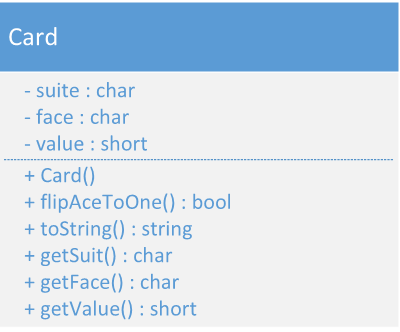
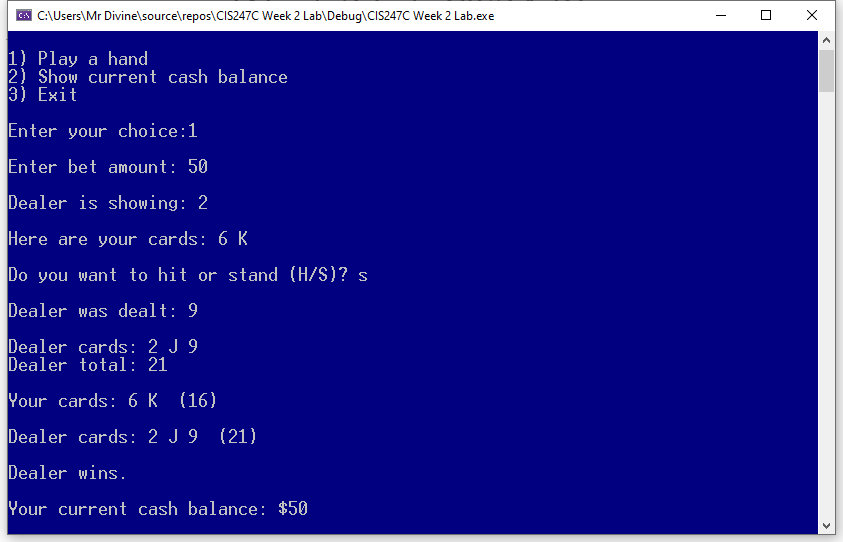
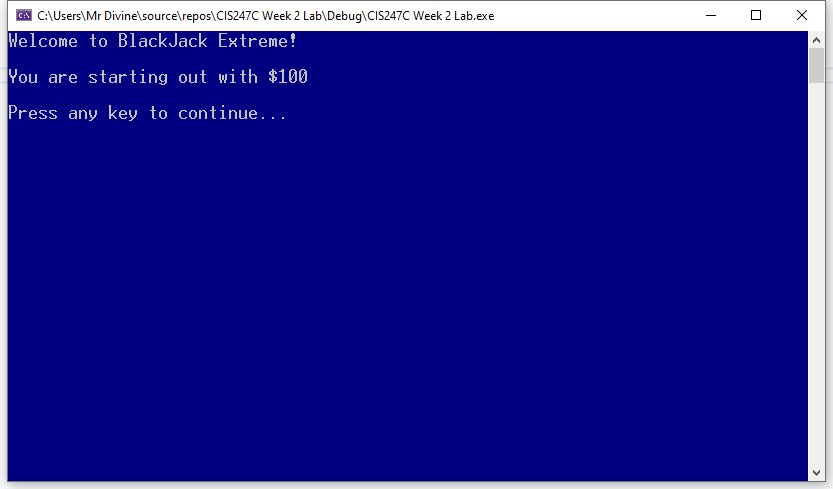
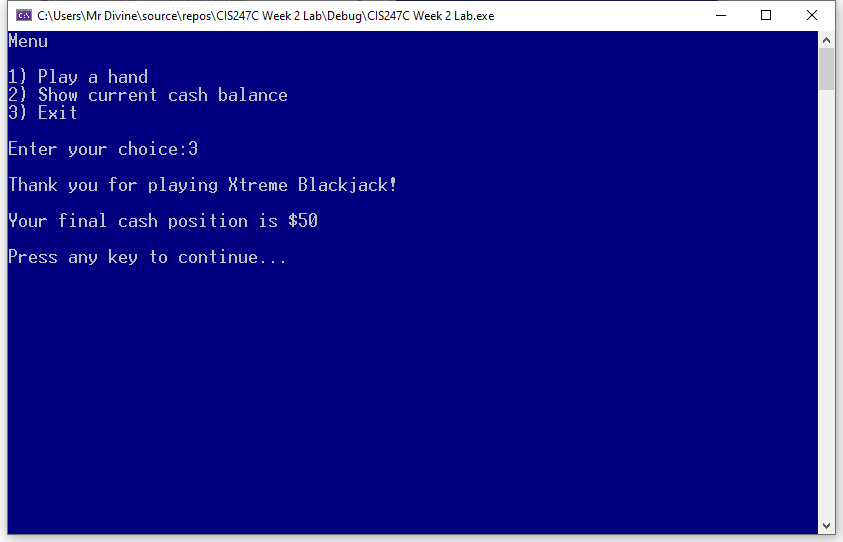
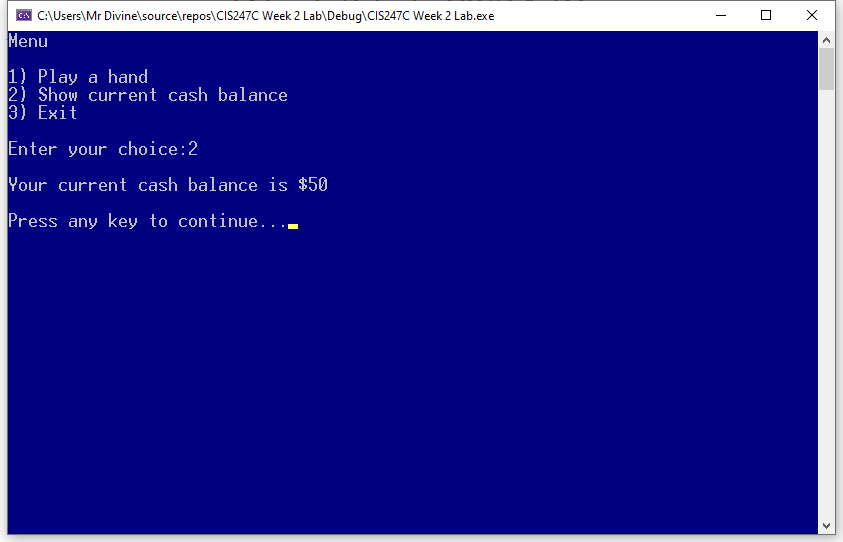
UML Class Diagram:



