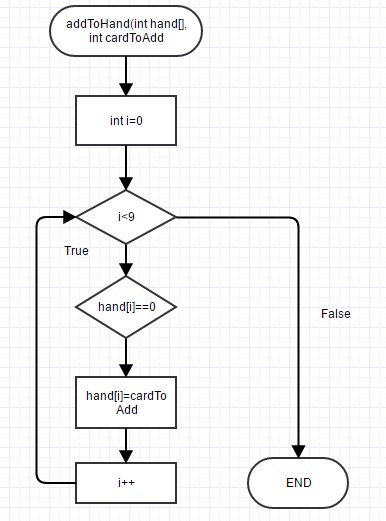
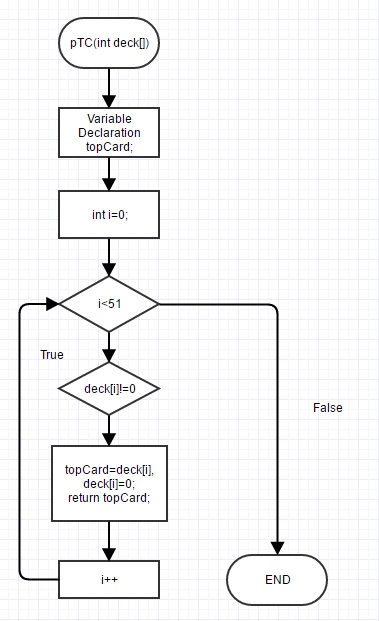
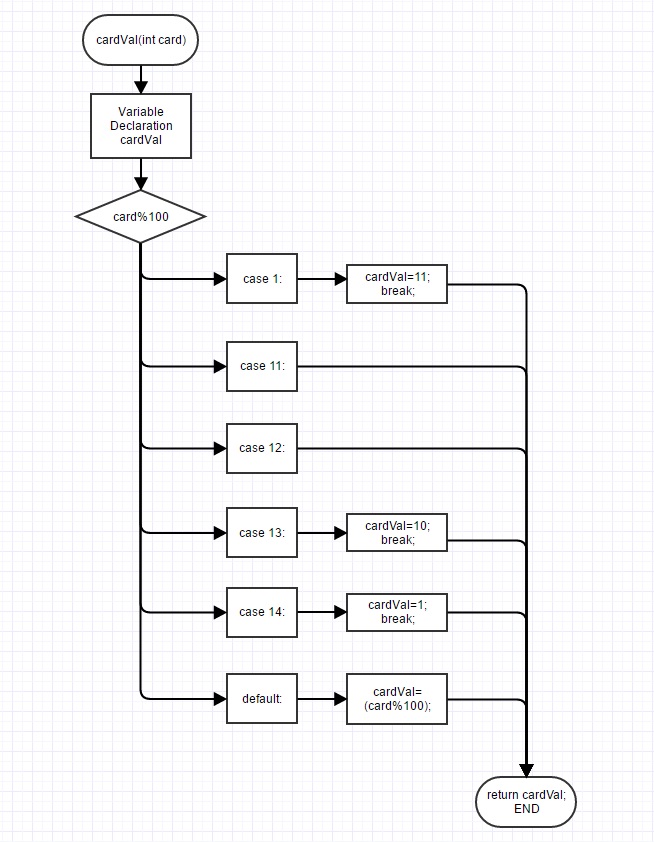
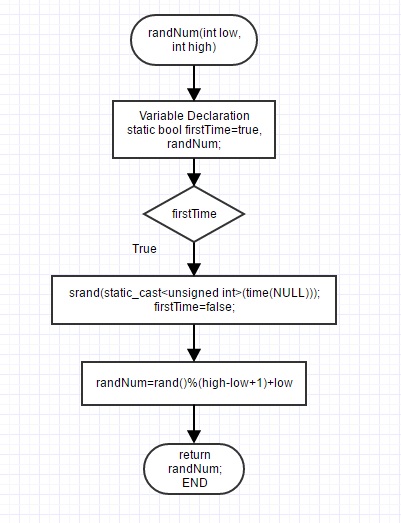
