Project 1

Five Card Poker

Joshua Konechy
CIS-17c (40369)
Date: 4/26/2020
Prof. Mark Lehr

Introduction

I am coding this project to demonstrate the use of several functions, algorithms, and containers of the STL library. To do this I have chosen the game of poker, specifically a version called Five Card, to demonstrate the use of the STL functions. Altogether, the time sent on this project was probably around twenty two hours spent on this project over the course of about a week and a half. It was approximately four hours of research, four hours of initial implementation, ten hours of adding the STL components, and about four hours of testing and bug squashing. The project uses one struct to house the card types and clocks in at about 876 lines of code (906 with Comments). The Project is located on my public Github account, in a Public Repo called *Project1_FiveCardPoker.*

Summary

The Concept of the game was to be able to play a hand of five card poker, either against yourself or the computer and then to win chips accordingly. After, the player could continue to play a round and have their chips roll over to the next round. When it came to version control, the three major sections I chose to section off were, the game without STL containers, the game with STL containers, and the game with an AI option.

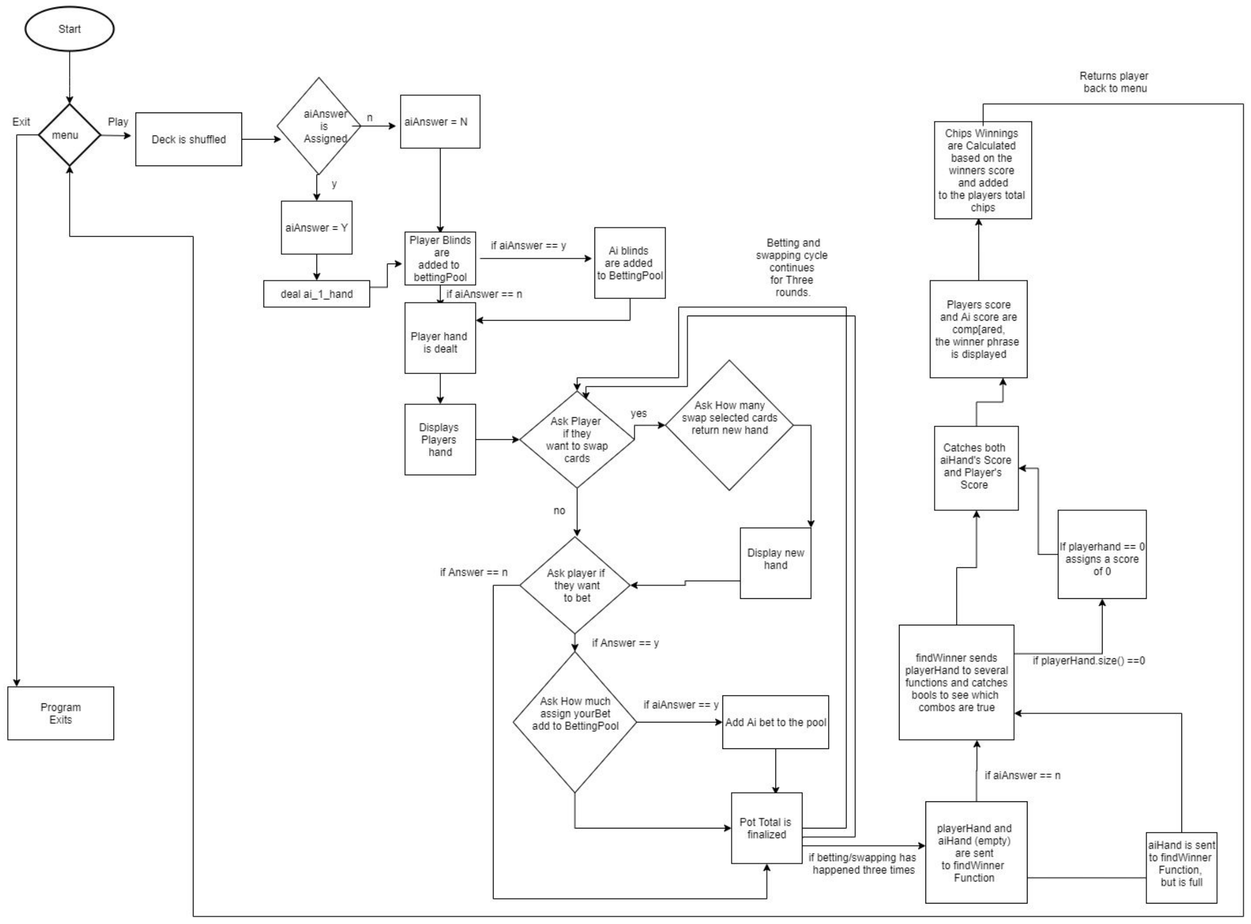
Game Rules

The game Rules are fairly simple. The player first answers if they would like to play by themselves or with a computer player.Then the player enters the game and enters the amount of chips they would like to play with. After the chips are rewarded, the game takes either the small or big blind (An amount taken each round to assure the the winner will always win a minimum amount of chips). Then, the player's hand is shown to them and they are asked if they would like to change any amount of cards. If they choose yes, the cards are swapped with the cards they chose, if not, the game moves onto betting. The player can bet any amount of chips they have to the pot and they Ai will do the same (if the player is playing the computer). This will occur for three rounds and after the three rounds are complete the game will calculate a winner and adjust the players chips accordingly. Then the player can return to the main menu to play again or exit the program. Their winnings or losses will stay with them through the rounds.

Description

