Project 2
Blackjack

CSC-5
Name: Vincent Lam
Date: 2/11/2023

Table of contents

Introduction

Blackjack is a popular card game played in casinos and online gaming platforms. The game's objective is to beat the dealer by having a hand value of 21 or as close to 21 as possible without going over. The game is played with one or more decks of standard playing cards, and each player, including the dealer, is dealt two cards. The player can hit and take additional cards to improve their hand value or stand and keep their current hand value. The dealer must follow specific rules, such as taking additional cards until they have 17 points. If the player's hand value exceeds 21, they lose the game, and if their hand value is equal to or closer to 21 than the dealer's, they win.

Rules Card Game Works

Step 1: Place a bet amount before starting the game
Step 2: You get drawn two cards out of a deck while the dealer is also drawn two cards, but one is revealed while the other card is hidden.
Step 3: you have the option to hit or stand. Hit means to draw another card. Stand means to keep your current cards.
Step 4: Afterward, when you are done dealing with the cards, the dealer reveals their cards.
Step 5: If your card value exceeds the dealers, you win, but if your card value is less than the dealers, you lose the game and the money you bet.

How to Blackjack

Winning at blackjack requires a combination of luck and strategy. The game's objective is to beat the dealer by having a hand value of 21 or as close to 21 as possible without exceeding it. The hand's value is determined by adding up the values of the individual cards. Here are some tips for improving your chances of winning.

The way I play blackjack is If your card value is 16 or under, you should hit. If your card value is over 17, you should stay. The rule of thumb is high-value cards such as 10s, Jacks, Queens, and Kings are beneficial for the player as they increase the chances of getting a hand value close to 21, while low-value cards such as 2s, 3s, 4s, 5s, and 6s are more beneficial for the dealer.

Development story
When coding the game, a few questions appeared in my brain.

“How do I make a card game?”
“How do I make the card games random?”
“How do I show the statistics of the game?”

To solve this problem, I decided to do hours of research throughout the week and make a lot of trailing and error for my code repeatedly until it worked. After doing some research, I got most of my problems solved, and by going to the lab aids were able to help me solve the bugs in my code and help my code function, and teaching how to use flowcharts for my code as well.

After doing project one, it took a while to figure out how to use checklist two to satisfy the other components of the project, but after doing some tinkering, I could incorporate most of it.

Similarity to the game

My blackjack game does operate similarly to a regular blackjack game.
You have to produce an output greater than the dealer to win.
Also, in a casino-style blackjack game, you must have a set amount of money to bet.
Since this mimics a casino, the game can ask the user for their age before they play so they can be law-abiding citizens.

