Master Test Plan

Proposed Tests:

playGame Test:

Coverage expected:

- Setting up the black jack game
- Player and dealer actions
- Game logic to determine winner of the round
- UI input and output during game and after round

Mocks used: Deck, Cards, Player, Hand, UI

Test playGame function of Blackjack

1. Game Setup
    a. Create mock Player objects (simulation pretends 2 other players)
    b. Create mock Deck object
    c. Create mock Card object
    d. Create mock PlayerUI object
    e. Blackjack instance is created, with mocks injected
    f. Blackjack playGame() is called
2. playRound() is called:
    a. Deck::shuffle() is called once
    b. Deck::dealCard(Player*) is called 6 times:
        - Hand::addCard(Card*) is called 6 times
        - Mock two aces for Player 1
        - Mock 9 and 4 for Player 2
        - Real 9 and 8 for Dealer
    c. checkNatural() is called
        i. MockPlayer::getHand() called three times
        ii. MockHand::getScore() called three times, returns 12, 13, 17
    d. playerPlay() expected to be called for player 1 (player 1's turn)
        i. MockPlayer::hit() is called twice for each Player object (to receive dealt cards)
        ii. setUI is called once
        iii. MockUI::handleInput() is called 4 times
            - Returns "split", "hit", "stand", "hit"
                1. MockPlayerUI::displayHand is called 4 times
                2. MockPlayer::split() is called
                3. Hand() constructor to be called
                4. MockPlayer::hit(Card*) is called twice, passed in cards with Rank::SEVEN (for first hand) and Rank::TEN (for second hand)
                5. MockHand::setStatus(bool) is called twice, passed parameter true
                6. MockPlayer::setHand(int), value is set to 1
                7. MockPlayer::stand() is called twice (player chooses to stand on 18, but stands automatically on second hand when 21 is hit)
                8. MockPlayer::checkBust() is called twice, returns false both times

            - MockHand::getScore() is called twice, returning 18 and 21
            - MockCard::getValue() is called 4 times, returning 11, 7, 11, and 10
            - MockCard::getRank() is called 4 times, returning ACE, SEVEN, ACE, TEN
        9. MockHand::setScore(int) is called twice, setting hand to 18 and 21

e. playerPlay() is called the second time (for Player 2)
    i. MockPlayerUI::displayHand() is called one time
    ii. MockUI::handleInput() is called once, returns "hit"
    iii. MockPlayer::hit(Card*) is called once, passed in card with Rank::QUEEN value
    iv. MockPlayer::checkBust() is called once, returns true
    v. MockHand::setStatus(bool) is called, passed in parameter true
        - MockHand::setScore(int) setting hand to 0

f. dealerPlay() is called:
    i. setUI is called
    ii. DealerUI::displayHand is called 1 time
    iii. Player::stand() is called once (dealer's hand score => 17)
    iv. Hand::setStatus(bool) is called once, passed in parameter true

g. Blackjack::checkWin() is called 2 times
    i. Player::getHand() called 4 times (assuming Dealer getHand() is called inside checkWin()), returns
    ii. Size 2 vector<Hand> with hands of score 21 and 18 (Player 1)
        - MockHand::getScore() is called twice, returning 18 and 21
    iii. Size 1 vector<Hand> with hand of score 0 (Player 2)
        - MockHand::getScore() is called once, returning 0
    iv. (twice) Size 1 vector<Hand> with hand of score 17 (dealer)
    v. MockPlayer::setScore is called twice
    vi. MockUI::displayWinner is called once

h. MockUI::playAgain is called once, returns true

i. playRound() is called a second time

Card Test

Coverage expected: Card methods

1. Instantiate Card(Suit::SPADE, Rank::THREE)
2. Verify getRank() returns Rank::THREE
3. Verify getSuit() returns Suit::Spade

PlayerHand Test

Coverage expected: Player Methods, some Hand methods

1. Instantiate Player(string "John")
2. Call Player::getName() expect return "John"
3. Instantiate Card Pointers (Card(Rank::ACE, Suit::SPADE), Card(Rank::ACE, Suit::Heart))
4. Call Player::hit(Card*) twice, adding both Ace cards
    - Indirectly calls Hand::addCard(Card*)
5. Call Player::getHand(), verify size of vector returned is 1
6. Call Player::getHand(), call Hand::getCards(), call getRank() on first index of vector, expect return Rank::ACE
7. Call Player::split()

8. Call Player::getHand(), verify size of vector returned is 2
9. Instantiate Card Pointers (Card(Rank::TEN, Suit::SPADE), Card(Rank::EIGHT, Suit::SPADE), Card(Rank::KING, Suit::Heart), Card(Rank::JACK, Suit::CLUB)
10. Call Player::hit(Card*) twice, passing in the Ten card and the Eight Card
11. Call Player::checkBust(), expect return true
    - Hand::setScore() to 0
    - Hand::setStatus() to true
    - Call player::getHand(), call Hand::getScore(), expect 0
12. Call Player::setHand(int), pass in 1
13. Call Player::hit(card*), passing in Jack card
    - Hand::setScore to 21
    - Hand::setStatus() to true
    - Call Player::getHand, call Hand::getScore() expect 21
14. Call Player::getHand(), call getStatus(), verify true
15. Call Player::checkBust(), expect return false
16. Call Player::getScore(), expect 0

HandRemove Test

1. Instantiate Hand()
2. Instantiate Card(Rank::TEN, Suit::SPADE), Card(Rank::EIGHT, Suit::SPADE), Card(Rank::KING, Suit::Heart)
3. Call Hand::Hit(Card*) three times for all three cards
4. Call Hand::getCards(), verify size of vector is 3
5. Call Hand::removeCard()
6. Call Hand::getCards(), check rank and suit for each card is Ten of Spades and Eight of Spades respectively

Test Coverage Required:

Class Blackjack

- Blackjack() – cover: playGame Test 1
- playRound() – cover: playGame Test 2
- resetRound() – cover : playGame Test
- dealCard(Player*) – cover: playGame Test 2 b
- checkBust() – cover: playGame Test 2 d
- checkWin() – cover: playGame Test 2 g
- dealerPlay() – cover: playGame Test 2 f
- playerPlay() – cover: playGame Test 2 d, e
- setUI(UI*) – cover: playGame Test 2 d, e, f

Class Player

- Player(string) – cover: Player 1
- getName() – cover: Player 2
- hit(Card*) - cover: Player 4, 9, 12

- split() – cover: Player 6
- checkBust() – cover: Player 10, 14
- getScore() – cover: Player 15
- getHand() – cover: Player 5, 13

Class Hand

- Hand() – cover: PlayerHand 1
- getCards() – PlayerHand 6
- addCard(Card*) – cover: PlayerHand 4
- getScore() – cover: PlayerHand 11, 13
- setScore() - cover: PlayerHand 11, 13
- removeCard()
- setStatus() – cover: PlayerHand 11, 13
- getStatus() – cover: PlayerHand 14

Class Card

- Card(Rank, Suit) – cover: Card Test 1
- getRank() – cover: Card Test 2
- getValue() – cover: Card Test 3