Project 1

<Texas Hold'em Game>

CIS-17C
Name: Shuai Xiong
Date: 10.20.19

Introduction

Title: Texas Hold'em

Texas hold'em is a variation of the card game of poker. The game can be played with minimum of two people with up to ten people. However, for the best game experience, the maximum is six people.

In Texas hold'em players are trying to make the best five-card poker hand according to traditional poker rankings. Which is ranging from highest to lowest is

1. Royal flush, A,K,Q,J,10, al the same suit;
2. Straight flush, five cards in a sequence, all in the same suit;
3. Four of a kind, all four cards of the same rank
4. Full house, Three of a kind with a pair
5. Flush, five cards all of the same suit
6. Straight, five cards of sequential rank, not all of the same suit
7. Three of a kind, three cards of the one rank and two cards of two other ranks
8. Two pair, two cards of on rank, tow cards of another rank and one card of a third rank
9. One pair, two cards of one rank
10. High card, also known as no pair or simply nothing,

In Texas hold'em each player is dealt two cards face down (the "hole cards"), then over the course of subsequent rounds five more cards are eventually dealt face up in the middle of the table, called "community cards", which each player uses them to make a five-card poker hand.

The five community cards are dealt in three stages. The first three community cards are called the "flop." Then just one card is dealt, called the "turn." Finally one more card, the fifth and final community card, is dealt called the "river"

Plyers construct their five-card poker hands using the best available five cards out of the seven total cards (the two hole cards and five community cards). If the betting cause all but one plyer to fold, the lone remaining player wins the pot without having to show any cards.

Play moves clockwise around the table, starting with action to the left of the dealer button, which rotated one seat to the left every hand.

Before every new hand, two players at the table are obligated to post small and big blinds. These are forced bets that begin the wagering. In the first betting round, pre-flop action, two "hole cards" are dealt face down and the first round of betting begins. The first player to act is the layer to the left of the big blind. The player has three option,

- Call: match the amount of the big blind
- Raise: increase the bet within the specific limits of the game
- Fold; throw the hand away.

If the player chooses to fold, the player is no longer eligible to win the current hand. After the first player “under the gun” acts, play proceeds in a clockwise fashion around the table with each player also having the same three option, to call, to raise, or fold.

Second betting round, the flop, which three community cards are dealt on the table and new betting round begins. In this betting round, actions starts with the first active player to the left of the button. Along with the options to call, raise and fold. Now play has the options to “check” if no betting action has occurred beforehand. A check means to pass the actions to the next player in the hand.

Third betting round, the turn, which the fourth community card is called the “turn” ad again a new round of betting starts.

Final betting round, the river, which the last community card is called the “river. “ this is followed by the last round of betting and finally the “showdown”. After all betting actions has been completed, the remaining players in the hand with hole cards now expose their holdings to determine a winner. This is called the showdown.