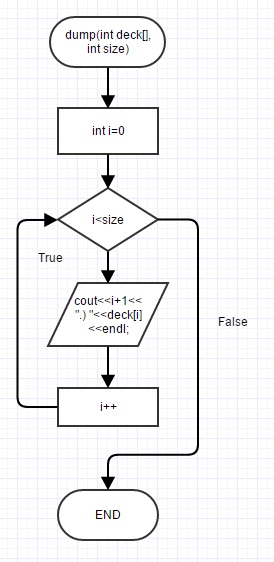
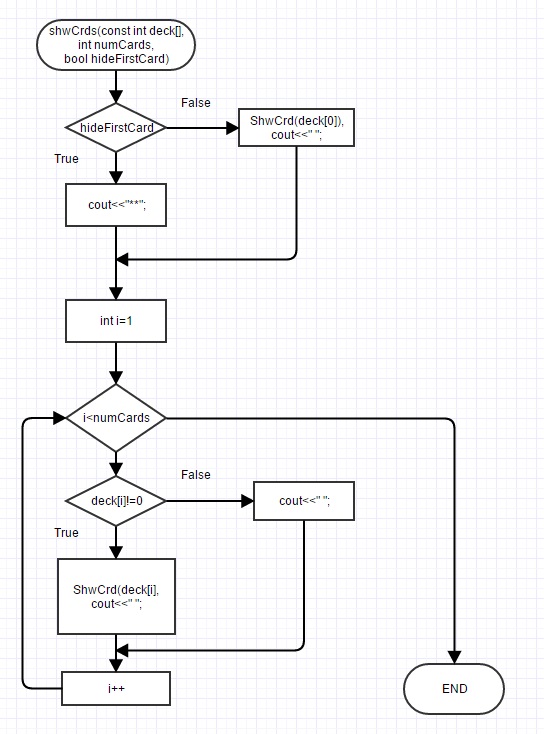
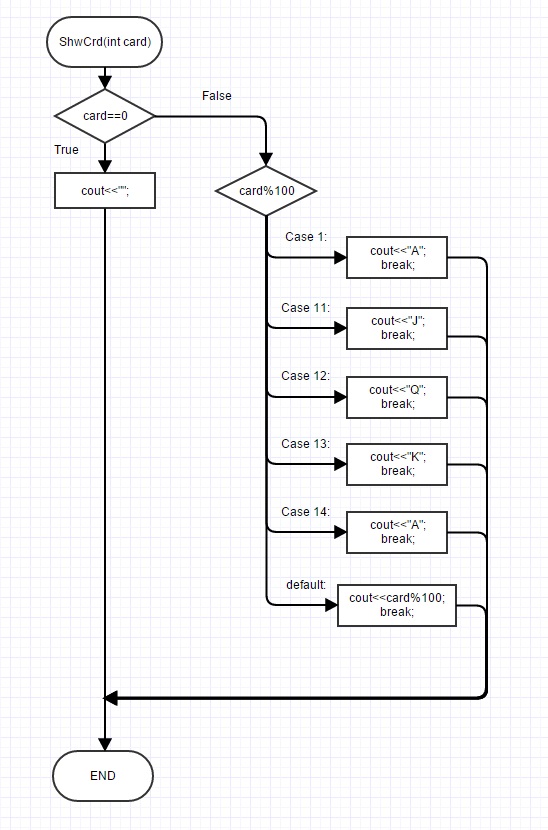
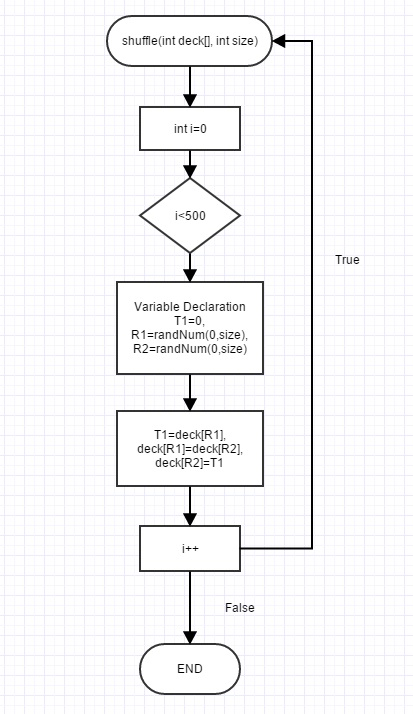
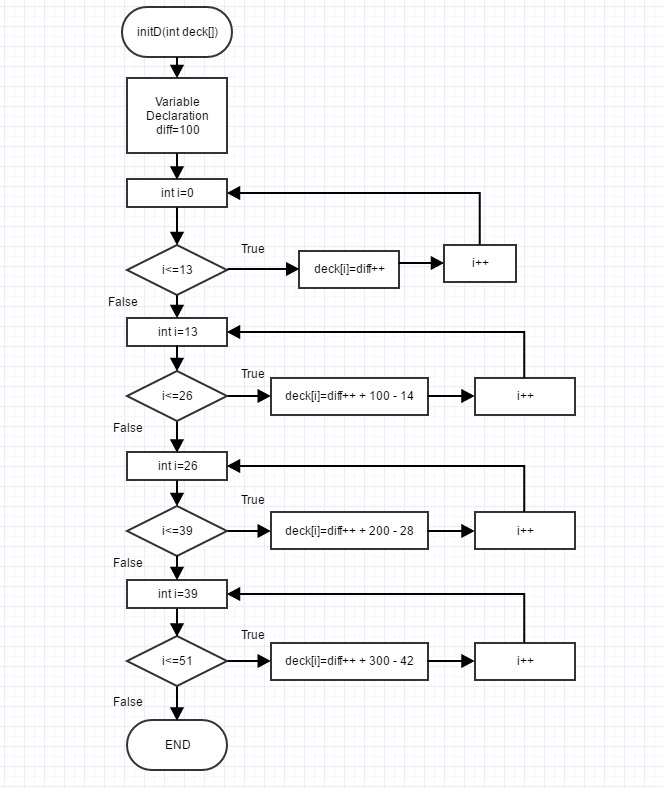
