Branch: master ▾ Find file Copy path

LeeByoungmo_CIS_17c_47698 / Proj / texasHoldem_17C_Ver1.3 / Dealer.cpp

libmo No commit message

024972d 1 hour ago

0 contributors

Raw Blame History

423 lines (354 sloc) 12.4 KB

```
/*
 * File:   Dealer.cpp
 * Author: Byoung Mo Lee
 * Created on May 17, 2019 00:19 AM
 * Purpose: Texas Holdem
 *
 */
#include <string>
#include <iostream>
#include <cstdlib>
#include "Dealer.h"

using namespace std;
Dealer::Dealer(int num){
    int menu=1;
    this->setPlayers(num);
    this->DeckOfCards();
    do{
        this->setBlind();
        this->setIniCont(50);
        this->resetPot();

        this->shuffle();
        this->preflop();

        int amount=bettingPrompt1();
        cout << "bettingPrompt1 is Done" << endl;
    //    for(int i=0;i<this->numPlayers;i++){
    //        cout << this->players[i].getBalance() << endl;
    //    }
    //    cout << this->potAmount << endl;


        int cnt=1;
        int nP=this->numPlayers;
        int x=1;

        do{

            amount=this->bettingPrompt2((cnt+this->smallBlind)%nP,amount);
            if((cnt+this->bigBlind)%nP==this->bigBlind) x=0;
            if(this->Dealer::getNumAct()==1) {
                cout << "find the winner and terminate the round" << endl;
                x=0;
            }
            cnt++;
        }while(amount-this->players[(cnt+this->smallBlind)%nP].putInThePot()||x);
        cout << "bettingPrompt2 is Done" << endl;

        this->flop();
        cnt=0;
```

```
52          x=1;
53          while((amount-this->players[(cnt+this->bigBlind)%nP].putInThePot())||x){
54              int aP=this->Dealer::getNumAct();
55              amount=this->bettingPrompt3((cnt+this->bigBlind)%nP,amount);
56              cnt++;
57              if((cnt+this->bigBlind)%nP==this->bigBlind) x=0;
58              if(this->Dealer::getNumAct()==1) {
59                  cout << "find the winner" << endl;
60                  x=0;
61              }
62
63          }
64          cout << "bettingPrompt3 is Done" << endl;
65
66          this->turn();
67
68          cnt=0;
69          x=1;
70          while((amount-this->players[(cnt+this->bigBlind)%nP].putInThePot())||x){
71              int aP=this->Dealer::getNumAct();
72      //        cout << "ap=" << aP <<endl;
73              amount=this->bettingPrompt3((cnt+this->bigBlind)%nP,amount);
74              cnt++;
75              if((cnt+this->bigBlind)%nP==this->bigBlind) x=0;
76              if(this->Dealer::getNumAct()==1) {
77                  cout << "find the winner" << endl;
78                  x=0;
79              }
80
81          }
82          cout << "bettingPrompt3 is Done" << endl;
83
84          this->river();
85          cnt=0;
86          x=1;
87          while((amount-this->players[(cnt+this->bigBlind)%nP].putInThePot())||x){
88              int aP=this->Dealer::getNumAct();
89              amount=this->bettingPrompt3((cnt+this->bigBlind)%nP,amount);
90              cnt++;
91              if((cnt+this->bigBlind)%nP==this->bigBlind) x=0;
92              if(this->Dealer::getNumAct()==1) {
93                  cout << "find the winner" << endl;
94                  x=0;
95              }
96
97          }
98          cout << "bettingPrompt3 is Done" << endl;
99
100
101
102         for(int i=0;i<this->numPlayers;i++){
103             int order=(i+this->bigBlind)%this->numPlayers;
104             cout << "Player" << order << ": " << this->players[order].getHandsName() << endl;
105
106     //        this->players[order].checkStraight();
107     //        this->players[order].setFaces();
108     //        this->players[order].setSuits();
109     //        this->Dealer::displayPlayersInfo(order);
110         }
111         this->calBal();
112         for(int i=0;i<this->numPlayers;i++){
113             int order=(i+this->bigBlind)%this->numPlayers;
114             this->Dealer::displayPlayersInfo(order);
115         }
116
117         menu=this->resetGame();
```

```
118     }while(menu!=0);
119 }
120 
121 
122 void Dealer::setPlayers(int num){
123     this->numPlayers=num;
124     this->players=new Hands[this->numPlayers];
125 
126     for(int i=0;i<this->numPlayers;i++) {
127               this->players[i].setPlayerName("player#" + to_string(i));
128             }
129 }
130 
131 void Dealer::setBlind(){
132     this->bigBlind=this->getRound()%(this->getNumPlayers()-1);
133     this->smallBlind=this->bigBlind+1;
134 }