Summary

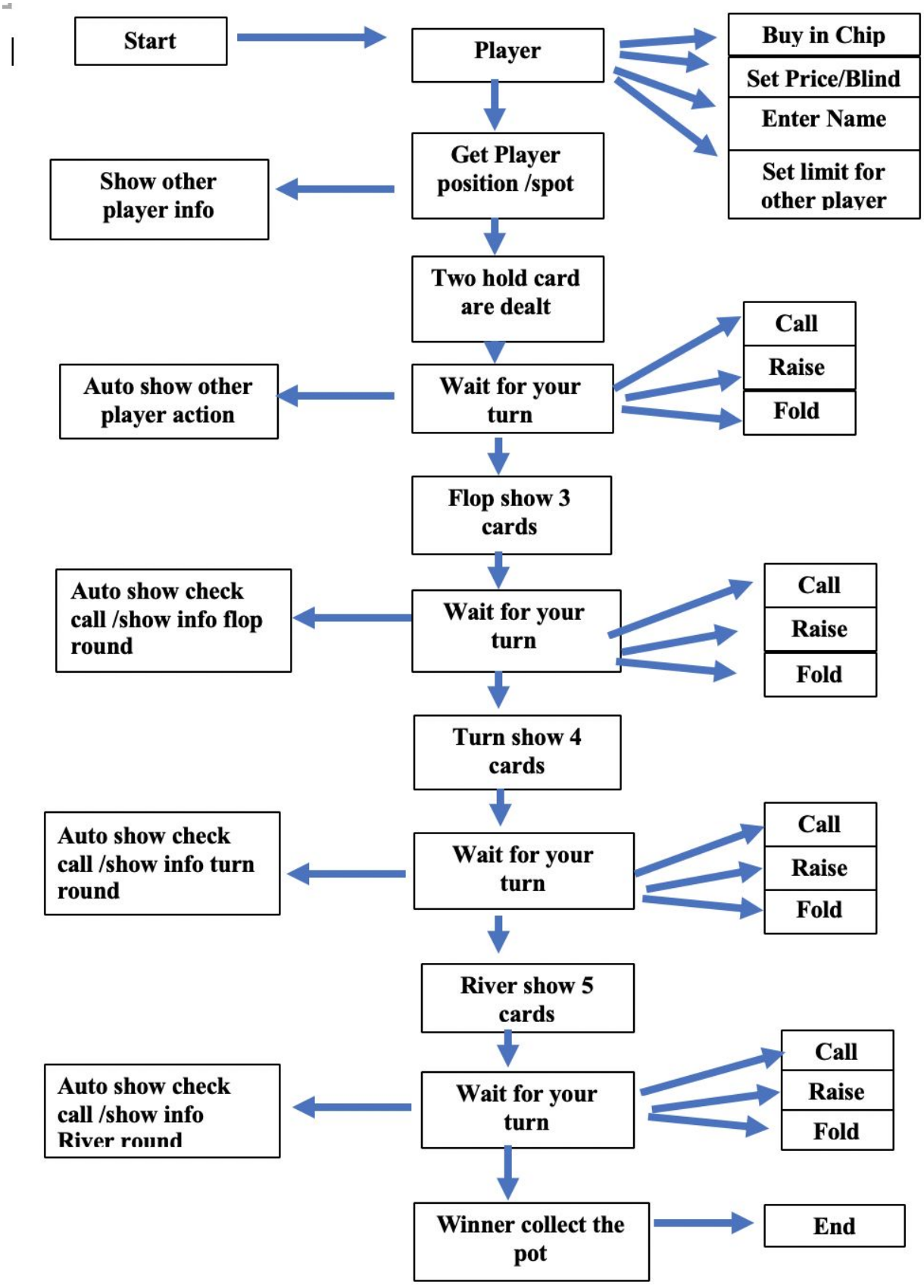
Project size: 700+ lines
?? The number of variables: about 30
The number of method: 17

This project took around 5 days to completed. It was no so hard because of all the pass project experience. However I met some problems in the beginning with setting up the game, and also with all plyer betting case scenario. I refer to the past project, book and also some web game rule and scenario for some idea and suggestions.

Description

The objective of Texas Holdem is to make the best five-card hand you can, using a combination of the two “hole cards” the player are dealt and the five community cards on the board.

Flow Chart


Start
Player
Buy in Chip
Set Price/Blind
Enter Name
Set limit for other player
Get Player position /spot
Show other player info
Two hold card are dealt
Wait for your turn
Call
Raise
Fold
Auto show other player action
Flop show 3 cards
Wait for your turn
Call
Raise
Fold
Auto show check call /show info flop round
Turn show 4 cards
Wait for your turn
Call
Raise
Fold
Auto show check call /show info turn round
River show 5 cards
Wait for your turn
Call
Raise
Fold
Auto show check call /show info River round
Winner collect the pot
End

```
/Users/william/CLionProjects/TexasHoldem/cmake-build-debug/TexasHoldem
Welcome to play Texas Holdem!
Please Enter the Blind:
10
Please Enter the name:
william
How many chips do you need to buy?
400
How many players do you want to play with?(2-6)
2
Hello william you will start to play blind 10/20 game! please waiting for other player
Alexis : $201 Join the game!

position is :
Alexis--->BB
william--->SB

game start...
shuffer card...

william blind: 10
william chips: 390
Alexis big blind: 20
Alexis chips: 181
william please choose:
1:Call
2:Fold
3:Raise
4:All in
```