Screenshots:





**Code:**

**Source:**

/\*

Edward Alvarado

Class: CIS247C

Date: 3/8/2021

\*/

#include <iostream>

#include <string>

#include <conio.h>

#include "Card.h"

#include <vector> // Standard Template Library (STL)

using namespace std;

// prototypes

string showCards(vector<Card> cards);

short sumCardValues(vector<Card> cards);

void playHand(short &cash); // & means "address of" variable sent by reference

/// Entry point to the applicatrion

int main()

{

// set the starting cash and display it to the player

short cash = 100;

cout << "Welcome to BlackJack Extreme!" << endl;

cout << "\nYou are starting out with $" << cash << endl;

// Pause

cout << "\nPress any key to continue...";

\_getch();

// create the loop variable

short choice = 0;

// run the application loop

do

{

//show the menu and get player's choice

system("cls"); // clear the console

cout << "Menu\n" << endl;

cout << "1) Play a hand" << endl;

cout << "2) Show current cash balance" << endl;

cout << "3) Exit" << endl;

cout << "\nEnter your choice:";

cin >> choice;

// switch based on the choice

switch (choice)

{

case 1:

playHand(cash);

break;

case 2:

cout << "\nYour current cash balance is $" << cash << endl;

break;

case 3:

cout << "\nThank you for playing Xtreme Blackjack!" << endl;

cout << "\nYour final cash position is $" << cash << endl;

break;

default:

cout << "\nError. Please select from the menu." << endl;

break;

}

// Pause

cout << "\nPress any key to continue...";

\_getch();

} while (choice != 3); // ! means NOT

return 0;

}

/// Show the cards in the vector (resizable array)

string showCards(vector<Card> cards)

{

string output = "";

for (short i = 0; i < cards.size(); i++)

{

output += cards[i].toString() + " ";

}

return output;

}

/// Add up the total value of the cards

short sumCardValues(vector<Card> cards)

{

short total = 0;

for (Card c : cards)

{

total += c.getValue(); // total = total + c.getValue();

}

return total;

}

/// Play a single hand of Blackjack

void playHand(short& cash)

{

// create two ArrayLists that can hold card objects

// An Arraylist is a resizeable array

// in C++, an Arraylist is called a vector

vector<Card> dealerCards;

vector<Card> playerCards;

short dealerCardsTotal = 0;

short playerCardsTotal = 0;

// get bet amount

short bet = 0;

cout << "\nEnter bet amount: ";

cin >> bet;

// create two cards for the dealer and show the first one

Card card1;

Card card2;

dealerCards.push\_back(card1);

dealerCards.push\_back(card2);

dealerCardsTotal = sumCardValues(dealerCards);

cout << "\nDealer is showing: " << dealerCards[0].toString() << endl;

// create two cards for the player and show them both

playerCards.push\_back(Card()); //create the card and pit it directly into the ArrayList (vector)

playerCards.push\_back(Card());

playerCardsTotal = sumCardValues(playerCards);

cout << "\nHere are your cards: " << showCards(playerCards) << endl;