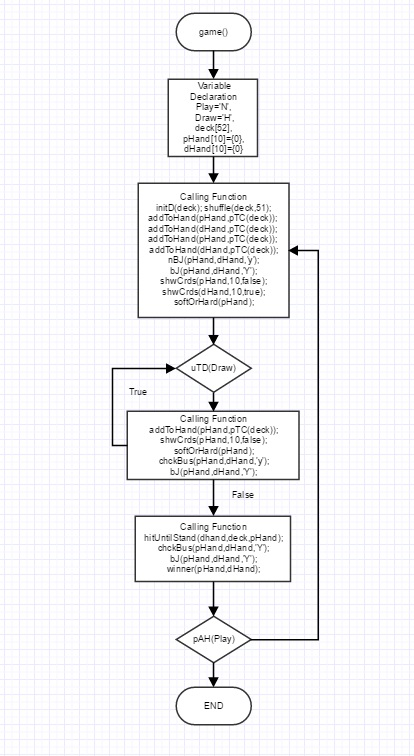
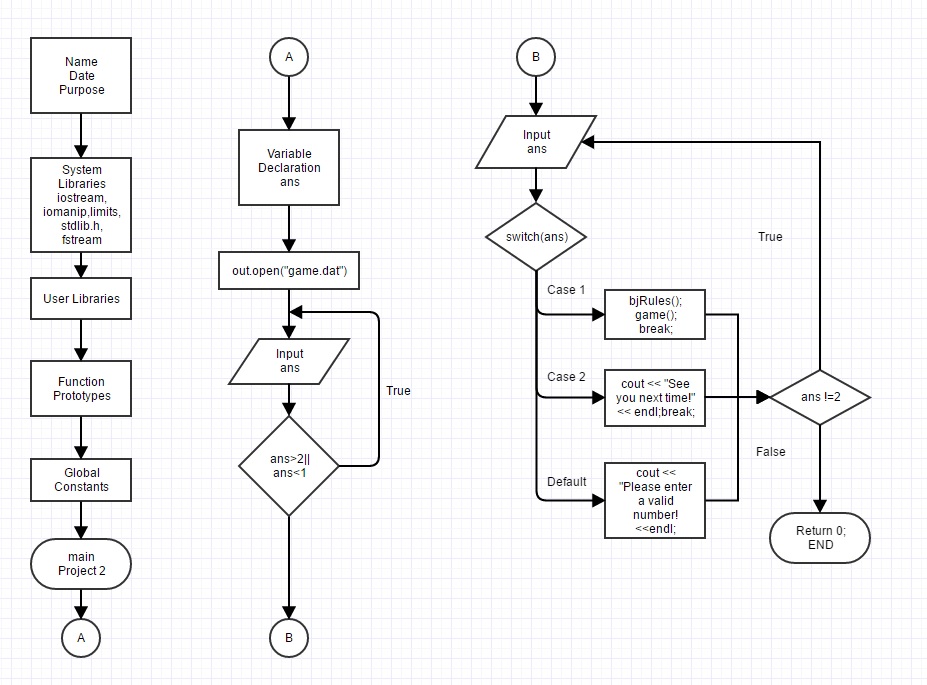
|  |  |
| --- | --- |
| Name | Frequency |
| If  If/else  If/else if  switch | 48  10  7  4 |

