**Black Jack**

**Card Game**

By:John Haller

CSC 5

Spring 2018

42483

Table of Contents

**Introduction\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2**

**The Game\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2**

**Object of the Game\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2**

**Rules\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2**

**Scoring of cards\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_3**

**Going Through the Game\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_3**

**Logic of the game\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_4**

**PseudoCode\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_4**

**FlowChart\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_5**

**Concepts Utilized\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_14**

**Proof Program Works\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_16**

**Full Code\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_18**

**Introduction**

Black Jack or sometimes 21, is a card game that is played by friends or family looking for a simple way to have fun. Or even at casinos with people waging money on the outcome. As well as even at the professional level with high stakes being waged on the highest level of players. The rules of the game are very simple and anyone can get started with it fairly easily, but there is a low of thought and skill that separates the good players from the bad ones.

**The Game**

**Object of the Game**

To win by getting a higher score than the dealer but while keeping your score below 21

And to win as many hands as possible until you are done playing.

**Rules**

1. For my iteration of the game each player is dealt a starting hand of 2 cards.
2. Then one by one each player decides if they are going to get another card (hit) or if they are going to stop at their current cards (hold).
3. Players may hit as many times as they like until they either choose to stop or have gotten a total score over 21 (busting)
4. The dealer must continue hitting until he has a score of at least 17

**Scoring of Cards**

1. Cards 2-10 have a value equal to their number (i.e a 4 of hearts is worth 4 points)
2. Face cards are all worth 10 points
3. Aces are worth either 1 or 11, depending on if you will bust or not

**Going Through the Game**

**Player Section**

The game starts with dealing both the player’s and dealer’s hand, although the player with only see their hand for now. It will then show you your current score and prompt you on whether or not you would like to hit. If so it will then show you your new hand, including your new card. And displaying your new score. It will then ask if you want to hit again, if yes then it will repeat the previous process. If no then it will output your final hand and score. If you bust while you are hitting it will cease to prompt you to hit and notify you that you have busted and continue on to the dealer’s portion. If you chose not to hit in the first place then it will once more display your score and move on.

**Dealer Section**

Now that we are in the Dealer section the player no longer has to do anything. It will start with displaying the dealer’s hand and score. Then it moves on to decide if the dealer is going to hit or hold. If he hits it will go through the same process of displaying hand, score, and if necessary a bust message.

**Results**

At this point the winner is being calculated and will output who wins or if it is a tie. The player will then be prompted if they want to play another round, if so this whole process will repeat. If they decide they are done playing the program will then output who won and lost for each round that was played with a ‘W’ for win or a ‘L’ for loss. After that the programs exit stage right and it is all over.

**Logic of the Game**

**PseudoCode**

Opening comments

System Libraris

Iostream

Cstdlib

Ctime

Vector

Iomanip

2-D Array Column

Players

Function Prototypes

filDeck

selSort

suitVal

faceVal

prtDeck

shuffle

deal

prntHnd

score

winner

pause

pause(overloaded)

bust

display

markBub

Enter main

Set random number seed

Declare Variables,Arrays, and Vectors

(more later)

Fill the deck

Sort the deck using selection sort

Shuffle the deck

Sort using marksort

Shuffle once more

Increment rounds counter

Declare variables that need

To be initialized each round

Deal player hand

Deal dealer hand

Print player hand

Score player hand

Determine if hitting or not

If yes

Deal new card

Print new hand

Score new hand

Determine if lower than 21

Hit again?

Bust?

Done hitting

Run bust function

If bust=true set bool

Print final player score

Print dealer hand

Score dealer hand

while(score<17)

