CS31 Project 7—Black Jack

YaHan Hsu 304513231

1. A brief description of notable obstacles you overcame.

Deal with the busted situation in the outcome.

Throw the exception and catch them in main.

Output the card in hand with overwriting operator<<.

(Friend inside the class)

Cases when both black jack or anyone got black jack.

1. A list of the test data that could be used to thoroughly test your functions, along with the reason for each test. You must note which test cases your program does not handle correctly.

// test code

assert(p.handcount() == 0);

assert(p.hasBlackJack() == false);

assert(p.cardCount() == 0);

p.acceptCard(c);

assert(p.handcount() == 11);

assert(p.hasBlackJack() == false);

assert(p.cardCount() == 1);

assert(p.getCard(0) == c);

p.acceptCard(d);

assert(p.handcount() == 21);

assert(p.hasBlackJack() == true);

p = Player();

p.acceptCard(c);

p.acceptCard(c);

p.acceptCard(c);

assert(p.handcount()==13);

//test the logic error when ask the cards that you dont have

try {

p.getCard(5); //should throw an excption

}

catch (std::logic\_error & le)

{

cerr << le.what() << endl;

}

//test the logic error when deal too many cards

p = Player();

try {

p.acceptCard(c);

p.acceptCard(c);

p.acceptCard(c);

p.acceptCard(c);

p.acceptCard(c);

p.acceptCard(c);

p.acceptCard(c);

p.acceptCard(c);

p.acceptCard(c);

p.acceptCard(c);

p.acceptCard(c);

p.acceptCard(c);

p.acceptCard(c); //should throw an excption

}

catch (exception& e)

{

cerr << e.what()<<endl;

}

p = Player();

Player dealer;

Game g(p, dealer);

g.playerStands();

g.dealerStands();

//no one should be busted at the beginning

assert(g.dealerBusted() == false);

assert(g.playerBusted() == false);

//both has no cards

assert(g.playerTied() == true);

assert(g.playerWon() == false);

assert(g.playerLost() == false);

assert(g.playerHasBlackJack() == false);

assert(g.dealerHasBlackJack() == false);

//case when player got BlackJack

p = Player();

p.acceptCard(c);

p.acceptCard(d);

dealer = Player();

dealer.acceptCard(e);

dealer.acceptCard(f);

g = Game(p, dealer);

g.playerStands();

g.dealerStands();

assert(g.dealerBusted() == false);

assert(g.playerBusted() == false);

assert(g.playerTied() == false);

assert(g.playerWon() == true);

assert(g.playerLost() == false);

assert(g.playerHasBlackJack() == true);

assert(g.dealerHasBlackJack() == false);

//case when player is busted

p = Player();

p.acceptCard(c);

p.acceptCard(d);

p.acceptCard(e);

p.acceptCard(f);

dealer = Player();

dealer.acceptCard(e);

dealer.acceptCard(f);

g = Game(p, dealer);

g.playerStands();

g.dealerStands();

assert(g.dealerBusted() == false);

assert(g.playerBusted() == true);

assert(g.playerTied() == false);

assert(g.playerWon() == false);

assert(g.playerLost() == true);

assert(g.playerHasBlackJack() == false);

assert(g.dealerHasBlackJack() == false);

//case when both 21, with player having BlackJack

p = Player();

p.acceptCard(c);

p.acceptCard(d);

dealer = Player();

dealer.acceptCard(d);

dealer.acceptCard(e);

dealer.acceptCard(f);

g = Game(p, dealer);

g.playerStands();

g.dealerStands();

assert(g.dealerBusted() == false);

assert(g.playerBusted() == false);

assert(g.playerTied() == true);

assert(g.playerWon() == false);

assert(g.playerLost() == false);

assert(g.playerHasBlackJack() == true);

assert(g.dealerHasBlackJack() == false);

//case when both black jack

p = Player();

p.acceptCard(c);

p.acceptCard(d);

dealer = Player();

dealer.acceptCard(d);

dealer.acceptCard(c);

g = Game(p, dealer);

g.playerStands();

g.dealerStands();

assert(g.dealerBusted() == false);

assert(g.playerBusted() == false);

assert(g.playerTied() == true);

//clearScreen();

g.dealerPlays();

if (g.playerWon())

{

g.display("WON!", true);

}

else if (g.playerLost())

{

g.display("LOST!", true);

}

else

{

g.display("TIED!", true);

}

cout << "all tests passed!" << endl;

cin.get();

return 0;