### Loops

|  |  |
| --- | --- |
| Name | Frequency |
| For  While  Do-while | 27  4  6 |

# Flowchart:

Flowchart will be spread out.



# Pseudo Code:

**Main-**

*//Put in opening comments*

*//Bring in system libraries*

*//Enter main*

*//Declare variable*

*//Open file that we write data to*

*//Welcome user to Menu-based operation*

*Of Black Jack*

*//Provide user with options*

*//Enter do-while loop*

*//Input variable*

*//Begin switch conditional statement*

*If case 1 play game*

*If case 2 stop game*

*If default, remind user to input correct numbers*

*//End do-while loop*

**Game Function-**

*//Welcome user to the game*

*//Declare variables*

*//Enter do-while loop*

*//Call on functions with respective parameters*

*//Output that Dealer is shuffling deck*

*//Call on functions with respective parameters*

*//Output both player and dealer hands*

*Call functions for both player and dealer*

*//Enter while loop*

*//Call functions with respective parameters*

*//Output current player and dealer hand*

*//End while loop*

*//Call on the rest of the functions*

*//End the do-while loop with condition*

**initD Function-**

//START

//Declare variables

//FOR given iteration bounds

Deck[i]=diff++;

//ENDFOR

//FOR given iteration bounds

Deck[i]=diff++ +100-14;

//ENDFOR

//FOR given iteration bounds

Deck[i]=diff++ +200-28;

//ENDFOR

//FOR given iteration bounds

Deck[i]=diff++ +300-42;