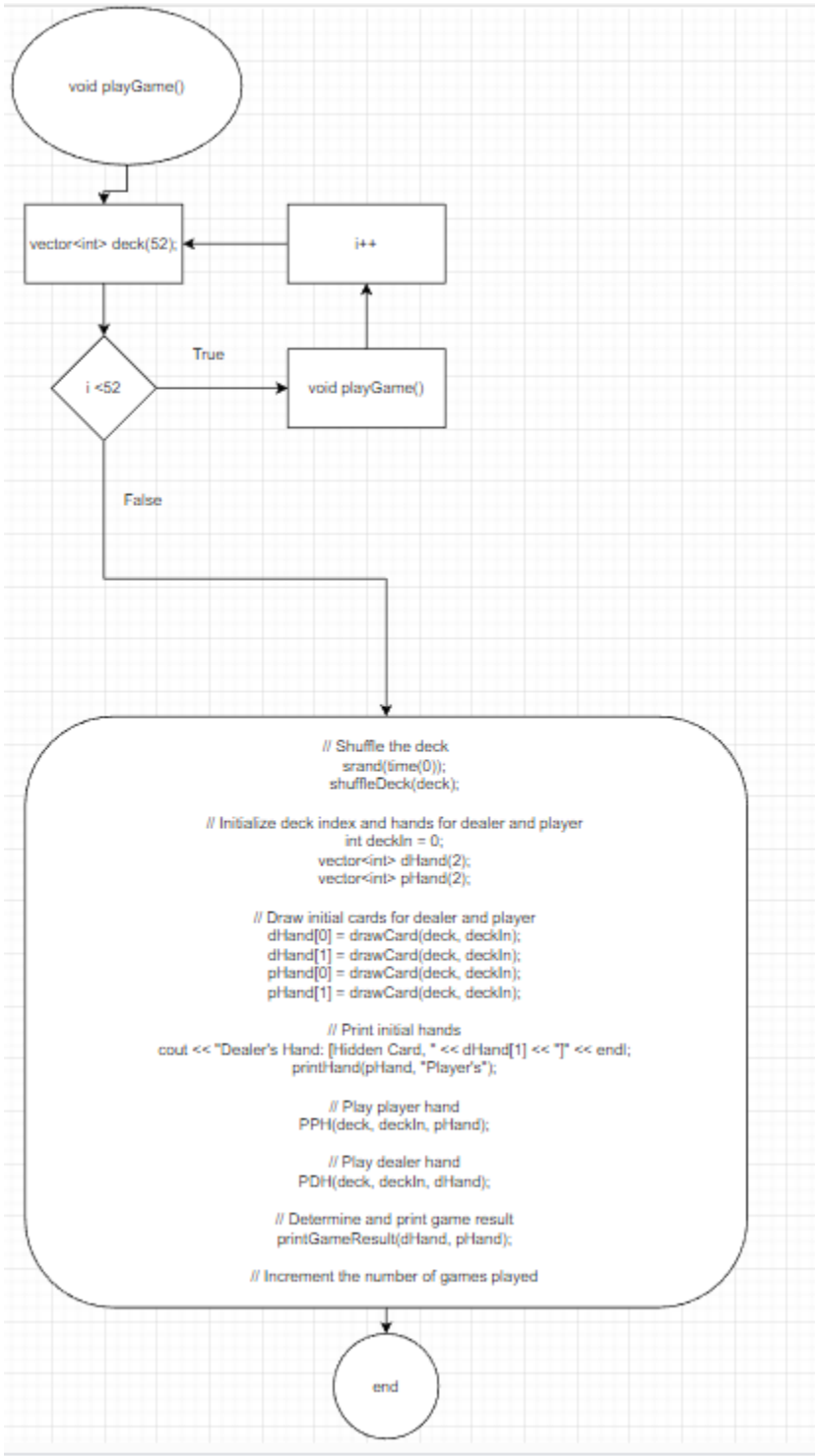
The difference in the game

Right now the difference between my game and a normal blackjack is that it ask the users a lot of questions. It will ask the player their age, it will display the sorted deck, it will tell you the card that you are looking for, and in the index, it will do a message display of your potential earnings if you play. It will display the rule sets, and it also tests the user if they can count up to 21. The main similarity is the blackjack game itself.

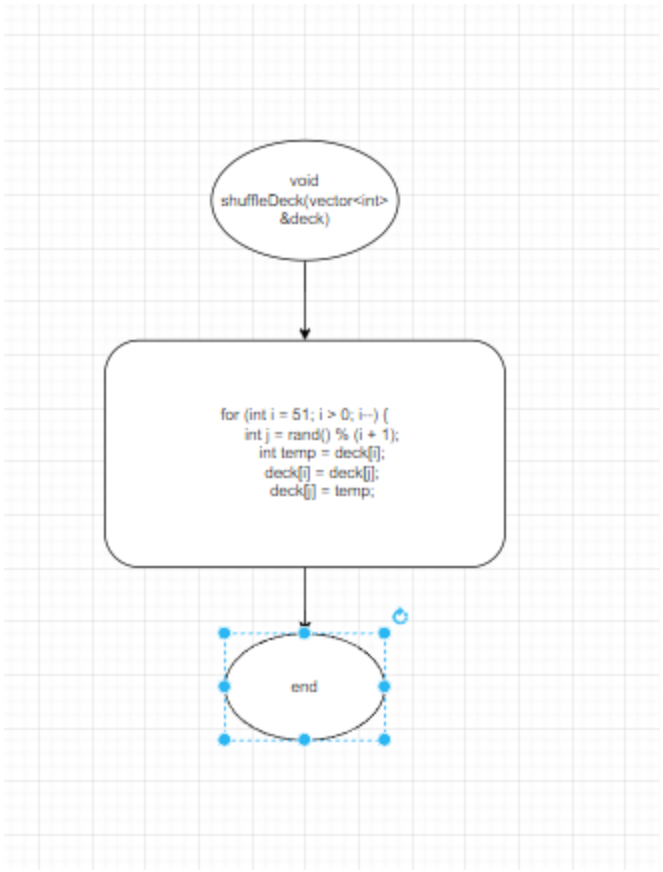
Pseudo code

```
INITIALIZE deck of cards
DISAPLY rules to the player
DEAL two cards to player and two cards to dealer

DISPLAY player's cards and one of dealer's cards

WHILE player's hand value is less than 21
  PROMPT player to hit or stand
  IF player chooses to hit
    DEAL another card to player
  END IF
END WHILE

IF player's hand value is over 21
  DECLARE dealer as the winner
  END GAME
END IF

WHILE dealer's hand value is less than 17
  DEAL another card to dealer
END WHILE

IF dealer's hand value is over 21
  DECLARE player as the winner
ELSE
  COMPARE player and dealer's hand values
    IF player's hand value is higher
      DECLARE player as the winner
    ELSE
      DECLARE dealer as the winner
    END IF
END IF

DISPLAY statistics
```