Pseudo Code
Major Variable

Position	Button	btn
	Big Blind	BB
	Small Blind	SB
	Under the Gun	UTG
	Cut OFF	CO
	Middle Position	MP
		FLOW(below)
Order	Before flip card	UTG => MP => CO =>Btn =>SB => BB

	After flip card	SB => BB => UTG => MP => MP => CO => Btn
Action	Bet, Call, Fold, Check, Raise, Reraise, All In	
Flow	Pre-flop, flop, flop-round, turn, turn-round, river, river-round	
Suit	Spade, heart, club, diamond	
Actions	Shuffle, Burn, Dealt	
Player	2-6 person	
Order		
Player 2	Before flip card	SB =>BB
	After flip card	SB => BB
Player 3	Before flip card	Btn => SB => BB
	After flip card	SB => BB => Btn
Player 4	Before flip card	UTG => Btn => SB => BB
	After flip card	SB => BB => UTG => Btn
Player 5	Before flip card	UTG => CO => Btn =>SB =>BB
	After flip card	SB => BB => UTG => co => Btn
Player 6	Before flip card	UTG=> MP =>CO =>Btn =>SB => BB
	After flip card	SB => BB => UTG => MP => CO=>Btn

Return type	**Variable Name**	**function name**	**Location**
		Card()	Card.h
		string print();	Card.h
string,string	cardFace, cardSuit	Card(string cardFace, string cardSuit);	Card.h

string	suit;		Card.h
string	suit;		Card.h
Card	deck		Deckofcards.h
stack<Card>;		deckList()	Deckofcards.h
int	int currentCard;		Deckofcards.h
		DeckOfCards();	Deckofcards.h
stack<Card>		shuffle();	Deckofcards.h
Card		dealCard();	Deckofcards.h
string	name		Player.h
int	chip		Player.h
string	position		Player.h

Card	card[2]	Player()	Player.h
string,string	name,chips	Player(string name,int chips)	Player.h
string	UTG		Position.h
string	MP		Position.h
string	CP		Position.h
string	BTN		Position.h
string	SB		Position.h
string	BB		Position.h
map<string, int> Utils.h	(int n)	getRandomPlayers	Utils.h
set<string>	(int n)	RandomNames	Utils.h

map<Player*,string>	(map<string,int>)	getRandomPosition	Utils.h
map<string,Player*>	(map<Player*,string> players,DeckOfCards* deck,int blind)	processOrderByPreflop	Utils.h

Java Constructs

Reference:

1. Textbook
2. API/Languages (Java 2 Platform API Speficiation)
3. Jdk 1.2.2 – demo files

Program

main.cpp

```
#include <iostream>
#include <cstdlib>
#include <ctime>
#include "Player.h"
#include "Colors.h"
#include "Utils.h"
#include "DeckOfCards.h"
using namespace std;
int main() {
  //seed
  srand(static_cast<unsigned int>(time(0)));

  //Declare Variables
  int blind,name,chips,numPlayers,mainPot;
  string player1;
  Player player;


  //Input or initialize values Here
  cout <<FBLU( "Welcome to play Texas Holdem! ")<<endl;
  cout <<FBLU( "Please Enter the Blind: ")<<endl;
  //cin>>blind;
  blind=10;
  cout <<FBLU( "Please Enter the name: ")<<endl;
  //cin>>player.name;
  player.name="william";
  cout <<FBLU( "How many chips do you need to buy? ")<<endl;
  //cin>>player.chips;
  player.chips=200;
  cout <<FBLU( "How many players do you want to play with?(2-6)")<<endl;

  //player cant be n<2 or n>6
  //cin>>numPlayers;
  numPlayers=2;
  while(numPlayers<2||numPlayers>6){
     cout<<FRED("Player need to be 2-6!")<<endl;
     cin>>numPlayers;
  }
```

```
   cout<<"Hello "<<player.name<<" you will start to play blind "<<blind<<"/"<<blind*2<<"
game! please waiting for other player";

   //get Random player name and position
   Utils utils;
   map<string,int> Pl=utils.getRandomPlayers(numPlayers);
   Pl[player.name]=player.chips;
   map<Player*,string> Players=utils.getRandomPosition(Pl);
   map<Player*,string>::iterator itr;

   cout<<"game start..."<<endl;
   cout<<"shuffer card..."<<endl;
   DeckOfCards* deck=new DeckOfCards();
   deck->shuffle();
   cout<<endl;

   //Pre-flop everyone get two cards
   //push to queues
   map<string,Player*> players=utils.processOrderByPreflop(Players,deck,blind);
   mainPot=3;


   //call


   return 0;
}


Card.cpp
//
// Created by william shuai xiong on 10/20/19.
//

//assigns the 52 cards to deck
#include "Card.h"

Card::Card(string cardFace, string cardSuit)
{
   face = cardFace;
   suit = cardSuit;
}

Card::Card()
{
```

```
}
string Card::print()
{
  return (face + " of " + suit);
}
```