Dealer hits

Print new hand

Score new hand

Run bust function

If bust=true set bool

Done hitting if true

Hit again

Determine winner

Run winner function

Output winner

Store winner and looser for later

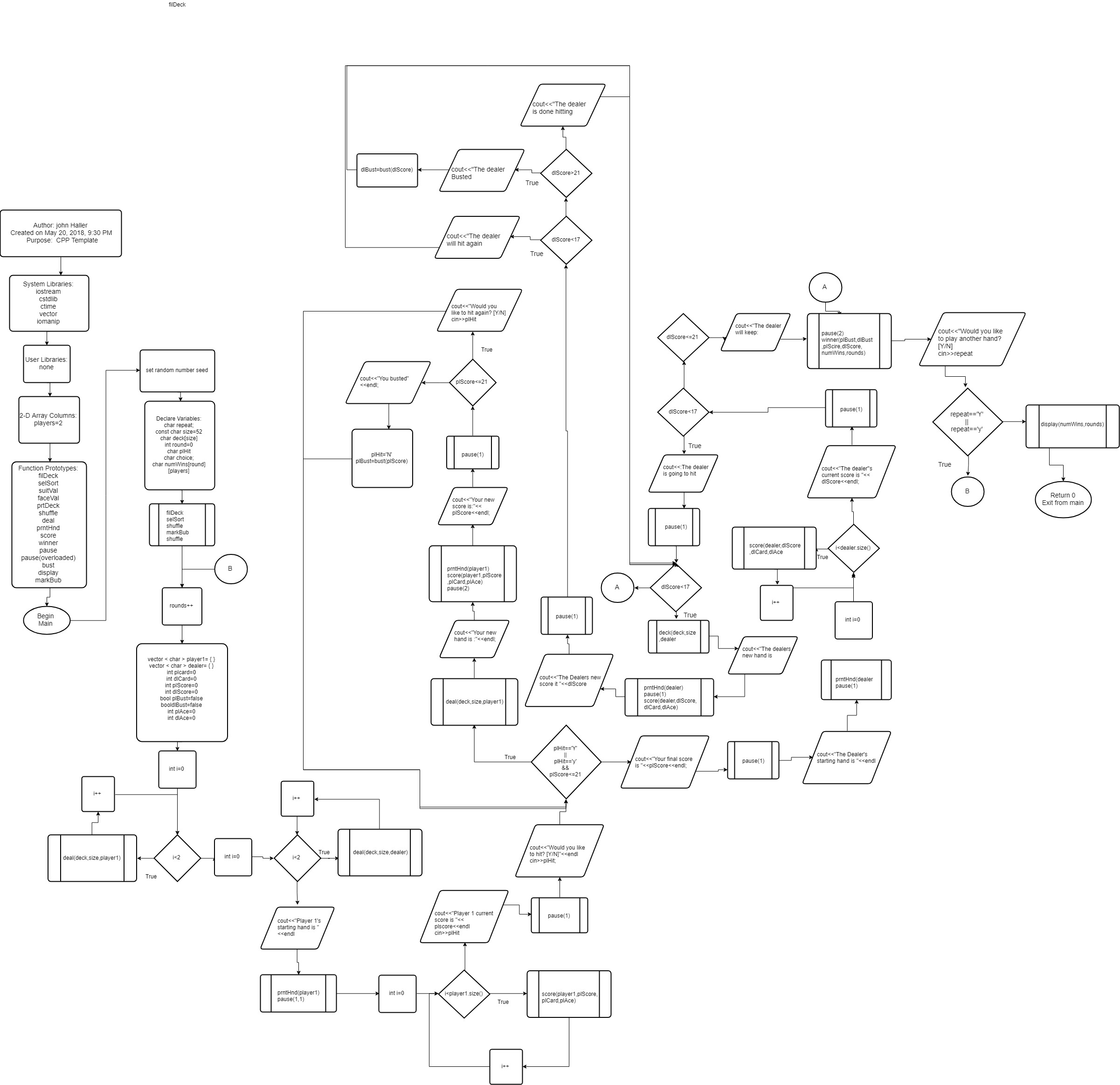
Prompt to play again

Display results of all rounds

Exit program

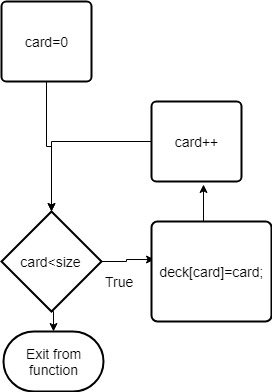
**FlowChart**

**Main**

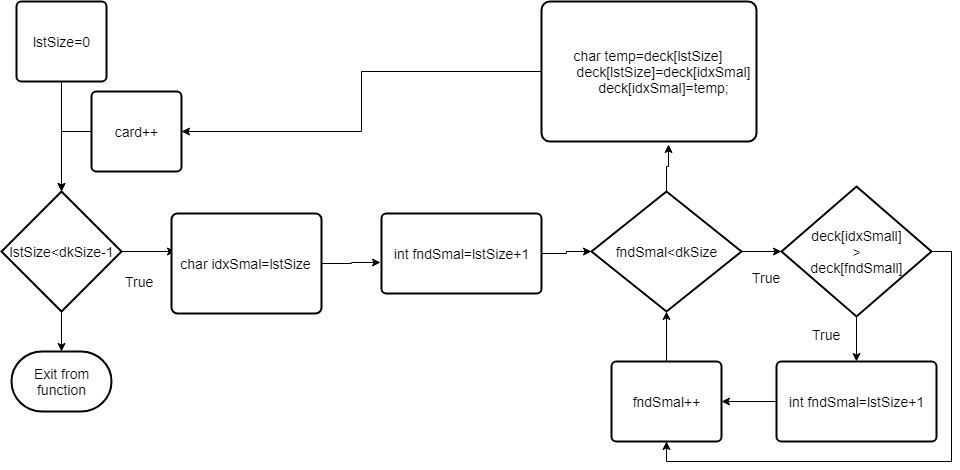


**Functions**

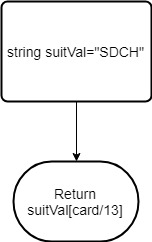
**filDeck**

****

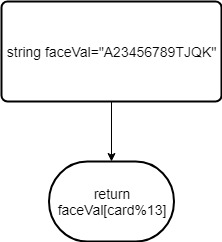
**selSort**

****

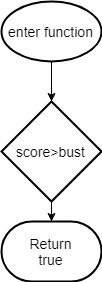
**suitVal**

****

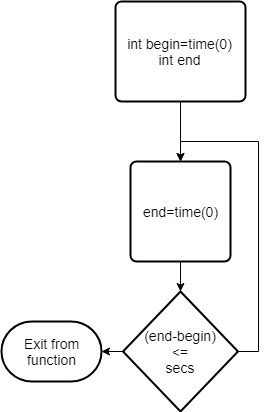
**faceVal**

****

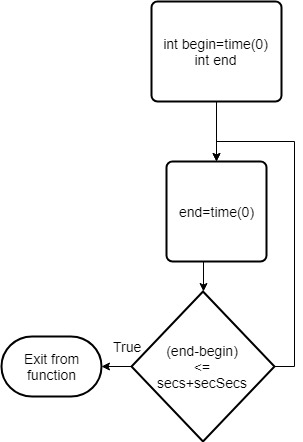
**Bust**

****

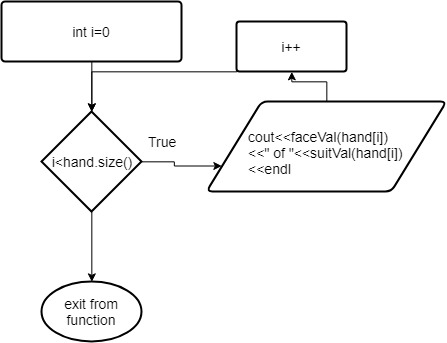
**Pause**

****

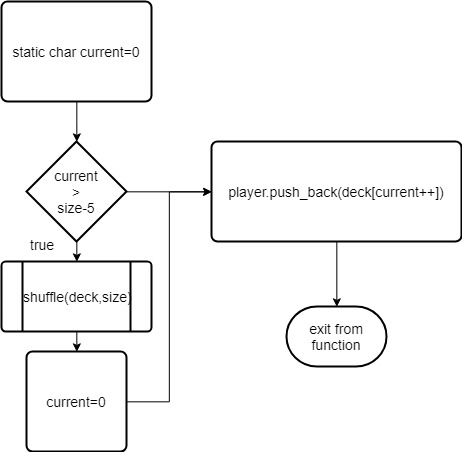
**Pause Overloaded**

****

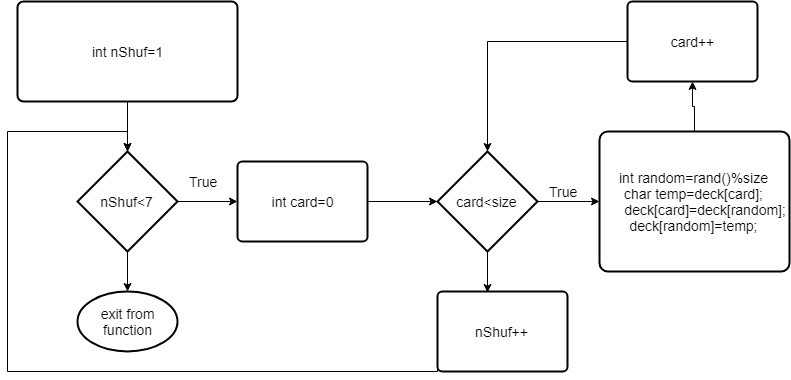
**PrntHnd**

****

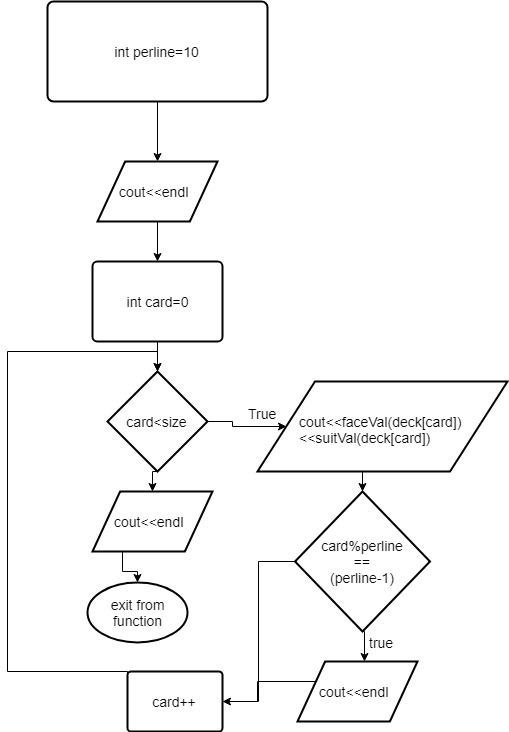
**Deal**

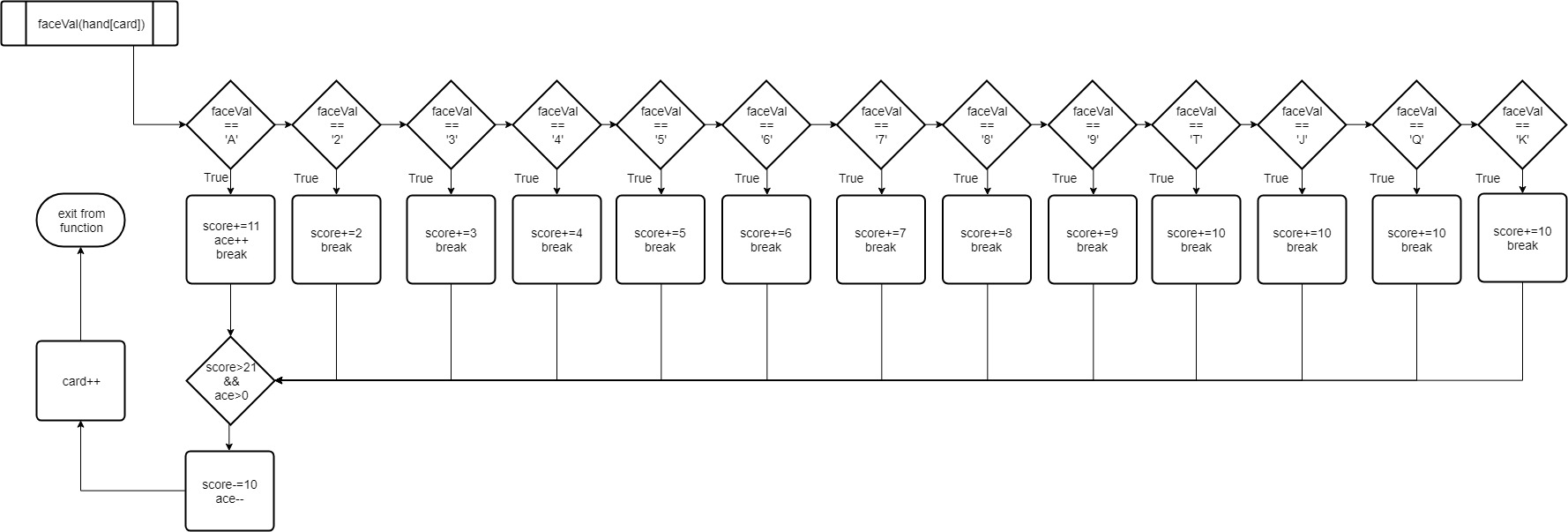
****

**Shuffle**

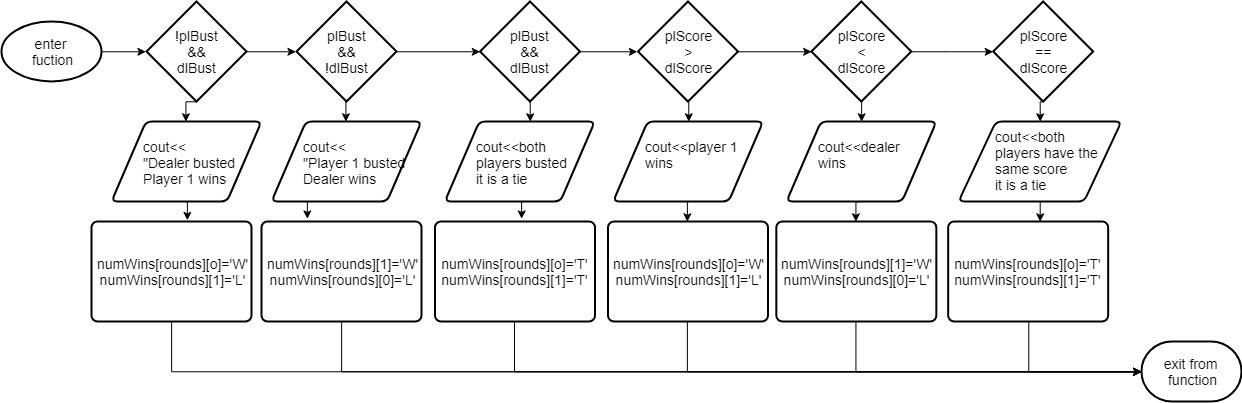