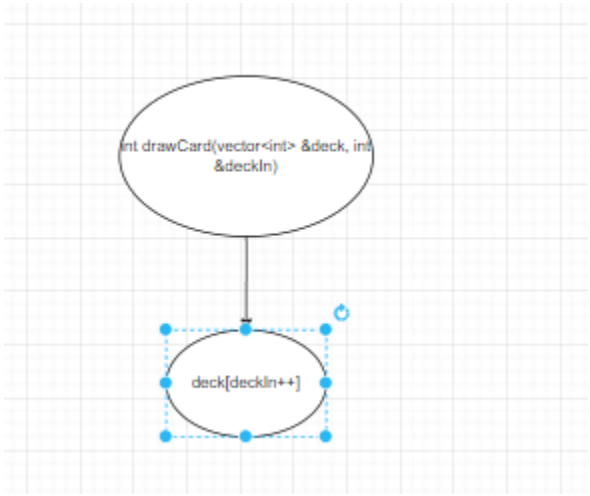
Flow Chart



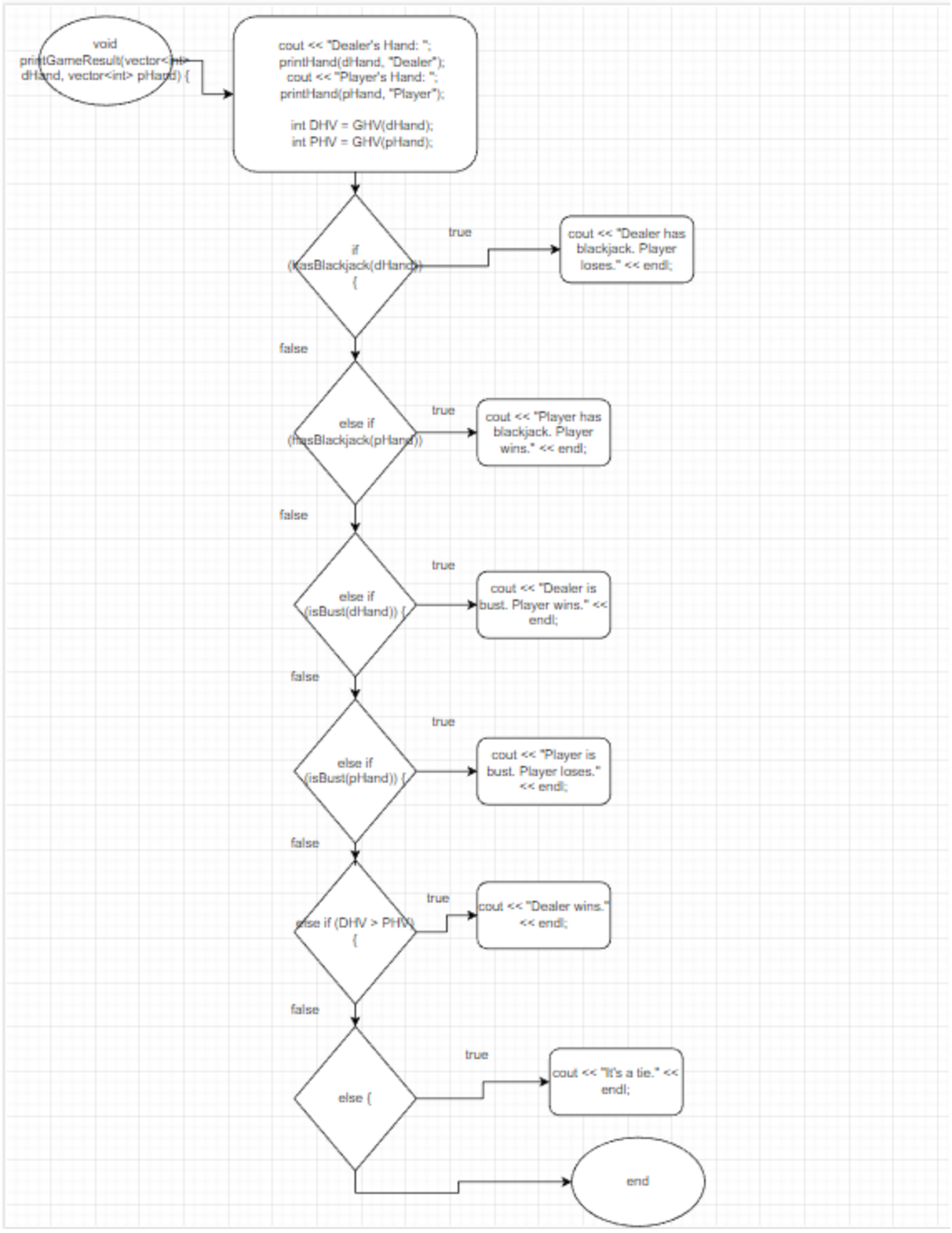
The void for the blackjack game.