135 
136 
137 void Dealer::displayPlayersInfo(int num){
138     cout << this->Dealer::players[num].getName() << endl;
139     if(num==this->Dealer::getBigBlind()) {cout << "BB" << endl;}
140     else if (num==this->Dealer::getSmallBlind()) {cout << "SB" << endl;}
141     cout << "Balance: $" <<this->Dealer::players[num].getBalance() << endl << endl;
142     for(auto& it:this->players[num].getMyCards()){
143         it.toString();
144     }
145     cout << endl;
146 }
147 
148 void Dealer::DeckOfCards() {
149     for(int i=0;i<52;i++){
150         this->deck[i]=new Card(i);
151         //cards[i]->toString();
152     }
153 }
154 
155 
156 
157 void Dealer::shuffle(){
158     // Initialize seed randomly
159     srand(time(0));
160 
161     for (int i=0; i<52 ;i++)
162     {
163         // Random for remaining positions.
164         int r = i + (rand() % (52 -i));
165 
166         swap(this->deck[i], this->deck[r]);
167     }
168 
169 }
170 
171 void Dealer::preflop(){
172     int nCards=2;
173     int order;
174     int nP=this->numPlayers;
175     int bB=this->bigBlind;
176     for(int i=0;i<nCards;i++){
177         for(int j=0;j<nP;j++){
178             order=(j+bB)%nP;
179             this->players[order].addMyCards(*this->deck[j+nP*i]);
180 
181 //            this->deck[j+i+nP*i]->toString();
182 //            this->Dealer::displayPlayersInfo(order);
183 
```

```
        }
    }
}

void Dealer::flop(){
    int nCards=3;
    int nP=this->numPlayers;
    int bB=this->bigBlind;
    for(int i=nP*2+1;i<nP*2+1+nCards;i++){
//        this->deck[i]->toString();
        for(int j=0;j<nP;j++){
            int order=(j+bB)%nP;
            this->players[order].addMyCards(*this->deck[i]);
        }
    }
}

void Dealer::turn(){
    int nCards=1;
    int nP=this->numPlayers;
    int bB=this->bigBlind;
    for(int i=nP*2+1+3+1;i<nP*2+1+3+1+nCards;i++){
//        this->deck[i]->toString();
        for(int j=0;j<nP;j++){
            int order=(j+bB)%nP;
            this->players[order].addMyCards(*this->deck[i]);
        }
    }
}

void Dealer::river(){
    int nCards=1;
    int nP=this->numPlayers;
    int bB=this->bigBlind;
    for(int i=nP*2+1+3+1+1+1;i<nP*2+1+3+1+1+1+nCards;i++){
//        this->deck[i]->toString();
        for(int j=0;j<nP;j++){
            int order=(j+bB)%nP;
            this->players[order].addMyCards(*this->deck[i]);
        }
    }
}

int Dealer::bettingPrompt1(){
    int input;
    int raise;
    int diff;
    int nP=this->numPlayers;
    int amount;
    this->players[bigBlind].bet(this->initialContributeAmount);
    this->potAmount+=this->initialContributeAmount;
    cout << "Player" << this->bigBlind << ": $" << this->initialContributeAmount <<" into the Pot"<<endl;
    cout << "Pot: $" << this->getPotAmount() << endl;
    amount=this->players[bigBlind].putInThePot();

    this->players[smallBlind].bet(this->initialContributeAmount/2);
    this->potAmount+=this->initialContributeAmount/2;
    diff=amount-this->initialContributeAmount/2;


    cout << "Player" << this->smallBlind << ": $" << this->initialContributeAmount/2<<" into the Pot" <<endl;
    cout << "Pot: $" << this->getPotAmount() << endl;


    do{
    cout << "To call you have to put $" << diff << endl;
```

```
    cout << "Player" << this->smallBlind <<": Call - 1, Raise - 2, Fold -3" << endl;
    cin >> input;
    if(input==1) {
            this->players[smallBlind].bet(diff);
            this->potAmount+=diff;
            cout << "Player#" << smallBlind << ": $" << diff << endl;
            cout << "Pot: $" << this->getPotAmount() << endl;

        }
        else if(input==2){
            cout << "Amount: " ;
            do{
                cin >> raise;
                if(raise<=diff) cout << "Amount should be greater than call Amount\n";
            }while(raise<=diff);
            this->players[smallBlind].bet(raise+diff);
            this->potAmount+=(raise+diff);
            cout << "Player#" << smallBlind << ": $" << raise+diff <<" into the Pot" << endl;
            cout << "Pot: $" << this->getPotAmount() << endl;
            amount = this->players[smallBlind].putInThePot();
        }
        else if(input==3){
            this->players[smallBlind].setPlayerInact();
            cout << "Player#" << smallBlind << ": folded" << endl;
            cout << "Pot: $" << this->getPotAmount() << endl;
        }
        else {cout << "wrong input" << endl;}
    }while(input<1||input>3);