//ENDFOR

//END

**Shuffle Function-**

*//START*

*//FOR given iteration bounds*

*Declare variables*

*T1=deck[R1];*

*Deck[R1]=deck[R2];*

*Deck[R2]=T1;*

*//ENDFOR*

*//END*

**ShwCrd Function-**

*//START*

*//If card is 0*

*Cout<<””;*

*//else*

*//switch*

*Case 1: cout<<”A”;break;*

*Case 11: cout<<”J”;break;*

*Case 12: cout<<”Q”;break;*

*Case 13: cout<<”K”;break;*

*Case 14: cout<<”A”;break;*

*Default: cout<<card%100;*

*//END SWITCH*

*//END IF/ELSE*

*//END*

**shwCrds Function-**

*//START*

*//If hideFirstCard*

*Cout<<”\*\*”;*

*//else*

*ShwCrd(deck[0];*

*Cout<<” “;*

*//FOR given iteration bounds*

*//If deck[i] does not equal to 0*

*ShwCrd(deck[i]);*

*Cout<< “ “;*

*//else*

*Cout<< “ “;*

*//END IF/ELSE*

*//ENDFOR*

*//END*

**dump Function-**

*//START*

*//FOR given iteration bounds*

*Cout<<i+1<<”.) “<<deck[i]<<endl;*

*//ENDFOR*

*//END*

**randNum Function-**

*//START*

*//Declare variables*

*//If firstTime*

*Srand(static\_cast<unsigned int>(time(NULL)));*

*firstTime=false;*

*//END If*

*//STATE*

*randNum=rand()%(high-low+1)+low;*

*//RETURN*

*randNum;*

*//END*

**cardVal Function-**

*//START*

*//Declare variables*

*//BEGIN switch statements*

*case 1: cardVal = 11; break;*

*case 11:*

*case 12:*

*case 13: cardVal = 10; break;*

*case 14:cardVal = 1; break;*

*default: cardVal = (card % 100);*

*//END switch statements*

*//RETURN*

*cardVal;*

*//END*

**pTC Function-**

*//START*

*//Declare variable*

*//FOR given iteration bounds*

*//BEGIN If statement*

*//If deck[i] does not equal to 0*

*topCard=deck[i];*

*deck[i]=0;*

*return topCard;*

*//END If statement*

*//ENDFOR*

*//END*

**addToHand Function-**

*//START*

*//FOR given iteration bounds*

*//BEGIN If statement*

*//If hand[i] is equal to 0*

*Hand[i]=cardToAdd;*

*Break;*

*//END If statement*

*//ENDFOR*

*//END*

**hitUntilStand Function-**

*//START*

*//FOR given iteration bounds*

*//BEGIN If statement*

*//If handval(dHand) is less than 17*

*//If level is equal to 1*

*addToHand(dHand,pTC(deck));*

*//ELSE If level is equal to 2*

*addToHand(dHand,pTC(deck));*

*softOrHardD(dHand,pHand);*

*//END If/else statement*

*//Else*

*//If I is equal to 0*

*Cout<< “The Dealer stands.”<< endl;*

*Break;*

*//Else if I is equal to 1*

*Cout<< “The Dealer hit a card and stands.” << endl;*

*Break;*

*//Else*

*Cout<< “The Dealer hit”<<i<< “ cards and stands.”<<endl;*

*Break;*

*//END IF/ELSE statement*

*//ENDFOR*

*//END*

**handval Funtion-**

*//START*

*//Declare variables*

*//FOR given iteration bounds*

*addCardValues+=cardVal(hand[i]);*

*//RETURN*

*addCardValues;*

*//ENDFOR*

*//END*

**pAH Function-**

*//START*

*//Prompt user to decide whether to play again or not*

*//INPUT*

*Ans;*

*//If answer is ‘Y’ or ‘y’*

*Game();*

*Return(true);*

*//else*

*Return(false);*

*//End If/else statement*

*END*

**uTD Function-**

//START

//Prompt user to decide whether to hit or stand

//INPUT

Draw;

//If draw is ‘h’ or ‘H’

Return(true);

//else

Return(false);

//END

**winner Function-**

*//START*

*//Declare variables*

*//CALL FOR FUNCTION*

*Score(pHand,dHand);*

*//If playerScore is less than 22 and is greater than the dealerScore or the*

*//dealerScore is greater than 21 and playerScore is less than 22*

*Cout<< “Congrats! You win!”<< endl;*

*//else*

*//If playerScore is equal to dealerScore*

*Cout<< “It’s a tie!” << endl;*

*//else*

*Cout<< “Sorry! You lose! Maybe next time!” << endl;*

*//END If/else statements*

*//END*

**chckBus Function-**

*//START*

*//Declare variables*

*//If playerScore is greater than 21*

*Cout<< “You bust with” << handval(pHand)<< “points.”<<endl;*

*//Else if dealerScore is greater than 21*

*Score(pHand,dHand);*

*Cout<< “The dealer bust! You win!” << endl;*

*pAH(Play);*

*//END If/elseif statements*

*//END*

**bD Function-**

*//START*

*//If cardVal(hand[0]) mod 100 is 10 and cardVal(hand[1]) mod 11*

*Cout<< “You pulled a backdoor”<<endl;*

*//END*

**bJ Function-**

*//START*

*//Declare variables*

*//If playerScore is 21 and dealerScore does not equal 21*

*Score(pHand,dHand);*

*Cout<< “Blackjack! Congrats you win!”<<endl;*

*bD(pHand);*

*pAH(Play);*

*//else if playerScore is 21 and dealerScore is 21*

*Score(pHand,dHand);*

*Cout<< “The Dealer and you both got Blackjack!”<<endl;*

*pAH(Play);*

*//END If/elseif statements*

*//END*

**nBJ Funtion-**

*//START*

*//Declare variables*

*//If playerScore is 21 and dealerScore is not 21*

*Score(pHand,dHand);*

*Cout<< “Blacjack! Congrats you win!”<<endl;*

*bD(pHand);*

*pAH(Play);*

*//END If statement*

*//END*

**score Function-**

*//START*

*//Output player and dealer hand with points*

*//Call on other functions*

*//END*

**softOrHard Function-**

*//START*

*//Declare variables*

*//If checkAce is 1 or 11*

*//Output to user to decide whether to use Ace as 1 or 11*

*//Input*

*SoftOrHard;*

*//If SoftOrHard is equal to 1*

*//If checkAce is equal to 11*

*pHand[i]+=13;*

*//End second if statement*

*//Else if SoftOrHard is equal to 11*

*//If checkAce is equal to 1*

*pHand[i]-=13;*

*//End second if statement*

*//Else if SoftOrHard is neither 1 or 11*

*Cin.clear();*

*Cin.ignore(numeric\_limits<streamsize>::max());*

*Cout<< “Please enter the 1 or 11:”<<endl;*

*softOrHard(pHand);*

*//END If/else if statements*

*//END*

**softOrHardD Function-**

*//START*

*//FOR given iteration bounds*

*//If cardVal(dHand[i]) is equal to 1 or 11*

*//If cardVal(dHand[i] is equal to 11*

*//If handval(dHand) minus cardVal(dHand[i]) +1 is greater than handval(pHand))*

*//If handval(dHand) minus cardVal(dHand[i]) +1 is less than 22*

*dHand[i]+=13;*

*//END second fourth if statement*

*//Else if handval(dHand) is greater than 21*

*dHand[i]+=13;*

*//Else*

*//If handval(dHand) minus cardval(dHand[i]) +11 is greater than //handval(pHand)*