Card.h

```
//
// Created by william shuai xiong on 10/20/19.
//

#ifndef TEXASHOLDEM_CARD_H
#define TEXASHOLDEM_CARD_H

#include <string>
using namespace std;

class Card
{

private:
  string face;
  string suit;

public:
  Card();
  string print();
  Card(string cardFace, string cardSuit);
};



#endif //TEXASHOLDEM_CARD_H
```

DeckOfCards.cpp

```
//
// Created by william shuai xiong on 10/20/19.
```

```
//



#include "DeckOfCards.h"
#include <list>
#include <iostream>
using namespace std;
DeckOfCards::DeckOfCards()
{
//put all the face values in an array as strings
  string faces[] = {"Ace", "2", "3", "4", "5", "6", "7", "8", "9", "10", "Jack", "Queen",
"King"};
  string suits[] = {"Heart", "Diamond", "Club", "Spade"};

  for(int count = 0; count < 52; count++)
  {
    deck[count] = Card(faces[count % 13], suits[count / 13]);
  }
  for(int i=0;i<52;i++){
    deckList.push(deck[i]);

  }
}


stack<Card> DeckOfCards::shuffle()
{
  currentCard = 0;
  stack<Card> l;
  for(int first = 0; first < 52; first++)
  {
    int second = rand()% 52;
    Card tmp = deck[first];
    deck[first] = deck[second];
    deck[second] = tmp;
  }

  for(int i=0;i<52;i++){
    deckList.push(deck[i]);
    return deckList;
  }
  return l;
```

```
}


Card DeckOfCards::dealCard()
{
  Card card=deckList.top();
  deckList.pop();
  return card;
}
```

DeckOfCards.h

```
//
// Created by william shuai xiong on 10/20/19.
//

#ifndef TEXASHOLDEM_DECKOFCARDS_H
#define TEXASHOLDEM_DECKOFCARDS_H

#include "Card.h"
#include <stack>

class DeckOfCards {

private:
  Card deck[52]; // an array of cards of size SIZR
  stack<Card> deckList;
  int currentCard;

public:
  DeckOfCards();
  stack<Card> shuffle();
  Card dealCard();

};


#endif //TEXASHOLDEM_DECKOFCARDS_H
```

colors.h

```
//
// Created by william shuai xiong on 10/20/19.
//

#ifndef TEXASHOLDEM_COLORS_H
#define TEXASHOLDEM_COLORS_H

#define RST  "\x1B[0m"
#define KRED  "\x1B[31m"
#define KGRN  "\x1B[32m"
#define KYEL  "\x1B[33m"
#define KBLU  "\x1B[34m"
#define KMAG  "\x1B[35m"
#define KCYN  "\x1B[36m"
#define KWHT  "\x1B[37m"

#define FRED(x) KRED x RST
#define FGRN(x) KGRN x RST
#define FYEL(x) KYEL x RST
#define FBLU(x) KBLU x RST
#define FMAG(x) KMAG x RST
#define FCYN(x) KCYN x RST
#define FWHT(x) KWHT x RST

#define BOLD(x) "\x1B[1m" x RST
#define UNDL(x) "\x1B[4m" x RST
#endif //TEXASHOLDEM_COLORS_H
```

names.txt

```
Sophia
Isabella
Emma
Olivia
Ava
Emily
Abigail
Madison
Mia
Chloe
Elizabeth
Ella
```