The main organization of the project is as follows. The Card Struct is created at the beginning of the project and is used to hold the values of the card and the card type. After, the card is sent to a stack of objects card where it creates the deck, and also the empty list of the player hand and the ai hand. Then the deck is sent to a function where the deck is filled by a map that generates all the possible cards and then the function pushes all the elements of the map onto a stack after shuffling the cards. The Newly shuffled deck is then sent out to main. Following that, the program draws five cards for each player off the random deck and then proceeds through the betting and swapping rounds, pulling cards from the deck as needed. Then after betting, the hands are sent to a function to calculate the card combinations and a score is returned. The Score is compared to the other players and a winner is determined. Chips are rewarded accordingly.

FlowChart


Start
Exit
menu
Play
Deck is shuffled
aiAnswer is Assigned
n
y
aiAnswer = N
aiAnswer = Y
deal ai_1_hand
Player Blinds are added to bettingPool
if aiAnswer == y
Ai blinds are added to BettingPool
if aiAnswer == n
Player hand is dealt
Displays Players hand
Ask Player if they want to swap cards
yes
Ask How many swap selected cards return new hand
no
Display new hand
Betting and swapping cycle continues for Three rounds.
if Answer == n
Ask player if they want to bet
if Answer == y
Ask How much assign yourBet add to BettingPool
if aiAnswer == y
Add Ai bet to the pool
Pot Total is finalized
if betting/swapping has happened three times
playerHand and aiHand (empty) are sent to findWinner Function
aiHand is sent to findWinner Function, but is full
if aiAnswer == n
findWinner sends playerHand to several functions and catches bools to see which combos are true
if playerHand.size() ==0
If playerhand == 0 assigns a score of 0
Catches both aiHand's Score and Player's Score
Players score and Ai score are comp[ared, the winner phrase is displayed
Chips Winnings are Calculated based on the winners score and added to the players total chips
Returns player back to menu
Program Exits

Pseudo Code

```
Initialize.

    Bool stop is set to false.

    While false is not true, run main.

        Deck, playerHand, and ai_1_hand are created.
```

The Deck is shuffled and returned.

Menu prompts, the player can play or exit.

If Play, the player is prompted to add Ai
if yes, ai_1_hand is dealt, else it is not.

Player is asked how many chips they want.

Chips are then added to the player's amount.
if aiAnswer is true, Ai player gets the same amount.

The blinds are minused from the players chips based on the number of rounds.

The Player is dealt a hand and it is displayed.

The player is then prompt to see if they want to swap cards and or bet; this is done three times.

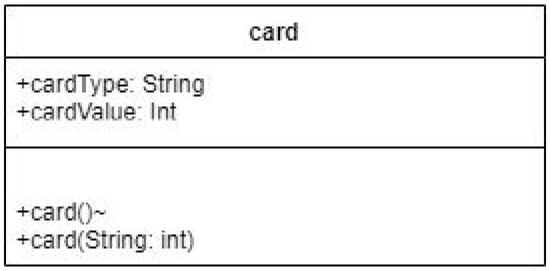
If aiAnswer is true, the Ai bets a random amount each round.