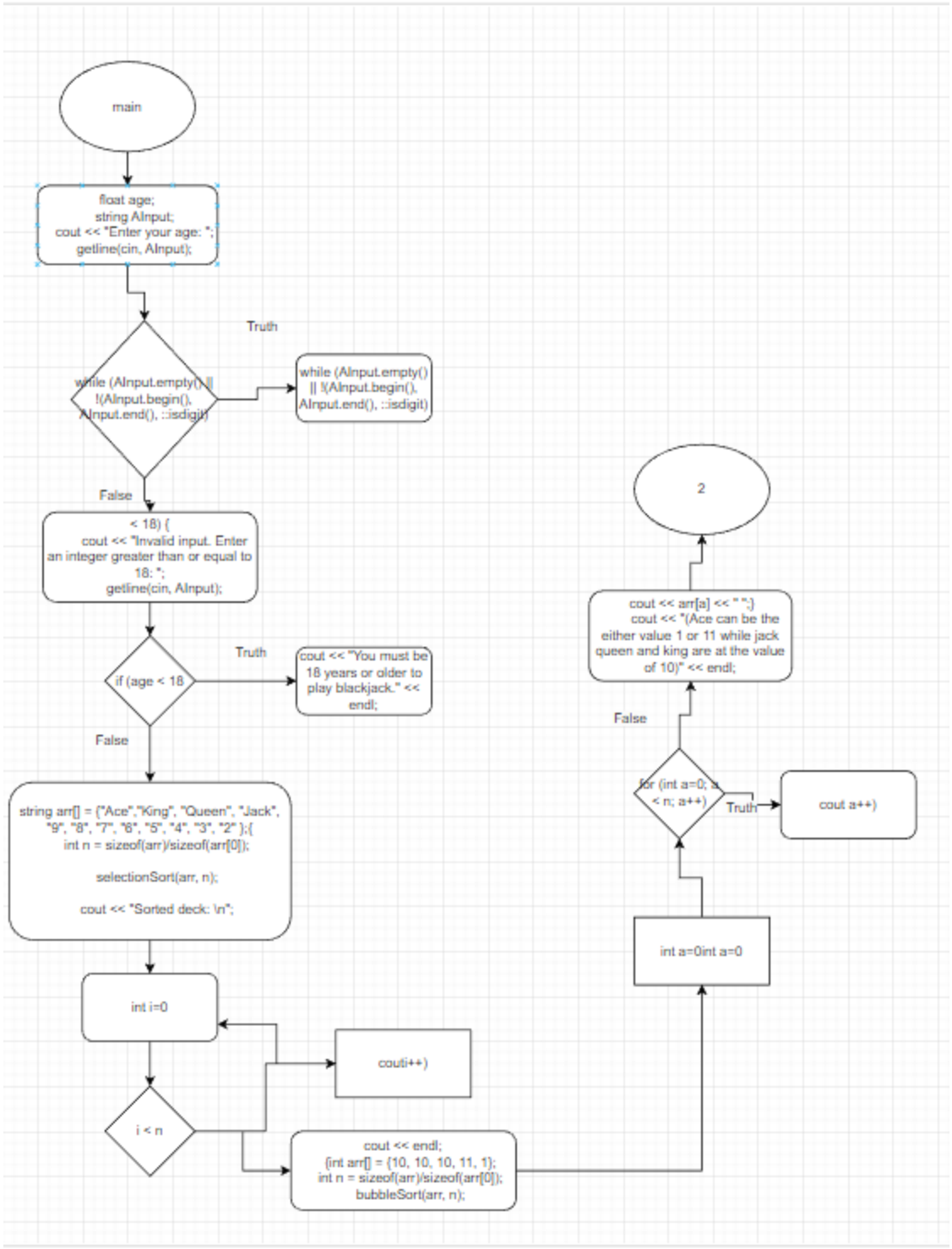
Shuffle deck of cards using Fisher-Yates shuffle algorithm