****

**prtDeck**

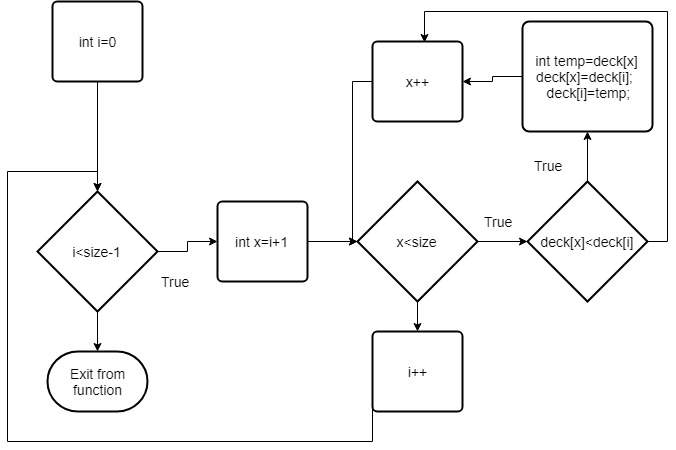
****

**Score**

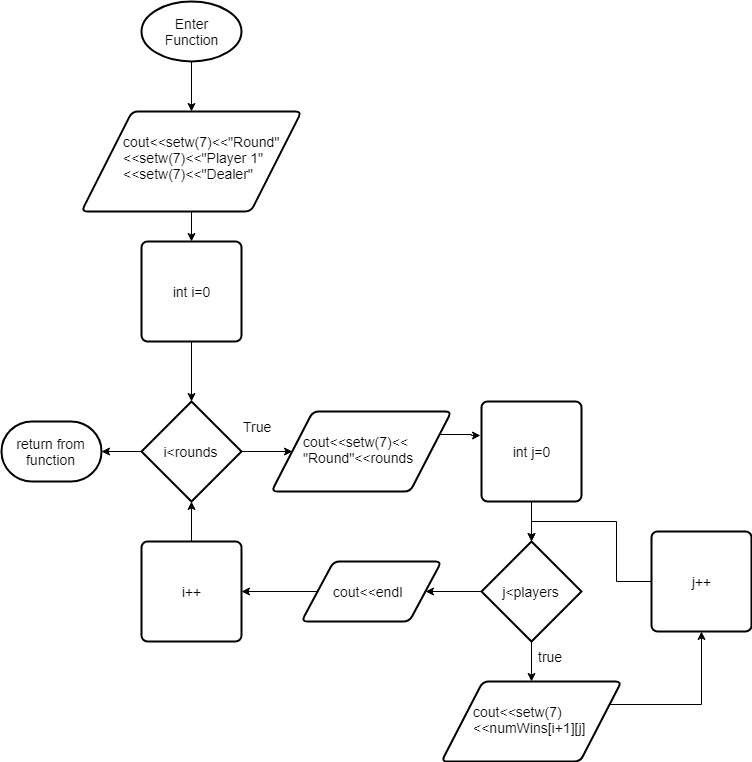
**Winner**

****

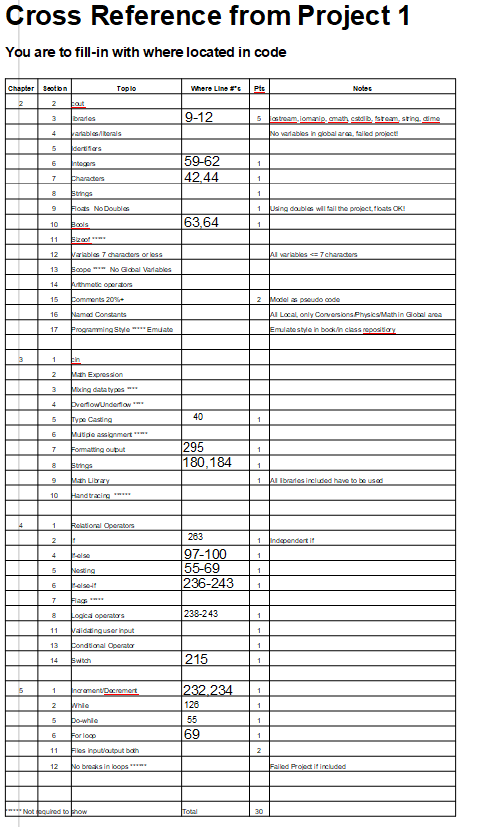
**markBub**

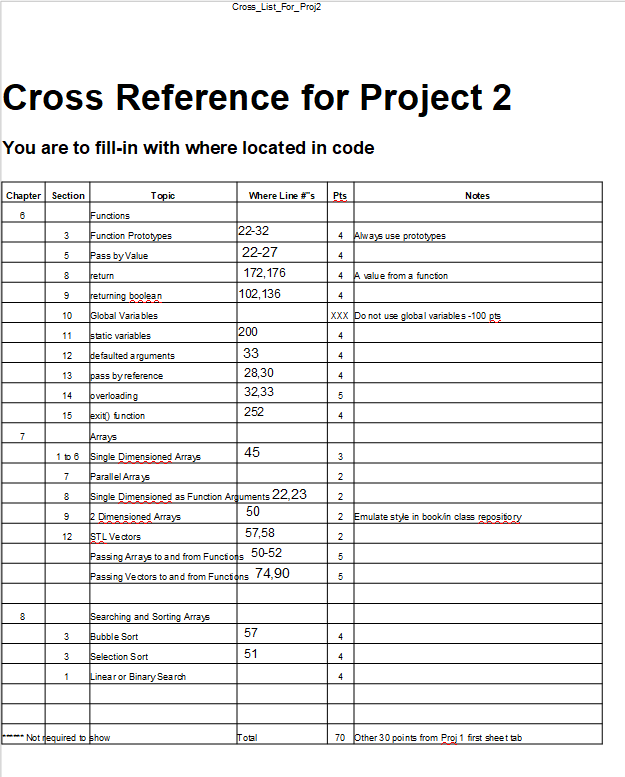
****

**Display**

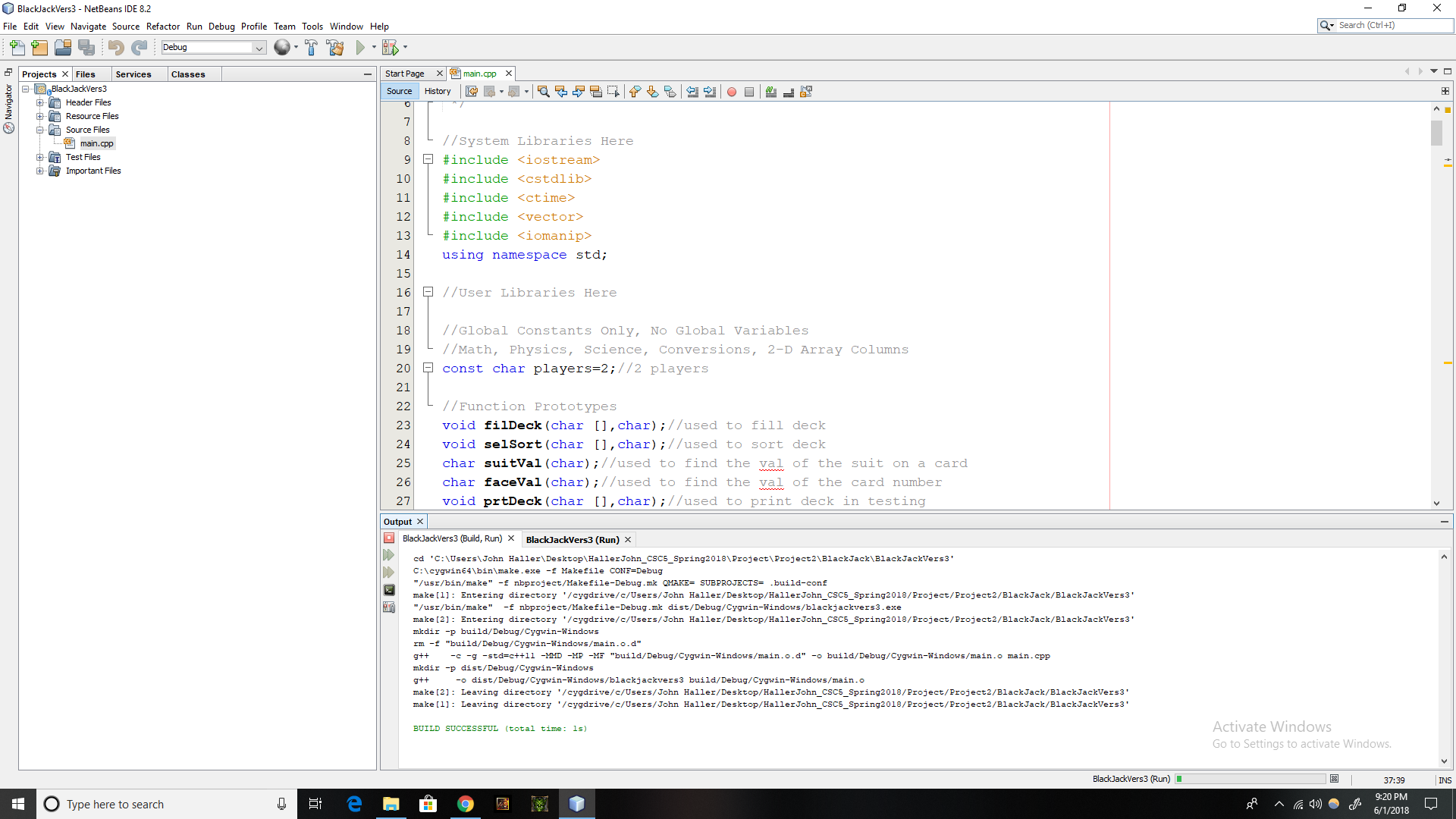
****

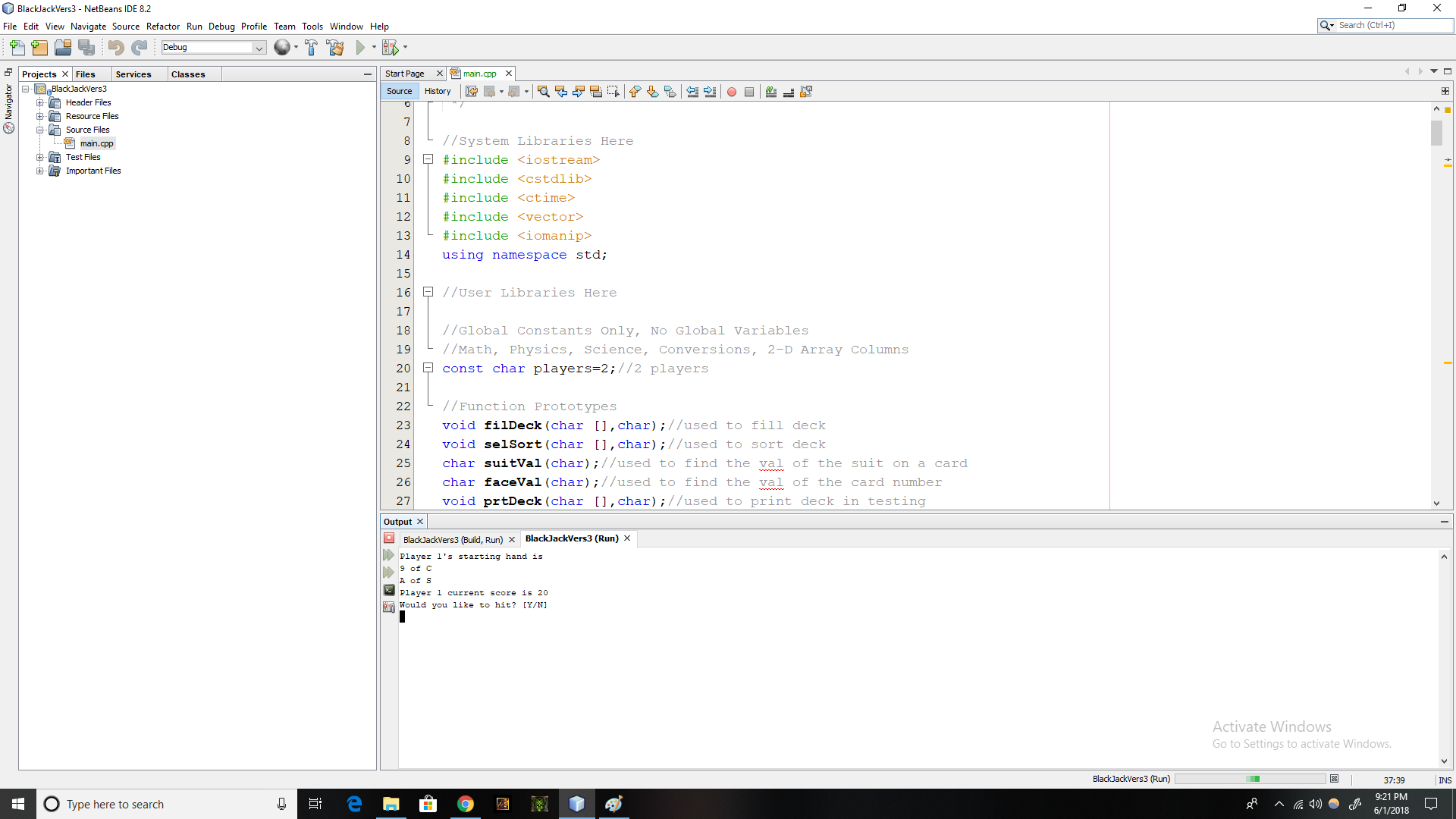
**Concepts Utilized**

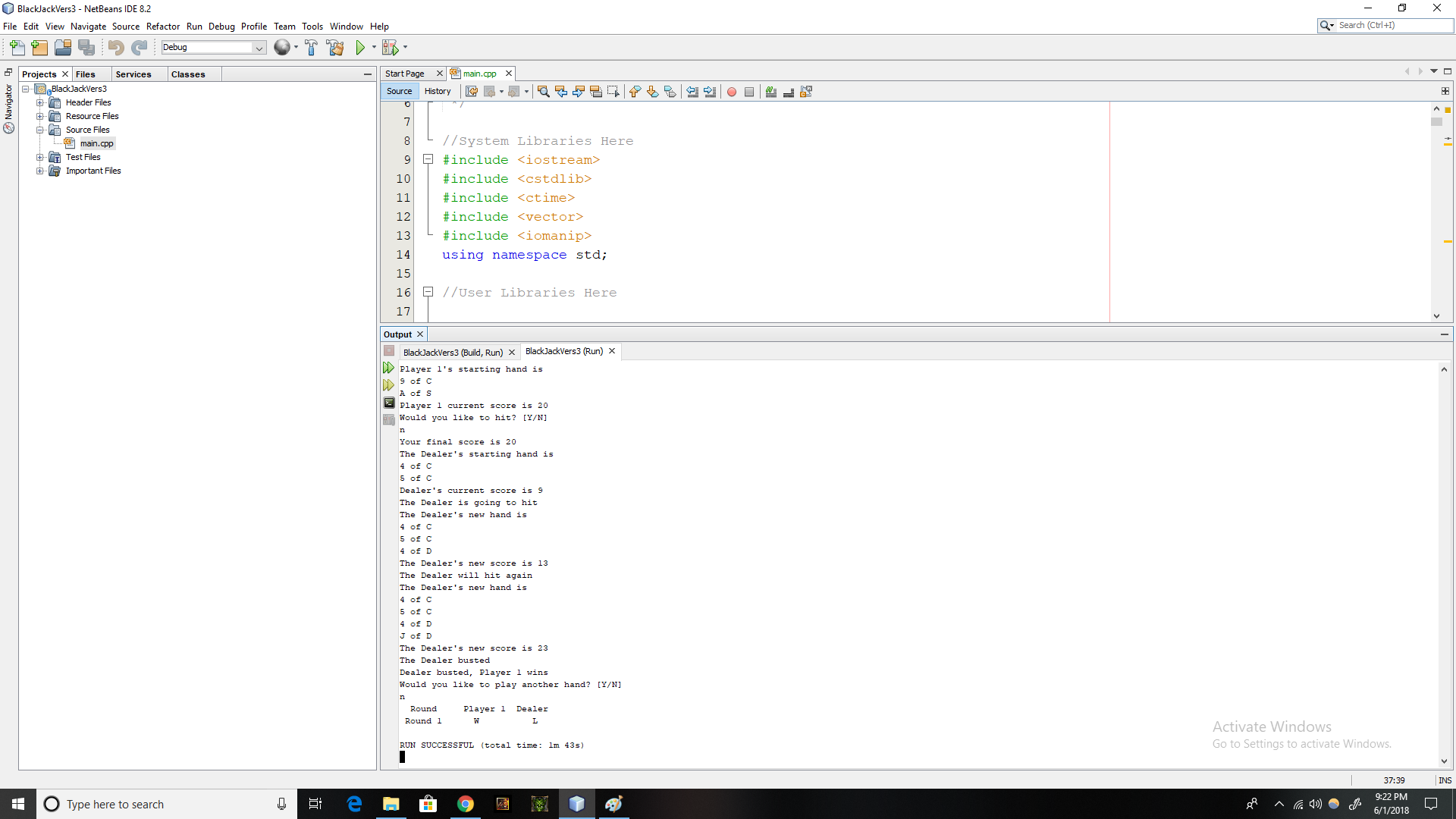
****



**Proof Program Works**

****

****

****

**Full Code**

/\*

\* File: CPPTemplate.cpp

\* Author: john Haller

\* Created on May 20, 2018, 9:30 PM

\* Purpose: Black Jack

\*/

//System Libraries Here

#include <iostream>

#include <cstdlib>

#include <ctime>

#include <vector>

#include <iomanip>

using namespace std;

//User Libraries Here

//Global Constants Only, No Global Variables

//Math, Physics, Science, Conversions, 2-D Array Columns

const char players=2;//2 players

//Function Prototypes

void filDeck(char [],char);//used to fill deck

void selSort(char [],char);//used to sort deck

char suitVal(char);//used to find the val of the suit on a card

