

COMP 3000 – Assignment 5
Due 11:59 pm – Friday, Nov 15, 2019
Submit Assign05.zip via Canvas

Program: Assign05

For Assignment 5, we will be writing the engine that controls our game of war. You may use the instructor's posted solution for Assignment 4 as a base, if you feel it is needed. Because the rules of war are a bit ambiguous, we are making a few judgement calls regarding how wars work. First, a war is done using 4 cards, three prize cards and one comparison card. Second, If a player enters a war with 0 cards in his deck, he automatically loses the war. Finally, if the player has fewer than 4 cards, they play as many as they can, with the last card being used for comparison against the other's player card. The other player still plays the full amount of cards.

Changes to Deck class:

Two new functions need to be added to the Deck class:

`void addCard(Deck):`

An overloaded version of `addCard(Card)` that adds all cards from one deck to the bottom of another. For instance, `deck1.addCard(deck2)` will add all cards from `deck2` to the bottom of `deck1`.

Pseudocode:

Store initial contributing deck size in a variable (this is to avoid it changing as cards are dealt).

For each card in the contributing deck,

Deal a card from the contributing deck

Add that card to receiving deck.

`int getSize():`

Returns the size of the cards vector.

Overloaded operators:

Four operators need to be overloaded to work with items of type `card`: `<`, `>`, `==`, and `<<`

`<`, `>`, `==`:

Comparison operators should return true or false based on the two cards' numeric values. For instance, in `card1 > card2`, if `card1`'s numeric value is 5 and `card2`'s numeric value is 3, the operator should return true. Remember, numeric values are the values where face cards are assigned a number such as 11 for jacks. Also, aces are considered "high" in War and should have a numeric value of 14.

`<<`:

The insertion operator should write "`card.value` of `card.suit`" to the output stream. For instance, if `myCard` is the 3 of clubs, using "`cout << myCard;`" should output "3 of clubs".

GameEngine Class:

A new class type called GameEngine should be created. GameEngine includes the following:

Private Variables:

Deck player1Deck	the human player's deck
Deck player2Deck	the computer's deck
string playerName	the player's name

Public functions:

void greetPlayer()	A function that greets the player and reads, and stores their name. The program should capture the full line the player enters for their name.
void dealDecks()	A function that makes a deck, shuffles it, and uses that deck to create two decks of half the original deck's size.
void playRound()	A function that outputs each player's deck size, deals one card from each deck, outputs each card, and checks to see which card is higher. If the player's card is higher, The game outputs that the player won the round. Both the player and the computer's cards are added to the player's deck Else if the computer's card is higher, The game outputs that the computer won the round. Both the player and the computer's cards are added to the computer's deck Else, A variable is created to keep track of the number of cards won (starting with 2). The war(int&) function is called, and its result is stored in a variable. If the player won the war, Output that the player won and how many cards. Add both the cards dealt before the war to the player's deck. If the computer won the war, Output that the computer won and how many cards. Add both the cards dealt before the war to the computer's deck.
bool checkWinner()	Output: true or false based on if a winner was found. A function that checks for a winner by checking if either deck's size is equal to 0 and returns if a winner was found. If the player's deck size is 0, Output that the computer won. Return true. Else if the computer's deck size is 0, Output that the computer won. Return true. Else Tell the user to "Press enter to go to the next round." Use "cin.get()" to pause the command line. Return false

bool offerReplay()

Output: True or False based on if the user wishes to play again.
A function that asks if the user would like to play again.
If the user does not wish to play again,
 The program should output a goodbye message and return false.
Else if the user does wish to play again,
 The program should return true.
Otherwise,
 The program should tell the player they entered an invalid response.

Note: This input should be validated (the program should keep asking until a correct response is entered) and should accept answers regardless of case (if “yes” is a valid response, “Yes” should also be valid).

Remember that toUpper() and toLower() only work on a one character at a time.

int war(int&)

Input: a call-by-reference integer that tracks the number of cards won
Output: an integer that indicates if the player won the war (1) or the computer won the war (2)
Pseudocode:
If the player’s deck size is 0,
 Return the computer wins the war.
If the computer’s deck size is 0,
 Return the player wins the war.
Find which is smaller: (the number of cards in the player’s deck – 1) or 3. For help, look into the min() function from the <algorithm> library.
Make a prize deck of cards of size equal to the smaller of the two above numbers.
Repeat for the computer’s deck.
Add the size of both decks + 2 to the input parameter integer.
Deal one card from each deck, output them, and compare them.
If the player’s card is higher,
 Add both cards and both prize decks to the player’s deck.
 Return that the player won the war.
Else if the computer’s card is higher,
 Add both cards and both prize decks to the computer’s deck.
 Return that the computer won the war.
Else,
 Call the war(int&) function and store the winner of the called war in a variable.
 If the player won the above war,
 Add both cards and both prize decks to the player’s deck.
 Else,
 Add both cards and both prize decks to the computer’s deck.
 Return the winner of the war.

Main()

Create an instance of your game engine.

Greet the player.

While the player wants to repeat the game,

 Deal the decks

 While the game has not been won

 Play a round

 Check for a winner

 Offer for the player to replay the game

Testing tip: Some games of war can go for a while. Commenting out the “cin.get()” line will make your program run until one side wins.

Sample Output:

<pre>Welcome to the game of WAR! What is your name? Matthew Landrum Welcome, Matthew Landrum! Let's play! Dealing Decks. Decks Dealt. Matthew Landrum's deck size: 26 Computer's deck size: 26 Matthew Landrum's card: K of diamonds Computer's card: 10 of clubs Matthew Landrum wins the round. Press enter to go on to the next round.</pre>	<pre>Computer wins the round. Press enter to go on to the next round. Matthew Landrum's deck size: 28 Computer's deck size: 24 Matthew Landrum's card: Q of clubs Computer's card: 6 of diamonds Matthew Landrum wins the round. Press enter to go on to the next round. Matthew Landrum's deck size: 29 Computer's deck size: 23 Matthew Landrum's card: 9 of diamonds Computer's card: 9 of hearts Matthew Landrum's card: 10 of spades Computer's card: 8 of hearts Matthew Landrum wins the war and 10 cards! Press enter to go on to the next round.</pre>	<pre>Matthew Landrum wins the round. Press enter to go on to the next round. Matthew Landrum's deck size: 51 Computer's deck size: 1 Matthew Landrum's card: J of clubs Computer's card: 10 of diamonds Matthew Landrum wins the round. You win! Good job! Would you like to play again? (yes/no) I would like to play again! Please enter "yes" or "no" Would you like to play again? (yes/no) no Thanks for playing!</pre>
---	---	--

Submit via Canvas:

Assign05.zip

Compressed Visual Studio Solution folder