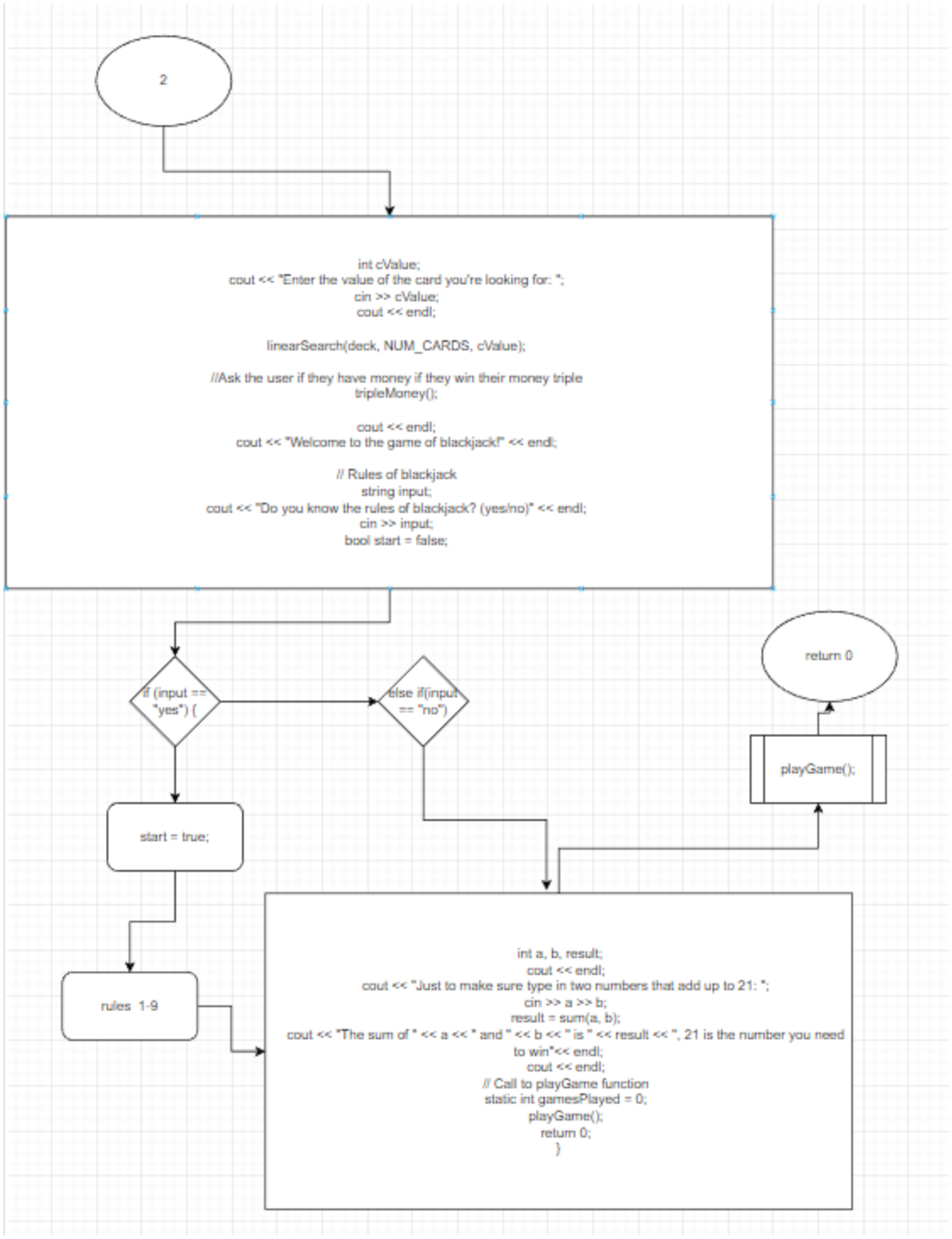
Draw a card from the deck



// Print the result of the game



//Ask for age
//print the card selection sort using selection sort and bubble sort



// ask the user for for the card value in the deck
//ask the user if they want to know the rules
// test the user if they can find two numbers that can add up to 21 if not it will display that you need 21 to win the game
//play the actual blackjack game
//the game ends

Project Checklist

Chapter	Section	Topic	Where in line#	Pts	Notes
2	3	Libraries	#1-7	5	<iostream> <vector> <ctime> <cstdlib> <string> <cmath> <map>
	6	integers	#34 #35 #36 #63-65 #79 #81	1	deckIn CV n i, j, HV dHand pHand a, b, result
	7	Characters	#107	1	HorS
	8	Strings	#77 #224 #277	1	playerN Ainput input
	9	Floats No Doubles	#214 #217 #223	1	Money Triple age
	10	Bools	#256	1	
	15	Comments 20%+	#22,29,33,38 ,44,48,51,54, 57,61,71,76, 88,93,98,105 ,120,133,161 ,168,186,199 ,212,222,266 ,272,305	2	
3	5	Type casting	#229	1	int age =

			#262 #265 #301		stoi(AInput) cin >> cValue linearSearch(deck, NUM_CARDS, cValue) cin >> a >> b
	7	Formatting output	#46 #47	1	cout << "Dealer's Hand: [Hidden Card, " << dHand[1] << "]" << endl; (output formatting with '<<') printHand(pHand, "Player
	8	string	#225 #231 #239	1	string AInput; getline(cin, AInput); string arr[] = {"Ace","King", "Queen", "Jack", "9", "8", "7", "6", "5", "4", "3", "2" };
	9	Math Library	#6 #219	1	include <cmath> triple = pow(money, three);
4	2	if	#82	1	if (i <

					hand.size() - 1) {
	4	If else	#144-157	1	if (hasBlackjack(dHand)) { cout << "Dealer has blackjack. Player loses." << endl; } else if (hasBlackjack(pHand)) { cout << "Player has blackjack. Player wins." << endl;
	5	nesting	#107-119	1	while (!isBust(pHand) && !hasBlackjack(pHand)) if else{
	6	If else if	#278-280	1	if (input == "yes") { start = true; } else if(input == "no") {
	8	Logical operators	#109	1	while (!isBust(pHand) && !hasBlackjack(pHand)) {
	11	(Validating user input)	#224-237	1	getline(cin, AInput); if (age < 18) {

	13	Conditional operator		1	
	14	switch		1	
5	1	Increment/Decrement		1	
	2	while	#100	1	while (GHV(dHand) < 17) {
	5	Do while		1	
	6	For while		1	
	11	Files input/output both		2	

6	3	Function Prototypes	#11
			#12
			#13
			#14
				4	void shuffleDeck(vector<int> &deck);
					
					int drawCard(vector<int> &deck, int &deckIndex);
					
					void printHand(vector<int> hand, string playerName);
					
					bool hasBlackjack (vector<int> hand);

	5	Pass by Value	#162-166 #29-303	4	For counting to 21
	8	return	#18	4	int getHandValu

					e(vector<int> hand);
	9	returning boolean	#14 #15	4	bool hasBlackjack (vector<int> hand); bool isBust(vector <int> hand);
	11	static variables	#306	4	static int gamesPlayed = 0;
	12	defaulted arguments		4	

	13	pass by reference	#11
			#12
			#16
			#17
				4	void shuffleDeck(v ector<int> &deck);
					
					int drawCard(ve ctor<int> &deck, int &deckIn);
					
					void PDH(vector<i nt> &deck, int &deckIn, vector<int> &dHand);
					
					void PPH(vector<i nt> &deck, int &deckIn, vector<int> &pHand);

	14	overloading		5	
	15	exit() function	#235	4	exit(1);
7	1 to 6	Single Dimensioned Arrays	#201 #202	3	int deck[NUM_CARDS] int arr[]
	7	Parallel Arrays		2	
	8	Single Dimensioned as Function Arguments	#187-198 #250 #202	2	bubbleSort(arr, n); linearSearch(deck, NUM_CARDS, cValue);
	9	2 Dimensioned Arrays		2	