char faceVal(char);//used to find the val of the card number

void prtDeck(char [],char);//used to print deck in testing

void shuffle(char [],char);//used to shuffle deck

void deal(char [],char,vector < char > &);//used to deal deck

void prntHnd(vector < char >);//used to print hands

void score(vector < char >, int &,int &, int &);//used to score cards

void winner(bool,bool,int,int,char [][players],int);//used to determine winner

void pause(int);//used to pause

void pause(int,int);//overloaded pause

bool bust(int,int bust=21);//used to determine if bust or not

void display(char [][players],int);//used to display scoreboard

void markBub(char [],char);//mark sort

//Program Execution Begins Here

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

//Seed random number generator

srand(static\_cast <unsigned int>(time(0)));

//Declare all Variables, no doubles

char repeat;//Used to play another hand or not

const char size=52;//number of cards in a deck

char deck[size];//deck used to deal hands

int rounds=0;//Number of rounds played

char plHit;//Used to determine if player wants to hit first time

char choice;//Used to decide if player wants to hit again

char numWins[rounds][players];//array to keep track of wins/losses

//Fill the deck and shuffle it

filDeck(deck,size);//fill the deck

selSort(deck,size);//sort it using selection sort

shuffle(deck,size);//shuffle the deck

markBub(deck,size);//sort the deck using marksort

shuffle(deck,size);//shuffle the deck once more

//Play the game

do{

rounds++;//increment number of rounds played

//Declare all Variables, no doubles

vector< char > player1={ };//player 1 hand

vector< char > dealer= { };//dealer hand

int plCard=0;//current card being scored

int dlCard=0;//current card being scored

int plScore=0;//player score

int dlScore=0;//dealer score

bool plBust=false;//Set to true if player busts

bool dlBust=false;//Set to true if dealer busts

int plAce=0;//counter for player's aces

int dlAce=0;//counter for dealer's aces

//Deal Player 1 starting hand

for(int i=0;i<2;i++){//2 cards in starting hand

deal(deck,size,player1);

}

//Deal Dealer's starting hand