*//If handval(dHand) minus cardVal(dHand[i]) plus 11 is less than 22*

*dHand[i]-=13;*

*//END ALL IF/ELSEIF/ELSE STATEMENTS*

*//END*

**bjRules Function-**

//START

//OUTPUT RULES OF THE GAME

//END

# Program:

/\*

\* File: main.cpp

\* Author: Isaac Kang

\* Created on July 12, 2016, 11:14 PM

\* Purpose: Project v2

\*/

#include <cstdlib>

#include <iostream>

#include <iomanip>

#include <limits>

#include <stdlib.h>

#include <fstream>

using namespace std;

/\*

\*

\*/

int level;

int randNum(int low, int high);

int cardVal(int card);

int pTC(int deck[]);

int handval(const int hand[]);

bool pAH(char&);

bool uTD(char&);

void initD(int deck[]);

void dump(int deck[], int size);

void shuffle(int deck[], int size);

void ShwCrd(int card);

void shwCrds(const int cards[], int numCards, bool hideFirstCard);

void winner(const int pHand[], const int dHand[]);

void chckBus(const int pHand[], const int dHand[], char Play);

void bJ(const int pHand[], const int dHand[], char Play);

void nBJ(const int pHand[], const int dHand[], char Play);

void game();

void score(const int pHand[], const int dHand[]);

void bD(const int hand[]);

void softOrHard(int pHand[]);

void softOrHardD(int dHand[], int pHand[]);

void addToHand(int hand[], int cardToAdd);

void hitUntilStand(int dHand[], int deck[], int pHand[]);

void game()

{

// Welcome user to game

cout << "Let's begin!" << endl << endl;

// Declare variables

char Play = 'N';

int deck[52], pHand[10] = {0}, dHand[10] = {0};

// Do-while loop with different void functions used

do

{

char Draw = 'H';

initD(deck);

shuffle(deck, 51);

addToHand(pHand, pTC(deck));

addToHand(dHand, pTC(deck));

addToHand(pHand, pTC(deck));

addToHand(dHand, pTC(deck));

cout << "The Dealer shuffled the deck and gave you your cards." << endl << endl;

nBJ(pHand, dHand, 'y');

bJ(pHand, dHand, 'Y');

cout << "Your hand: " << endl;

shwCrds(pHand, 10, false);

cout << endl;

cout << "Dealer's hand: " << endl;

shwCrds(dHand, 10, true);

cout << endl;

softOrHard(pHand);

while(uTD(Draw))

{

addToHand(pHand, pTC(deck));

cout << "The Dealer dealt you another card." << endl;

cout << "Your current hand: " << endl;

cout << endl;

shwCrds(pHand, 10, false);

softOrHard(pHand);

chckBus(pHand, dHand, 'y');

bJ(pHand, dHand, 'Y');

}

hitUntilStand(dHand, deck, pHand);

chckBus(pHand, dHand, 'Y');

bJ(pHand, dHand, 'Y');

winner(pHand, dHand);

cout << endl;

}

// Prompt user to see if they would like to play again

while (pAH(Play));

}

// Void function for game 2, which initializes the deck before the game begins

void initD(int deck[])

{

int diff = 101;

int i = 0;

for (i = 0 ; i <= 13; i++)

{

deck[i] = diff++;

}

for (i = 13 ; i <= 26; i++)

{

deck[i] = diff++ + 100 - 14;

}

for (i = 26 ; i <= 39; i++)

{

deck[i] = diff++ + 200 - 28;

}

for (i = 39 ; i <= 51; i++)

{

deck[i] = diff++ + 300 - 42;

}

}

// Void function for game 2, which shuffles the deck so that it generates random

// cards every time

void shuffle(int deck[], int size)

{

for(int i = 0; i < 500; i++)

{

int T1 = 0;

int R1 = randNum(0, size);

int R2 = randNum(0, size);

T1 = deck[R1];

deck[R1] = deck[R2];

deck[R2] = T1;

}

}

// Void function for game 2, which shows the players hand

void ShwCrd(int card)

{

if(card == 0)

{

cout << "";

}

else

{

switch(card % 100)

{

case 1:

cout << "A";

break;

case 11:

cout << "J";

break;

case 12:

cout << "K";

break;

case 13:

cout << "Q";

break;

case 14:

cout << "A";

break;

default:

cout << card % 100;

}

}

}

// Void function for game 2, which hides first card

// and shows the second card of the computer

void shwCrds(const int deck[], int numCards, bool hideFirstCard)

{

if(hideFirstCard)

{

cout << "\*\* ";

}

else

{

ShwCrd(deck[0]);

cout << " ";

}

for(int i = 1; i < numCards; i++)

{

if(deck[i] != 0)

{

ShwCrd(deck[i]);

cout << " ";