	12	STL Vectors	#24
			#26
			#35
			#36
				2	vector<int> deck(52);
					deck[i] = i % 13 + 1;
					vector<int> dHand(2);
					vector<int> pHand(2);

		Passing Arrays to and from Functions	#250 #264	5	bubbleSort(arr, n); linearSearch(deck, NUM_CARDS, cValue);

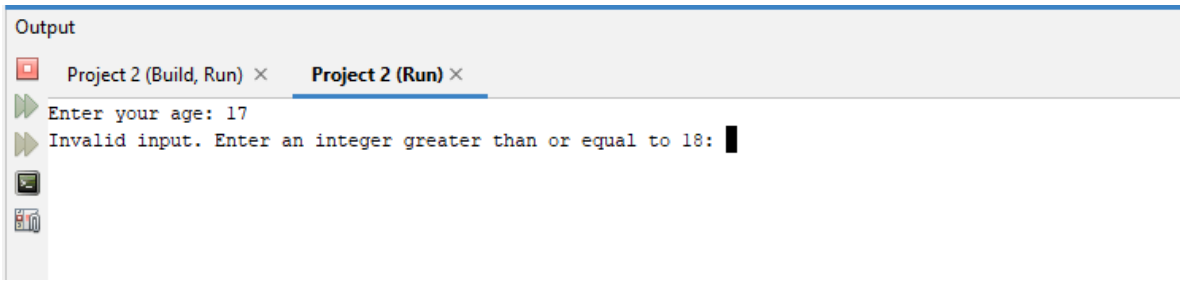
		Passing Vectors to and from Functions	#24
			#35
			#36
				5	vector<int> deck(52);
					vector<int> dHand(2);
					vector<int> pHand(2);

8	3	Bubble Sort	#187-198	4	bubbleSort
	3	Selection	#169-184	4	selectionSort

		Sort			
	1	Linear or Binary Search	#200-211	4	void linearSearch

Proof of a Working Product

```
Enter your age: 20
Sorted deck:
2 3 4 5 6 7 8 9 Jack Queen King Ace
1 10 10 10 11 (Ace can be the either value 1 or 11 while jack queen and king are at the value of 10)
```




```
Enter the value of the card you're looking for: 4

The value 4 was found at index 2 in the deck.
```

```
How much money do you have?
5
If you invest $5 right now, you can earn your investment 3 time over to $125. WIN BIG PLAY NOW
```

```
Welcome to the game of blackjack!
Do you know the rules of blackjack? (yes/no)
no
Step 1: The game is played with a standard deck of 52 cards.
Step 2: The objective of the game is to beat the dealer by having a hand
        value of 21 or as close to 21 as possible without going over.
Step 3: Each player, including the dealer, receives two cards at the start of the
        game. One of the dealer's cards is face up, while the other is face down.
Step 4: The value of a player's hand is determined by adding up the point values of
        the individual cards. Face cards (Kings, Queens, and Jacks) are worth 10 points each,
        Aces can be worth either 1 or 11 points, and all other cards are worth their face value.
Step 5: Players take turns choosing to hit (take another card) or stand (keep their current hand).
Step 6: If a player's hand value exceeds 21, they bust and lose the game.
Step 7: If the dealer busts, all remaining players win.
Step 8: If the dealer does not bust, then the player with the highest hand value that is less than
        or equal to 21 wins.
Step 9: If a player and the dealer have the same hand value, the game is a push (tie) and the player
        does not win or lose.
```

```
Welcome to the game of blackjack!
Do you know the rules of blackjack? (yes/no)
yes
```

Output

Project 2 (Build, Run) × Project 2 (Run) ×

```
Enter your age: 20
Sorted deck:
2 3 4 5 6 7 8 9 Jack Queen King Ace
1 10 10 10 11 (Ace can be the either value 1 or 11 while jack queen and king are at the value of 10)

Enter the value of the card you're looking for: 4

The value 4 was found at index 2 in the deck.
How much money do you have?
9
If you invest $9 right now, you can earn your investment 3 time over to $729. WIN BIG PLAY NOW

Welcome to the game of blackjack!
Do you know the rules of blackjack? (yes/no)
yes

Just to make sure type in two numbers that add up to 21: 30 -9
The sum of 30 and -9 is 21, 21 is the number you need to win
```

```
Just to make sure type in two numbers that add up to 21: 10 11
The sum of 10 and 11 is 21, 21 is the number you need to win
```

Output

Project 2 (Build, Run) × Project 2 (Run) ×

```
Enter your age: 23
Sorted deck:
2 3 4 5 6 7 8 9 Jack Queen King Ace
1 10 10 10 11 (Ace can be the either value 1 or 11 while jack queen and king are at the value of 10)

Enter the value of the card you're looking for: 11

The value 11 was found at index 12 in the deck.
How much money do you have?
3
If you invest $3 right now, you can earn your investment 3 time over to $27. WIN BIG PLAY NOW

Welcome to the game of blackjack!
Do you know the rules of blackjack? (yes/no)
yes

Just to make sure type in two numbers that add up to 21: 20 1
The sum of 20 and 1 is 21, 21 is the number you need to win

Dealer's Hand: [Hidden Card, 2]
Player's Hand: [7, 4]
Do you want to hit or stand? (h/s): s
Dealer's Hand: Dealer Hand: [3, 2, 5, 7]
Player's Hand: Player Hand: [7, 4]
Dealer wins.
```