// give cards to the player until they stand ('S)

char answer = '?';

do

{

cout << "\nDo you want to hit or stand (H/S)? ";

cin.sync(); //flush the input stream (keyboard buffer)

cin >> answer;

cin.sync();

if (toupper(answer) == 'H')

{

//give a card to the player

Card c;

cout << "\nYou were dealt this card: " << c.toString() << endl;

playerCards.push\_back(c); // add card to player's hand

// sum up the card values

playerCardsTotal = sumCardValues(playerCards);

// did the player bust?

if (playerCardsTotal > 21)

{

//do you have an Ace that can be dropped to a 1 value?

for (Card c : playerCards)

{

if (c.getValue() == 11)

{

cout << "\nYour total is: " << playerCardsTotal << endl;

c.flipAcetoOne();

cout << "However, you have an Ace that was converted to '1' value" << endl;

playerCardsTotal = sumCardValues(playerCards);

cout << "\nYour new total is " << playerCardsTotal << endl;

// if we're good now break the loop, otherwise keep looping

if (playerCardsTotal <= 21)

break;

}

}

}

// show the cards and the total

cout << "\nHere are your cards: " << showCards(playerCards) << endl;

cout << "Your total is " << playerCardsTotal << endl;

// If busted stop loop

if (playerCardsTotal > 21)

answer = 'S';

}

}

while (toupper(answer) != 'S');

// if Player's card total is more than 21, player busted so take money away

if (playerCardsTotal > 21)

{

cout << "\nYou busted!" << endl;

cash = cash - bet;

}

else

{

// player stands so the dealer hits until 17 or greater

do

{

if (dealerCardsTotal < 17)

{

Card c;

cout << "\nDealer was dealt: " << c.toString() << endl;

dealerCards.push\_back(c); // adds the card to dealer's hand

cout << "\nDealer cards: " << showCards(dealerCards) << endl;

dealerCardsTotal = sumCardValues(dealerCards);

cout << "Dealer total: " << dealerCardsTotal << endl;

}

} while (dealerCardsTotal < 17);

// show the cards for teh dealer and the player

cout << "\nYour cards: " << showCards(playerCards) << " (" << playerCardsTotal << ")" << endl;

cout << "\nDealer cards: " << showCards(dealerCards) << " (" << dealerCardsTotal << ")" << endl;

//who wins?

if (dealerCardsTotal > 21)

{

cout << "\nDealer busted!" << endl;

cash = cash + bet;

}

else if (dealerCardsTotal > playerCardsTotal)

{

cout << "\nDealer wins." << endl;

cash = cash - bet;

}

else if (playerCardsTotal > dealerCardsTotal)

{

cout << "\nYou win!" << endl;

cash = cash + bet;

}

else

{

cout << "\nYou pushed the dealer (tie)." << endl;

}

}

// show the current cash balance

cout << "\nYour current cash balance: $" << cash << endl;

}

Card.h

#pragma once

#include <ctime>

#include <stdlib.h>

#include <string>

using namespace std;

class Card

{

private:

//attributes

char suit;

char face;

short value;

static bool randomizerSeeded; // class-wide variable (one copy in memory in shared by all Card objects)

public:

// Constructor and destructor

Card();

~Card();

// behaviors

string toString();

bool flipAcetoOne();

//accessors and mutators

inline char getSuite()

{

return suit;

}

inline char getFace()

{

return face;

}

inline short getValue()

{

return value;

}

};

Card.cpp

#include "Card.h"

bool Card::randomizerSeeded = false;

Card::Card()

{

//seed the randomizer only once for all card objects

if (randomizerSeeded == false)

{

srand(time(NULL));

randomizerSeeded = true;

}

// create a random number from 3-6 for the card suit

short min = 3;

short max = 6;

suit = rand() % (max - min + 1) + min;

//create a random number from 2 to 14 -- cards are 2 to Ace

min = 2;

max = 14;

short number = rand() % (max - min + 1) + min;

if (number >= 2 && number <= 9)

{

value = number;

face = number + 48; // +48 to achieve proper ascii value

}

else if (number == 10)

{

value = number;

face = 'T';

}

else if (number == 11)

{

value = 10;

face = 'J';

}

else if (number == 12)

{

value = 10;

face = 'Q';

}

else if (number == 13)

{

value = 10;

face = 'K';

}

else if (number == 14)

{

value = 11;

face = 'A';

}

else

{

value = -1;

face = 'E';

}

}

string Card::toString()

{

string output = "";

output += suit;

output = +face;

return output;

}

bool Card::flipAcetoOne()

{

if (value == 11)

{

value = 1;

return true;

}

else

{

return false;

}

}

Card::~Card()

{

}