}

else

{

cout << " ";

}

}

}

// Void function for game 2, which dumps used cards so that the deck constantly

// changes and has random cards

void dump(int deck[], int size)

{

for(int i = 0; i < size; i++)

{

cout << i + 1 << ".) " << deck[i] << endl;

}

}

int randNum(int low, int high) {

static bool firstTime = true;

int randNum;

if (firstTime) {

srand(static\_cast <unsigned int> (time(NULL)) );

firstTime = false;

}

randNum = rand() % (high - low + 1) + low;

return randNum;

}

// Function for game 2, to determine the card values

int cardVal(int card)

{

int cardVal;

switch(card % 100)

{

// switch used for the different card possibilities and their value

case 1:

cardVal = 11;

break;

case 11:

case 12:

case 13:

cardVal = 10;

break;

case 14:

cardVal = 1;

break;

default:

cardVal = (card % 100);

}

return cardVal;

}

// Function for game 2, to pick out the top of the deck.

int pTC(int deck[])

{

int topCard;

for(int i = 0; i < 51; i++)

{

if(deck[i] != 0)

{

topCard = deck[i];

deck[i] = 0;

return topCard;

}

}

}

// Void function for game 2, which adds a card to both users

void addToHand(int hand[], int cardToAdd)

{

for(int i = 0; i < 9; i++)

{

if(hand[i] == 0)

{

hand[i] = cardToAdd;

break;

}

}

}

// Void function for game 2, which allows the user to choose between

// hit or stand

void hitUntilStand(int dHand[], int deck[], int pHand[])

{

for(int i = 0; i < 9; i++)

{

if(handval(dHand) < 17)

{

if(level == 1)

{

addToHand(dHand, pTC(deck));

}

else if(level == 2)

{

addToHand(dHand, pTC(deck));

softOrHardD(dHand, pHand);

}

}

else

{

if(i == 0)

{

cout << "The Dealer stands." << endl;

break;

}

else if(i == 1)

{

cout << "The Dealer hit a card and stands." << endl;

break;

}

else

{

cout << "The Dealer hit " << i << " cards and stands." << endl;

break;

}

}

}

}

// Function that calculates the total value of all your cards combined

int handval(const int hand[])

{

int addCardValues = 0;

for(int i = 0; i < 9; i++)

{

addCardValues = addCardValues + cardVal(hand[i]);

}

return addCardValues;

}

// Statement to prompt the user if they want to play again or not

bool pAH(char& ans)

{

cout << endl;

cout << "Would you like to play another round?? (Y/N): ";

cin >> ans;

if(ans == 'Y' || ans == 'y')

{

game();

return(true);

}

else

{

return(false);

}

}

// Statement that allows user to choose between hit and stand

bool uTD(char& draw)

{

cout << "Would you like to hit or stand? (H/S): " << endl;

cin >> draw;

if(draw == 'h' || draw == 'H')

{

return(true);

}

else

{

return(false);

}

}

// Void function for game 2, which calculates both scores and determines who

// the winner is

void winner(const int pHand[], const int dHand[])

{

int playerScore = handval(pHand);

int dealerScore = handval(dHand);

score(pHand, dHand);

if(playerScore < 22 && playerScore > dealerScore || dealerScore > 21 && playerScore < 22)

{

cout << "Congrats! You win!" << endl;

}

else

{

if(playerScore == dealerScore)

{

cout << "It's a tie!" << endl;

}

else

{

cout << "Sorry! You lose! Maybe next time!" << endl;

}

}

}

// Void function for game 2, which checks to see if the user busts

void chckBus(const int pHand[], const int dHand[], char Play)

{

int playerScore = handval(pHand);

int dealerScore = handval(dHand);

if(playerScore > 21)

{

cout << "You bust with " << handval(pHand) << " points." << endl;

pAH(Play);

}

else if(dealerScore > 21)

{

score(pHand, dHand);

cout << "The Dealer bust! You Win!" << endl;

pAH(Play);

}

}

void bD(const int hand[])

{

if(cardVal(hand[0]) % 100 == 10&& cardVal(hand[1]) % 100 == 11)

{

cout << "You pulled a backdoor" << endl;

}

}

// Void function for game 2, which checks to see if user receives

// a bJ after a hit or stand

void bJ(const int pHand[], const int dHand[], char Play)

{

int playerScore = handval(pHand);

int dealerScore = handval(dHand);

if((playerScore == 21) && (dealerScore != 21))

{

score(pHand, dHand);

cout << "Blackjack! Congrats you win!" << endl;

bD(pHand);

pAH(Play);