Program

//There is about 309 lines of code in this program

```
#include <iostream>
#include <vector>
#include <ctime>
#include <cstdlib>
#include <string>
#include <cmath>
#include <map>
using namespace std;

// Function prototypes
void shuffleDeck(vector<int> &deck);
int drawCard(vector<int> &deck, int &deckIn);
void printHand(vector<int> hand, string playerN);
bool hasBlackjack(vector<int> hand);
bool isBust(vector<int> hand);
void PDH(vector<int> &deck, int &deckIn, vector<int> &dHand);
void PPH(vector<int> &deck, int &deckIn, vector<int> &pHand);
int GHV(vector<int> hand);
void printGameResult(vector<int> dHand, vector<int> pHand);
static int gamesPlayed = 0;


void playGame() {
    // Initialize deck of cards
    vector<int> deck(52);
    for (int i = 0; i < 52; i++) {
        deck[i] = i % 11 + 1;
    }

    // Shuffle the deck
    srand(time(0));
    shuffleDeck(deck);

    // Initialize deck index and hands for dealer and player
    int deckIn = 0;
    vector<int> dHand(2);
    vector<int> pHand(2);

    // Draw initial cards for dealer and player
    dHand[0] = drawCard(deck, deckIn);
    dHand[1] = drawCard(deck, deckIn);
```

```
    pHand[0] = drawCard(deck, deckIn);
    pHand[1] = drawCard(deck, deckIn);

    // Print initial hands
    cout << "Dealer's Hand: [Hidden Card, " << dHand[1] << "]" << endl;
    printHand(pHand, "Player's");

    // Play player hand
    PPH(deck, deckIn, pHand);

    // Play dealer hand
    PDH(deck, deckIn, dHand);

    // Determine and print game result
    printGameResult(dHand, pHand);

    // Increment the number of games played

}

// Shuffle deck of cards using Fisher-Yates shuffle algorithm
void shuffleDeck(vector<int> &deck) {
    for (int i = 51; i > 0; i--) {
        int j = rand() % (i + 1);
        int temp = deck[i];
        deck[i] = deck[j];
        deck[j] = temp;
    }
}

// Draw a card from the deck
int drawCard(vector<int> &deck, int &deckIn) {
    return deck[deckIn++];
}

// Print player's or dealer's hand
void printHand(vector<int> hand, string playerN) {
    cout << playerN << " Hand: [";
    for (int i = 0; i < hand.size(); i++) {
        cout << hand[i];
        if (i < hand.size() - 1) {
            cout << ", ";
        }
    }
```

```
    cout << "]" << endl;
}

// Check if player has blackjack (21 points with only two cards)
bool hasBlackjack(vector<int> hand) {
    return hand.size() == 2 && GHV(hand) == 21;
}

// Check if player is bust (more than 21 points)
bool isBust(vector<int> hand) {
    return GHV(hand) > 21;
}

// Play dealer's hand
void PDH(vector<int> &deck, int &deckIn, vector<int> &dHand) {
    while (GHV(dHand) < 17) {
        dHand.push_back(drawCard(deck, deckIn));
    }
}

// Play player's hand
void PPH(vector<int> &deck, int &deckIn, vector<int> &pHand) {
    char HorS;
    while (!isBust(pHand) && !hasBlackjack(pHand)) {
        cout << "Do you want to hit or stand? (h/s): ";
        cin >> HorS;
        if (HorS == 'h') {
            pHand.push_back(drawCard(deck, deckIn));
            printHand(pHand, "Player's");
        } else {
            return;
        }
    }
}

// Get the value of a hand
int GHV(vector<int> hand) {
    int HV = 0;
    for (int i = 0; i < hand.size(); i++) {
        int cValue = hand[i];
        if (cValue > 10) {
            cValue = 10;
        }
        HV += cValue;
```

```
    }
    return HV;
}

// Print the result of the game
void printGameResult(vector<int> dHand, vector<int> pHand) {
    cout << "Dealer's Hand: ";
    printHand(dHand, "Dealer");
    cout << "Player's Hand: ";
    printHand(pHand, "Player");

    int DHV = GHV(dHand);
    int PHV = GHV(pHand);

    if (hasBlackjack(dHand)) {
        cout << "Dealer has blackjack. Player loses." << endl;
    } else if (hasBlackjack(pHand)) {
        cout << "Player has blackjack. Player wins." << endl;
    } else if (isBust(dHand)) {
        cout << "Dealer is bust. Player wins." << endl;
    } else if (isBust(pHand)) {
        cout << "Player is bust. Player loses." << endl;
    } else if (DHV > PHV) {
        cout << "Dealer wins." << endl;
    } else if (DHV < PHV) {
        cout << "Player wins." << endl;
    }
     else {
        cout << "It's a tie." << endl;
    }
}

//pass by reference
int sum(int a, int b) {
    a = a;
    b = b;
    return a + b;
}

//selectionSort
map<string, int> CV = {{"2", 2}, {"3", 3}, {"4", 4}, {"5", 5}, {"6", 6}, {"7", 7}, {"8", 8}, {"9", 9}, {"10",
10}, {"Jack", 10}, {"Queen", 10}, {"King", 10}, {"Ace", 11}};

void selectionSort(string arr[], int n) {
```

```
    int i, j, mIndex;
    string temp;
    for (i = 0; i < n-1; i++) {
        mIndex = i;
        for (j = i+1; j < n; j++) {
            if (CV[arr[j]] < CV[arr[mIndex]])
                mIndex = j;
        }