After betting 3 times, the player hand is sent to findWinner to be score;
if aiAnswer is Yes, the ai_1_hand is also scored, else it is set to 0.

After the scores are sent to a void display to print the winners phrase.

Depending on the scores, the players and ai Chips are altered to adjust for the winning status.
The Players and Ai hands are cleared and the player is returned to the menu.

Classes(UML)



Functions

int findWinner(list<card> &playerHand);

// Evaluate whether there is a winner in the hand
// Is set the parameter Playerhand by reference to alter the empty hand, and pass back a score
// This function is used for both the player and AI hand

float blinds(float originalBettingPool, int &rounds);

// Adds the blinds to the pot and keeps track of the rounds
// orginalBettingPool is the pot before the player or AI bets
//Rounds is used to switch the blinds between players

float addAiBlinds(float newBettingPool, int &rounds);

// Adds the blinds to the pot and keeps track of the rounds for the ai
// newBettingPool is the pot after the player blinds and only if AI is active
//Rounds is used to switch the blinds between players

float aiBetting(float newBettingPool, float aiChips);

//A function that is used to randomly generate an amount to add to the pot for the AI's "Bets"
//NewBettingPool is the pot that is being sent each round, if the AI is active. The AiChips is the amount of chips the ai has to make sure he doesn't bet more than he has.

bool isPair(list<card> playerHand);

// This Bool function is checking the hand that is sent to it to see if the cards make a pair. A bool is returned to the findWinner function
//playerhand is NOT sent by reference because we are not altering it

bool isTwoPair(list<card> playerHand);
// This Bool function is checking the hand that is sent to it to see if the cards make a two pair. A bool is returned to the findWinner function
//playerhand is NOT sent by reference because we are not altering it
// Alist is used to keep track of the amount of times that cards are matching (but no the same card)

bool isThreeOfAKind(list<card> playerHand);
//This Bool function is checking the hand that is sent to it to see if the cards make a three of a kind. A bool is returned to the findWinner function
//playerhand is NOT sent by reference because we are not altering it
//A simple counter is used this time to count the amount of times cards match but are not the same

bool isIsFourOfaKind(list<card> playerHand);
//This Bool function is checking the hand that is sent to it to see if the cards make a four of a kind. A bool is returned to the findWinner function
//playerhand is NOT sent by reference because we are not altering it
//A simple counter is used this time to count the amount of times cards match but are not the same

bool isFullHouse(list<card> playerHand);
//This Bool function is checking the hand that is sent to it to see if the cards make a three of a kind. A bool is returned to the findWinner function
//playerhand is NOT sent by reference because we are not altering it
//A simple counter is used with two bools to check if this time to count the amount of times cards match but are not the same

bool isFlush(list<card> playerHand);
// Bool functions to check the card to see if a flush is present in the hand
// is pasted player hand to compare it to temp

bool isStraight(list<card> playerHand);
//Function checks the hand sent to it and check to see if a straight is true
//Para: passes copy of the playerHand to function to see if they have a straight

card drawCard(stack<card>& deck);
//Function used to pop a card off the top of the stack in the swap card function

//The function is sent the stack deck by reference

stack<card> shuffleDeck(stack<card> &deck);

//This function creates an unordered map called new deck and fills it with all 52 cards in order. Then using an iterator, it puts the elements of the map into the stack that was sent.
// it does this in a random order to "Shuffle" the deck
// The function is sent the empty stack of type card, by reference

void displayWinnerPhrase(int result, int aiRoundResult, char aiAnswer);

//This function just spits out a phrase to congratulate the player if they win, lose, or tie
//This function is sent the players result, the airesults, and if the Ai is active

void dealHand(list<card> &playerHand, stack<card> &deck);

// Deals five random cards to player
// is sent the empty hand list and the shuffled deck stack

void dealAIHand(list<card> &ai_1_Hand, stack<card> &deck);

// Deals five random cards to ai
// is sent the empty hand list and the shuffled deck stack

void swapCards(list<card> &playerHand, stack<card> &deck);

// Allows player to swap cards within their hand
// is sent the full player hand and the shuffled deck minus the players cards

void displayHand(list<card> playerHand);

//displays the players hand
// is sent the list that is players hand

CheckOff Sheet

1)Container Classes

I) Sequences

a.) List - Used before main and in several functions to hold a temporary list of items or used to hold the players cards.