Addison
Natalie
Lily
Grace
Samantha
Avery
Sofia
Aubrey
Brooklyn
Lillian
Victoria
Evelyn
Hannah
Alexis
Charlotte
Zoey
Leah
Amelia
Zoe
Hailey
Layla
Gabriella
Nevaeh
Kaylee
Alyssa
Anna
Sarah
Allison
Savannah
Ashley
Audrey
Taylor
Brianna
Aaliyah
Riley
Camila
Khloe
Claire
Sophie
Arianna
Peyton
Harper
Alexa
Makayla
Julia

Kylie
Kayla
Bella
Katherine
Lauren
Gianna
Maya
Sydney
Serenity
Kimberly
Mackenzie
Autumn
Jocelyn
Faith
Lucy
Stella
Jasmine
Morgan
Alexandra
Trinity
Molly
Madelyn
Scarlett
Andrea
Genesis
Eva
Ariana
Madeline
Brooke
Caroline
Bailey
Melanie
Kennedy
Destiny
Maria
Naomi
London
Payton
Lydia
Ellie
Mariah
Aubree
Kaitlyn
Violet
Rylee

Lilly
Angelina
Katelyn
Mya
Paige
Natalia
Ruby
Piper
Annabelle
Mary
Jade
Isabelle
Liliana
Nicole
Rachel
Vanessa
Gabrielle
Jessica
Jordyn
Reagan
Kendall
Sadie
Valeria
Brielle
Lyla
Isabel
Brooklynn
Reese
Sara
Adriana
Aliyah
Jennifer
Mckenzie
Gracie
Nora
Kylee
Makenzie
Izabella
Laila
Alice
Amy
Michelle
Skylar
Stephanie
Juliana

Rebecca
Jayla
Eleanor
Clara
Giselle
Valentina
Vivian
Alaina
Eliana
Aria
Valerie
Haley
Elena
Catherine
Elise
Lila
Megan
Gabriela
Daisy
Jada
Daniela
Penelope
Jenna
Ashlyn
Delilah
Summer
Mila
Kate
Keira
Adrianna
Hadley
Julianna
Maci
Eden
Josephine
Aurora
Melissa
Hayden
Alana
Margaret
Quinn
Angela
Brynn
Alivia
Katie

Ryleigh
Kinley
Paisley
Jordan
Aniyah
Allie
Miranda
Jacqueline
Melody
Willow
Diana
Cora
Alexandria
Mikayla
Danielle
Londyn
Addyson
Amaya
Hazel
Callie
Teagan
Adalyn
Ximena
Angel
Kinsley
Shelby
Makenna
Ariel
Jillian
Chelsea
Alayna
Harmony
Sienna
Amanda
Presley
Maggie
Tessa
Leila
Hope
Genevieve
Erin
Briana
Delaney
Esther
Kathryn

Ana
Mckenna
Camille
Cecilia
Lucia
Lola
Leilani
Leslie
Ashlynn
Kayleigh
Alondra
Alison
Haylee
Carly
Juliet
Lexi
Kelsey
Eliza
Josie
Marissa
Marley
Alicia
Amber
Sabrina
Kaydence
Norah
Allyson
Alina
Ivy
Fiona
Isla
Nadia
Kyleigh
Christina
Emery
Laura
Cheyenne
Alexia
Emerson
Sierra
Luna
Cadence
Daniella
Fatima
Bianca

Cassidy
Veronica
Kyla
Evangeline
Karen
Adeline
Jazmine
Mallory
Rose
Jayden
Kendra
Camryn
Macy
Abby
Dakota
Mariana
Gia
Adelyn
Madilyn
Player.h

```
//
// Created by william shuai xiong on 10/20/19.
//

#ifndef TEXASHOLDEM_PLAYER_H
#define TEXASHOLDEM_PLAYER_H


#include <iostream>
#include "Card.h"
using namespace std;

class Player {
public:
  string name;
  int chips;
  string position;
  Card card[2];

  Player(){};

  Player(string name,int chips){
     this->name=name;
```

```
    this->chips=chips;
  }


};


#endif //TEXASHOLDEM_PLAYER_H
```

Position.h

```
//
// Created by william shuai xiong on 10/20/19.
//

#ifndef TEXASHOLDEM_POSITION_H
#define TEXASHOLDEM_POSITION_H

#include <string>
using namespace std;
struct Position{
  string UTG="UTG";
  string MP="MP";
  string CO="CO";
  string BTN="BTN";
  string SB="SB";
  string BB="BB";
};

#endif //TEXASHOLDEM_POSITION_H
```