        temp = arr[mIndex];
        arr[mIndex] = arr[i];
        arr[i] = temp;
    }
}

//bubble sort
void bubbleSort(int arr[], int n) {
    int a, b, temp;
    for (a = 0; a < n-1; a++) {
        for (b = 0; b < n-a-1; b++) {
            if (arr[b] > arr[b+1]) {
                temp = arr[b];
                arr[b] = arr[b+1];
                arr[b+1] = temp;
            }
        }
    }
}
//linearSearch
const int NUM_CARDS = 52;
int deck[NUM_CARDS] = {2, 3, 4, 5, 6, 7, 8, 9, 10, 10, 10, 10, 11};
void linearSearch(int arr[], int n, int x) {
    int i;
    for (i = 0; i < n; i++) {
        if (arr[i] == x) {
            cout << "The value " << x << " was found at index " << i << " in the deck." << endl;
            return;
        }
    }
    cout << "The value " << x << " was not found in the deck." << endl;
}
// triple money
void tripleMoney() {
    float money;
    float three = 3;
```

```
    cout << "How much money do you have?" << endl;
    cin >> money;
    float triple = pow(money, three);
    cout << "If you invest $" << money << " right now, you can earn your investment 3 time over
to $" << triple << ". WIN BIG PLAY NOW" << endl;
}
int main() {
    //Ask for age
    float age;
    string AInput;
    cout << "Enter your age: ";
    getline(cin, AInput);

    while (AInput.empty() || !(AInput.begin(), AInput.end(), ::isdigit) || (age = stoi(AInput)) < 18) {
        cout << "Invalid input. Enter an integer greater than or equal to 18: ";
        getline(cin, AInput);
    }

    if (age < 18) {
        cout << "You must be 18 years or older to play blackjack." << endl;
        exit(1);
    }

    string arr[] = {"Ace","King", "Queen", "Jack", "9", "8", "7", "6", "5", "4", "3", "2" };{
    int n = sizeof(arr)/sizeof(arr[0]);

    selectionSort(arr, n);

    cout << "Sorted deck: \n";
    for (int i=0; i < n; i++)
        cout << arr[i] << " ";
                                                                    }
    cout << endl;
    {int arr[] = {10, 10, 10, 11, 1};
    int n = sizeof(arr)/sizeof(arr[0]);
    bubbleSort(arr, n);


    for (int a=0; a < n; a++)
        cout << arr[a] << " ";}
        cout << "(Ace can be the either value 1 or 11 while jack queen and king are at the value of
10)" << endl;

    cout << endl;
```

```
    int cValue;
    cout << "Enter the value of the card you're looking for: ";
    cin >> cValue;
    cout << endl;

    linearSearch(deck, NUM_CARDS, cValue);

    //Ask the user if they have money if they win their money triple
    tripleMoney();

    cout << endl;
    cout << "Welcome to the game of blackjack!" << endl;

    // Rules of blackjack
    string input;
    cout << "Do you know the rules of blackjack? (yes/no)" << endl;
    cin >> input;
    bool start = false;
    if (input == "yes") {
        start = true;
    } else if(input == "no") {
        cout << "Step 1: The game is played with a standard deck of 52 cards." << endl;
        cout << "Step 2: The objective of the game is to beat the dealer by having a hand" << endl;
        cout << "        value of 21 or as close to 21 as possible without going over." << endl;
        cout << "Step 3: Each player, including the dealer, receives two cards at the start of the" <<
endl;
        cout << "        game. One of the dealer's cards is face up, while the other is face down. " <<
endl;
        cout << "Step 4: The value of a player's hand is determined by adding up the point values
of" << endl;
        cout << "        the individual cards. Face cards (Kings, Queens, and Jacks) are worth 10
points each," << endl;
        cout << "        Aces can be worth either 1 or 11 points, and all other cards are worth their
face value." << endl;
        cout << "Step 5: Players take turns choosing to hit (take another card) or stand (keep their
current hand)." << endl;
        cout << "Step 6: If a player's hand value exceeds 21, they bust and lose the game." <<
endl;
        cout << "Step 7: If the dealer busts, all remaining players win." << endl;
        cout << "Step 8: If the dealer does not bust, then the player with the highest hand value
that is less than" << endl;
        cout << "        or equal to 21 wins." << endl;
```

```
        cout << "Step 9: If a player and the dealer have the same hand value, the game is a push
(tie) and the player" << endl;
        cout << "        does not win or lose." << endl;
        cout << endl;
  }
    int a, b, result;
    cout << endl;
    cout << "Just to make sure type in two numbers that add up to 21: ";
    cin >> a >> b;
    result = sum(a, b);
    cout << "The sum of " << a << " and " << b << " is " << result << ", 21 is the number you need
to win"<< endl;
    cout << endl;

    // Call to playGame function
    playGame();
    return 0;
}
```