}

else if((playerScore == 21) && (dealerScore == 21))

{

score(pHand, dHand);

cout << "The Dealer and you both got Blackjack!" << endl;

pAH(Play);

}

}

// Void function for game 2, which checks to see if user receives a

// black jack right at the start when they receive their first two

// cards

void nBJ(const int pHand[], const int dHand[], char Play)

{

int playerScore = handval(pHand);

int dealerScore = handval(dHand);

if((playerScore == 21) && (dealerScore != 21))

{

score(pHand, dHand);

cout << "Blackjack! Congrats you win!" << endl;

bD(pHand);

pAH(Play);

}

}

// Void function for game 2, which calculates the score between the user

// and computer

void score(const int pHand[], const int dHand[])

{

cout << "Player hand: ";

shwCrds(pHand, 10, false);

cout << handval(pHand) << " (pts)." << endl;

cout << "Dealer hand: ";

shwCrds(dHand, 10, false);

cout << handval(dHand) << " (pts)." << endl;

}

// Void function for game 2, which determines whether the user needs to

// go soft or hard

void softOrHard(int pHand[])

{

for(int i = 0; i < 9; i++)

{

int checkAce = cardVal(pHand[i]);

int SoftOrHard;

if(checkAce == 1 || checkAce == 11)

{

cout << "Would you like your Ace to count as 1 or 11 points? (1/11): ";

cin >> SoftOrHard;

if(SoftOrHard == 1)

{

if(checkAce == 11)

pHand[i] = pHand[i] + 13;

}

else if(SoftOrHard == 11)

{

if(checkAce == 1)

{

pHand[i] = pHand[i] - 13;

}

}

else if (SoftOrHard != 1 || SoftOrHard != 11)

{

cin.clear();

cin.ignore(numeric\_limits <streamsize> ::max());

cout << "Please enter the 1 or 11: " << endl;

softOrHard(pHand);

}

}

}

}

// Void function for game 2, which determines whether the computer needs to

// go soft or hard

void softOrHardD(int dHand[], int pHand[])

{

for(int i = 0; i < 9; i++)

{

if(cardVal(dHand[i]) == 1 || cardVal(dHand[i]) == 11)

{

if(cardVal(dHand[i]) == 11)

{

if (handval(dHand) - cardVal(dHand[i]) + 1 > handval(pHand))

{

if (handval(dHand) - cardVal(dHand[i]) + 1 < 22)

{

dHand[i] = dHand[i] + 13;

}

}

else if (handval(dHand) > 21)

{

dHand[i] = dHand[i] + 13;

}

else

{

if (handval(dHand) - cardVal(dHand[i]) + 11 > handval(pHand))

{

if (handval(dHand) - cardVal(dHand[i]) + 11 < 22)

{

dHand[i] = dHand[i] - 13;

}

}

}

}

}

}

}

// Set the rules for Black Jack

void bjRules()

{

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl

<< "\*\*\*\*\*\*Rules\*\*\*\*\*\*" << endl

<< "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

cout << "Blackjack is a game where the goal is to hit 21 through a combination of cards or have the dealer bust (over 21)." << endl

<< "Each number has its own face value (2,3,4,5,etc…) while a Jack, Queen, King has the value of 10." << endl

<< "An ace can be represented as a 1 (hard) or an 11 (soft)." << endl

<< "The user is dealt two cards and the dealer is dealt two cards." << endl

<< "The user can select to hit (get another card) or stay (not get another card)." << endl

<< "The dealer must hit on a soft 17. For example, if the user has a 6 and A." << endl

<< "The value is 17 and it is known as soft (as in the ace is represented by 11)." << endl

<< "The user can stay or get another card, in which the Ace may turn into the value 1, becoming a hard ace." << endl

<< "One the user chooses to stay, the dealer must then continue to hit until it has a hard 17 or higher." << endl

<< "If the dealer has a higher number than the user, the user loses their initial bet." << endl

<< "Otherwise, the user wins." << endl;

}

int main(int argc, char\*\* argv) {

int ans;

ofstream out;

out.open("game.dat");

do{

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl

<< "Welcome to Black Jack" << endl

<< "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

cout << "Please choose an option:" << endl;

cout << "1) Play Black Jack" << endl;

cout << "2) Quit" << endl;

cin >> ans;

switch(ans){

case 1:

cin.clear();

bjRules();

game(); break;

case 2:

cin.clear();

cout << "See you next time!" << endl; break;

default:

cout << "Error. Please enter a valid number!" << endl;

cin >> ans; break;

}

}while(ans != 2);

out.close();

return 0;

}