Utils.cpp

```
//
// Created by william shuai xiong on 10/20/19.
//

#include <iostream>
#include <fstream>
```

```
#include <string>
#include <vector>
#include <cstdlib>   // for exit(), srand(), rand()
#include "list"
#include "Utils.h"
#include <fstream>
#include <vector>
#include <set>
#include <map>
#include <stack>
#include "Colors.h"
#include "Player.h"
#include "Position.h"
#include <vector>
#include <random>
#include <queue>
#include "DeckOfCards.h"

using namespace std;
int myrandom (int i) { return std::rand()%i;}
map<string,int> Utils::getRandomPlayers(int n){

      map<string,int> names;
      int chips;
      string name;

      set<string> setNames=RandomNames(n);
      set<string> :: iterator itr;
      for(itr=setNames.begin();itr!=setNames.end();++itr){
         chips=rand()%200+100;
         name=*itr;
         names[name]=chips;
         cout<<name<<" : $"<<chips<<FYEL( " Join the game! ")<<endl;
      }

      return names;

}
set<string> Utils::RandomNames(int n){

   string name_file="../names.txt";
   vector<string> name_vec;
   set<string> names;

   ifstream infile;
```

```
    infile.open(name_file.c_str());
    if (!infile) {
        cerr << "c" <<  name_file << endl;
        exit(1);
    }
    for (string someName; infile >> someName; ) {
        name_vec.push_back(someName);
    }
    infile.close();

    //get until different name
    while(1){
        names.insert(name_vec.at(rand()%200+1));
        if(names.size()>=n-1){
            break;
        }
    }
    return names;
}

//let position to player
map<Player*,string> Utils::getRandomPosition(map<string,int> p) {
    map<Player*,string> players;
    Position position;
    map<string,int>::iterator it;
    map<Player*,string>::iterator itr;

    for (it=p.begin(); it!=p.end(); ++it) {
        Player* player=new Player;
        player->name=it->first;
        player->chips=it->second;
        players[player]=position.BB;
    }

    //set 23456 player position,
    // we can always choose our position when we player real game
    switch(p.size()){
        case 2:
        {
            //shuffle
            vector<string> l;
            l.push_back(position.BB);
            l.push_back(position.SB);
            random_shuffle(l.begin(),l.end(),myrandom);
            vector<string>::iterator it;
```

```
    stack<string> s;
    for (it=l.begin(); it != l.end(); ++it)
        s.push(*it);

    for(itr=players.begin();itr!=players.end();++itr){
        itr->second=s.top();
        s.pop();
    }
    break;
}
case 3:
{
    //shuffle
    vector<string> l;
    l.push_back(position.BTN);
    l.push_back(position.BB);
    l.push_back(position.SB);
    random_shuffle(l.begin(),l.end(),myrandom);
    vector<string>::iterator it;
    stack<string> s;
    for (it=l.begin(); it != l.end(); ++it)
            s.push(*it);

    for(itr=players.begin();itr!=players.end();++itr){
        itr->second=s.top();
        s.pop();
    }
    break;
}
case 4:
{
    //shuffle
    vector<string> l;
    l.push_back(position.BTN);
    l.push_back(position.BB);
    l.push_back(position.SB);
    l.push_back(position.UTG);
    random_shuffle(l.begin(),l.end(),myrandom);
    vector<string>::iterator it;
    stack<string> s;
    for (it=l.begin(); it != l.end(); ++it)
        s.push(*it);

    for(itr=players.begin();itr!=players.end();++itr){
        itr->second=s.top();
```

```
        s.pop();
    }
    break;
}
case 5:
{
    //shuffle
    vector<string> l;
    l.push_back(position.BTN);
    l.push_back(position.BB);
    l.push_back(position.SB);
    l.push_back(position.UTG);
    l.push_back(position.CO);
    random_shuffle(l.begin(),l.end(),myrandom);
    vector<string>::iterator it;
    stack<string> s;
    for (it=l.begin(); it != l.end(); ++it)
        s.push(*it);

    for(itr=players.begin();itr!=players.end();++itr){
        itr->second=s.top();
        s.pop();
    }
    break;
}

case 6:
{
    //shuffle
    vector<string> l;
    l.push_back(position.BTN);