for(int i=0;i<2;i++){//2 cards in starting hand

deal(deck,size,dealer);

}

//Print player 1 hand

cout<<"Player 1's starting hand is "<<endl;

prntHnd(player1);

pause(1,1);

//Score player1 starting hand

for(int i=0;i<player1.size();i++){//loop the scoring for as many cards

score(player1,plScore,plCard,plAce);//as their are in hand

}

cout<<"Player 1 current score is "<<plScore<<endl;

pause(1);

//hit or not

cout<<"Would you like to hit? [Y/N]"<<endl;

cin>>plHit;

while(plHit=='Y'||plHit=='y'&&plScore<=21){

deal(deck,size,player1);//deals a new card

cout<<"Your new hand is "<<endl;

prntHnd(player1);//output hand with new card

score(player1,plScore,plCard,plAce);//score new hand

pause(2);

cout<<"Your new score is "<<plScore<<endl;

pause(1);

if(plScore<=21){//if player is still lower than 21

cout<<"Would you like to hit again? [Y/N]"<<endl;

cin>>plHit;

}else{//if player is over 21

cout<<"You Busted" <<endl;

plHit='N';//so it doesn't prompt again

plBust=bust(plScore);//set bust counter to true for p1

}

}

//Print final player 1 score

cout<<"Your final score is "<<plScore<<endl;

//Print Dealer's hand

pause(1);

cout<<"The Dealer's starting hand is "<<endl;

prntHnd(dealer);//print dealer hand

pause(1);

//Score dealer starting hand

for(int i=0;i<dealer.size();i++){//loop for number of cards in dl hand

score(dealer,dlScore,dlCard,dlAce);

}

cout<<"Dealer's current score is "<<dlScore<<endl;

pause(1);

//hit or not

if(dlScore<17){//if dealer has less than 17 he hits again

cout<<"The Dealer is going to hit"<<endl;

pause(1);

while (dlScore<17){//dl continues hitting until he reaches 17

deal(deck,size,dealer);

cout<<"The Dealer's new hand is "<<endl;

prntHnd(dealer);//print new card

pause(1);

score(dealer,dlScore,dlCard,dlAce);//score new card

cout<<"The Dealer's new score is "<<dlScore<<endl;

pause(1);

if(dlScore<17)cout<<"The Dealer will hit again"<<endl;

else if(dlScore>21){//if the dealer busted

cout<<"The Dealer busted"<<endl;

dlBust=bust(dlScore);

}

else cout<<"The Dealer is done hitting"<<endl;//if 17<dlHand<21

}

}else if(dlScore<=21)cout<<"The Dealer will keep"<<endl;