    return amount;
}

int Dealer::bettingPrompt2(int num, int amount){
    int input;
    do{
        if(this->players[num].getActStatus()){
            int nP=this->numPlayers;

            int raise;
            int diff=amount - this->players[num].putInThePot();
            cout << "To call you have to put $" << diff << endl;
            cout << "Player" << num <<": call - 1, Raise - 2, Fold -3" << endl;
            cin >> input;
            if(input==1) {
                this->players[num].bet(diff);
                this->potAmount+=(diff);
                cout << "Player#" << num << ": $" << diff << endl;
                cout << "Pot: $" << this->getPotAmount() << endl;
            }
            else if(input==2){
                cout << "Amount: " ;
                do{
                    cin >> raise;
                    if(raise<=diff) cout << "Amount should be greater than call Amount\n";
                }while(raise<=diff);

                this->players[num].bet(raise+diff);
                this->potAmount+=(raise+diff);
                cout << "Player#" << num << ": $" << raise+diff << endl;
                cout << "Pot: $" << this->getPotAmount() << endl;
                amount=this->players[num].putInThePot();
            }
            else if(input==3){
                this->players[num].setPlayerInact();
                cout << "Player#" << num << ": folded" << endl;
                cout << "Pot: $" << this->getPotAmount() << endl;
```

```
            }
            else {cout << "wrong input" << endl;}
        }
    }while(input<1||input>3);

    return amount;
}

int Dealer::bettingPrompt3(int num, int amount){
    int nP=this->numPlayers;
    int input;
    do{
        if(this->players[num].getActStatus()){
            int raise;
            int diff=amount - this->players[num].putInThePot();
            if(!diff){
                cout << "You can check" << endl;
                cout << "Player" << num <<": check - 1, Raise - 2, Fold -3" << endl;

            }
            else{
                cout << "To call you have to put $" << diff << endl;
                cout << "Player" << num <<": call - 1, Raise - 2, Fold -3" << endl;

            }

            cin >> input;
            if(input==1) {

                this->players[num].bet(diff);
                this->potAmount+=(diff);
                cout << "Player#" << num << ": $" << diff << endl;
                cout << "Pot: $" << this->getPotAmount() << endl;
            }
            else if(input==2){
                cout << "Amount: " ;
                do{
                    cin >> raise;
                    if(raise<=diff) cout << "Amount should be greater than call Amount\n";
                }while(raise<=diff);

                this->players[num].bet(raise+diff);
                this->potAmount+=(raise+diff);
                cout << "Player#" << num << ": $" << raise+diff << endl;
                cout << "Pot: $" << this->getPotAmount() << endl;
                amount=this->players[num].putInThePot();
            }
            else if(input==3){
                this->players[num].setPlayerInact();
                cout << "Player#" << num << ": folded" << endl;
                cout << "Pot: $" << this->getPotAmount() << endl;
            }
            else {cout << "wrong input" << endl;}
        }
    }while(input<1||input>3);

    return amount;
}

int Dealer::getNumAct(){
    int num=0;
    for(int i=0;i<this->numPlayers;i++){
        if(this->players[i].getActStatus()) num++;
    }

    return num;
```

```
}

int Dealer::decideWinner(){
    int max=0;
    int winner=0;
    int nP=this->getNumPlayers();

    for(int i=0;i<nP;i++){
        if(this->players[i].getActStatus()==1) {
            if(this->players[i].getHands()>max) {
                max=this->players[i].getHands();
                winner=i;
            }
        }
    }
//    cout << "Max=" << max << endl;
    return winner;

}

void Dealer::calBal(){
    int winner=this->decideWinner();
    cout << "Player" << winner << " won $" << this->getPotAmount() << "!" << endl;
    this->players[winner].addBal(this->getPotAmount());
    this->Dealer::resetPot();
}

int Dealer::resetGame(){
    int input;
    this->nextRound();

    for(int i=0;i<this->getNumPlayers();i++){

        this->players[i].resetHands();
    }


    cout << "if you want to exit press 0" << endl;
    cin >> input;

    return input;
}
```