    l.push_back(position.BB);
    l.push_back(position.SB);
    l.push_back(position.UTG);
    l.push_back(position.CO);
    l.push_back(position.MP);
    random_shuffle(l.begin(),l.end(),myrandom);
    vector<string>::iterator it;
    stack<string> s;
    for (it=l.begin(); it != l.end(); ++it)
        s.push(*it);

    for(itr=players.begin();itr!=players.end();++itr){
        itr->second=s.top();
        s.pop();
```

```
        }
        break;
      }
    }

    cout<<endl;
    cout<<"position is :"<<endl;
    for(itr=players.begin();itr!=players.end();++itr){
      cout<<itr->first->name<<"--->"<<itr->second<<endl;
      itr->first->position=itr->second;
    }
    cout<<endl;
return players;
}


map<string,Player*> Utils::processOrderByPreflop(map<Player*,string> players,DeckOfCards*
deck,int blind){
    Position position;
    map<string,Player*> map;
    queue<Player> q;
    int mainPot;

    //sort to queue
    switch(players.size()){
      case 2:{
        for (auto it = players.begin(); it != players.end(); ++it)
          if (it->second == position.SB) {
            it->first->card[0]=deck->dealCard();
            it->first->card[1]=deck->dealCard();
            it->first->chips=it->first->chips-blind;
            cout<<it->first->name<<" blind: "<<blind<<endl;
            cout<<it->first->name<<" chips: "<<it->first->chips<<endl;
            map[it->first->position]=it->first;
            q.push(*it->first);
          }
        for (auto it = players.begin(); it != players.end(); ++it)
          if (it->second == position.BB) {
            it->first->card[0] = deck->dealCard();
            it->first->card[1] = deck->dealCard();
            it->first->chips=it->first->chips-blind*2;
            cout<<it->first->name<<" big blind: "<<blind*2<<endl;
            cout<<it->first->name<<" chips: "<<it->first->chips<<endl;
            map[it->first->position]=it->first;
            q.push(*it->first);
```

```
            }
        break;
    }
}
mainPot=blind*3;
//if not empty keep call
list<Player> maplist;
int raise=0;
while(q.size()>0){
    int o;
    cout<<q.front().name<<" please choose: "<<endl;
    cout<<"1:Call \n2:Fold  \n3:Raise  \n4:All in\n";
    cin>>o;
    switch(o){
        case 1:{
            if(raise==0){
                q.front().chips= q.front().chips-blind;
                mainPot+=blind;
                maplist.push_back(q.front());
                q.pop();
            }else{
                q.front().chips= q.front().chips-raise;
                mainPot+=raise;
                maplist.push_back(q.front());
                q.pop();
            }
            break;
        }
        case 2:{
            cout<< "this turn finish, shuffer and play again. " <<endl;
            break;
        }
        case 3:{
            cout<< " how much you raise"<<endl;
            cin>>raise;
            q.front().chips= q.front().chips-raise;
            mainPot+=raise;
            maplist.push_back(q.front());
            q.pop();
            q.push(maplist.front());
            break;
        }
        case 4:{
         break;
        }
```

```
    }
  }

  return map;
}



//calculate bets
```

Utils.h

```
//
// Created by william shuai xiong on 10/20/19.
//

#ifndef TEXASHOLDEM_UTILS_H
#define TEXASHOLDEM_UTILS_H
#include <iostream>
#include <list>
#include <set>
#include <map>
#include "Player.h"
#include "Position.h"
#include "DeckOfCards.h"
#include <queue>
using namespace std;

class Utils {
public:
  map<string, int> getRandomPlayers(int n);              //get players and chips
  set<string> RandomNames(int n);                  //read file and get random name
  map<Player*,string> getRandomPosition(map<string,int> );          //get getRandomPosition
  map<string,Player*> processOrderByPreflop(map<Player*,string> players,DeckOfCards*
deck,int blind);  //proess orderby pre flop,get cards ,return queue
  // void showMianPots(queue<Player*> pq);                 //calculate bets

};
#endif //TEXASHOLDEM_UTILS_H
```