//^if the Dealers original hand is >=17

//Determine the winner

pause(2);

winner(plBust,dlBust,plScore,dlScore,numWins,rounds);//func to determine

cout<<"Would you like to play another hand? [Y/N]"<<endl;//winner

cin>>repeat;//play another hand

}while(repeat=='Y'||repeat=='y');

//Display results of rounds played

display(numWins,rounds);//display the wins/losses

//Exit Program!

return 0;

}

void filDeck(char deck[],char size){

for(int card=0;card<size;card++){//fills the array with number between 0-51

deck[card]=card;

}

}

void selSort(char deck[],char dkSize){//selection sort

for(int lstSize=0;lstSize<dkSize-1;lstSize++){

char idxSmal=lstSize;

for(int fndSmal=lstSize+1;fndSmal<dkSize;fndSmal++){

if(deck[idxSmal]>deck[fndSmal]){

idxSmal=fndSmal;

}

}

char temp=deck[lstSize];//swap begins here

deck[lstSize]=deck[idxSmal];

deck[idxSmal]=temp;

}

}

void markBub(char deck[],char size){//marksort

for(int i=0;i<size-1;i++){

for(int x=i+1;x<size;x++){

if(deck[x]<deck[i]){

int temp=deck[x];//swap begins here

deck[x]=deck[i];

deck[i]=temp;

}

}

}

}

char suitVal(char card){

string suitVal="SDCH";//inputs card from hand

return suitVal[card/13];//returns the suit of the card

}

char faceVal(char card){

string faceVal="A23456789TJQK";//inputs card from hand

return faceVal[card%13];//returns 1-9TJQKA

}

void prtDeck(char deck[],char size){//prints deck

int perline=10; //used in testing only

cout<<endl; //not present in final game

for(int card=0;card<size;card++){

cout<<faceVal(deck[card])<<suitVal(deck[card])<<" ";

if(card%perline==(perline-1))cout<<endl;

}

cout<<endl;

}

void shuffle(char deck[],char size){//shuffles the deck before being dealt

for(int nShuf=1;nShuf<=7;nShuf++){//shuffles 7 times

for(int card=0;card<size;card++){//does the swap 52 times

int random=rand()%size;//start of swap

char temp=deck[card];

deck[card]=deck[random];

deck[random]=temp;

}

}

}

void deal(char deck[],char size,vector < char > &player){//deals players hands

static char current=0;//counter for the current card

if(current>size-5){//Used to shuffle once their are only 5 cards remaining

shuffle(deck,size);//shuffle function

current=0;//reset current after shuffle

}

player.push\_back(deck[current++]);//pushback card into hand vectors

}

void prntHnd(vector < char > hand){//prints hand for player/dealer

for(int i=0;i<hand.size();i++){//loops for number of cards in hand

cout<<faceVal(hand[i])<<" of "<<suitVal(hand[i])<<endl;

}

}

void score(vector < char > hand, int &score, int &card,int &ace){//used to score

switch(faceVal(hand[card])){//returns the face value //cards

case'A':{score+=11;ace++;break;}//ace

case'2':{score+=2;break;}//2

case'3':{score+=3;break;}//3

case'4':{score+=4;break;}//4

case'5':{score+=5;break;}//5

case'6':{score+=6;break;}//6

case'7':{score+=7;break;}//7

case'8':{score+=8;break;}//8

case'9':{score+=9;break;}//9

case'T':{score+=10;break;}//10

case'J':{score+=10;break;}//jack

case'Q':{score+=10;break;}//queen

case'K':{score+=10;break;}//king

}

if(score>21&&ace>0){//used to determine if aces are present and score >21

score-=10;//minus 10 for each ace

ace--;//decrement ace counter for each -10

}

card++;//keeps track of which card is being scored from the hand

}

void winner(bool plBust,bool dlBust,int plScore,int dlScore,//determines winner

char numWins[][players],int rounds){

//start

if(!plBust&&dlBust){//Dealer busts not player 1

cout<<"Dealer busted, Player 1 wins"<<endl;

numWins[rounds][0]='W';

numWins[rounds][1]='L';

}

else if(plBust&&!dlBust){//player 1 busts not dealer

cout<<"Player 1 busted, Dealer wins"<<endl;

numWins[rounds][1]='W';

numWins[rounds][0]='L';

}

else if(plBust&&dlBust){//both bust

cout<<"Both players busted. It is a tie"<<endl;

numWins[rounds][0]='T';

numWins[rounds][1]='T';

}

else if(plScore>dlScore){//no busts player 1 had higher score

cout<<"Player 1 wins"<<endl;

numWins[rounds][0]='W';

numWins[rounds][1]='L';

}

else if(plScore<dlScore){//no busts dealer had higher score

cout<<"Dealer wins"<<endl;

numWins[rounds][1]='W';

numWins[rounds][0]='L';

}

else if(plScore==dlScore){//no busts both had same score

cout<<"Both players have the same score. It is a tie"<<endl;

numWins[rounds][0]='T';

numWins[rounds][1]='T';

}

}

void pause(int secs){

int begin=time(0);//time at beginning

int end;//time at end

do{

end=time(0);

}while((end-begin)<=secs);

}

void pause(int secs,int secSecs){

int begin=time(0);//time at beginning

int end;//time at end

do{

end=time(0);

}while((end-begin)<=secs+secSecs);

}

bool bust(int score,int bust){//determine if bust or not

if(score>bust)return true;//return true if bust

}

void display(char numWins[][players],int rounds){//displays scoreboard

cout<<setw(7)<<"Round"<<" "<<setw(7)<<"Player 1"<<" "//header

<<setw(7)<<"Dealer"<<endl;

for(int i=0;i<rounds;i++){//rows

cout<<setw(7)<<"Round "<<rounds;//displays which round

for(int j=0;j<players;j++){

cout<<setw(7)<<numWins[i+1][j]<<" ";//displays 'L' or 'W'

}

cout<<endl;

}

}