II) Associative Containers

a.) Set - Used in the findStraight functions as comparison to the playerhand

b.) Map - Used in the ShuffleDeck function to hold the the cards in order before they are shuffled and assigned to a stack

III) Container adapters

a.) Stack - Used in ShuffleDeck to hold the shuffle cards sent from the map.

2)Iterators

I) Concepts (Describe the iterators utilized for each Container)

a.) Trival Iterator

b.) Input Iterator - Used in input operations where each value is read once and then incremented.

c.) Output Iterator - Used in Input operations where each value is read once and then incremented but in reverse

d.) Forward Iterator - Used in Input and Output Iterators, allows an element to be accessed and modified.

e.) Bidirectional Iterator - Can be used in a list, map, set. Similar to forward Iterators but can also move backwards.

f.) Random Access Iterator - used in vectors and deques, Picks a random element in the containers.

**In my code I believe I used a Bidirectional Iterator to move through the list or set that I used to compare the playerHand to in the functions isStraight and isFlush

3)Algorithms

I) Non-mutating

a.) Find - Used in isFlush to locate the first suit of the hand to compare.

II) Mutating

a.) emplace - Used in ShuffleDeck to place the selected card on top of the map

III) Organization

a.) Sort - Used in isStraight function to sort the playerHand from lowest to highest

Sample Input/Output

```
 /==============\
/                \
  Five Card Poker
\                /
 \==============/

1. Play
2. Exit

Choose your option: 1
Would you like to add another Player? ( AI )
y
```

```
So, another player has entered the game.

Your Total Chips are: 0
You need Chips to Play, How many would you like? (Your Chip Limit is 1000)
1000
Ok, your buy in is 1000
You're in the Big Blind.
Your total after blinds is 975
Okay, the Pot is 35 after the Blinds

1 - 8 of Clubs
2 - 12 of Diamonds
3 - 11 of Hearts
4 - 10 of Hearts
5 - 10 of Diamonds

Do you want to change cards? (y/n): y
Would you like to swap card number 1?
y
Would you like to swap card number 2?
n
Would you like to swap card number 3?
n
Would you like to swap card number 4?
n
Would you like to swap card number 5?
y

Your new cards are:

1 - 12 of Diamonds
2 - 11 of Hearts
3 - 10 of Hearts
4 - 10 of Diamonds
5 - 5 of Diamonds
Would you like to bet? (Y/N)
y
How much would you like to bet?
200
Okay, Pot is Right. The Pot is 235
Round 2: would you like to change your hand again?
n
Would you like to bet?
y
How much would you like to bet?
200
Okay, Pot is Right. The New Pot is 465
Round 3: Final round, would you like to change any cards?
```

```
n
Final Bet?
n
Okay, Pot is Right. The Pot is 717
You are the Winner!
You have a Pair!
Your New chip Amount is 1292

Hit return to return to menu...

 /==============\
/                \
  Five Card Poker
\                /
 \==============/

1. Play
2. Exit

Choose your option: 1
Would you like to add another Player? ( AI )
n
Your Total Chips are: 1292
You need Chips to Play, How many would you like? (Your Chip Limit is 2250)
1000
Sorry invalid amount, Try again.
500
Ok, your buy in is 1792
You're in the Small Blind.
Your total after blinds is 1782
Okay, the Pot is 10 after the Blinds

1 - 1 of Spades
2 - 13 of Diamonds
3 - 10 of Hearts
4 - 4 of Spades
5 - 9 of Diamonds

Do you want to change cards? (y/n):
```

Stopped there to save space

References

"The C++ Standard Template Library (STL)." *GeeksforGeeks*, 19 Feb. 2020, www.geeksforgeeks.org/the-c-standard-template-library-stl/.