Programming Exam Grading Criteria

Part 1 (3.5%)

A. Run the program associated with Part1, you should see a message similar to the one below waiting for the player 1 input:

Player 1 Bet - $

A.A Enter 20 as the bet input. **(0.5%)**

B. You should see a message saying:

Player 2 Agree?

Enter No as the input and the program should terminate. **(0.5%)**

C. Run the program again enter 30 as player 1 bet amount this time for Player 2 input Yes as the answer to the "Player 2 Agree?" question. A random card should be dealt to each of the players. Look through student's code and make sure they generate this card randomly and not hardcoded **(0.25%)**

The output should be something like this **(0.25%)** (notice the cards are assigned randomly, so the values may be different for your case):

Player 1 Card – J

Player 2 Card – 8

D. If Player 1 has a higher value card than Player 2 then Player 1 should be announced as the winner and Player 2 as the loser. However, if both players have the same card values then war should start. If war starts skip to E and come back to D later. If you are coming back from part E play the game until one player is the winner and the other is the loser. In this case one player should be announced the winner and the other the loser as follows: (Note your outputs will differ depending on who the winner is.) If a winner and loser are announced, give **0.4%.** If the correct winner and loser are announced, give another **0.6%.**

Player 1 wins $30.

Player 2 loses $30.

Thank you for playing war.

E. If both players did not have the same card values for part D, run the game for couple of times until the card values for both players become the same. Or you can find the place in the student's code where he/she assigned card values and hardcode same value cards. If you are coming from part D you are fine (Lucky you!). for "Ready to continue?" enter Yes. **(1.0%)**

Part 2 (2.5%)

A. There should be a README for this part. Students will lose points if they don't have the README **(0.5%)**

B. Depending on how they implemented the game (whether server is one of the players or server is a separate entity) create two Players (Player 1 and Player 2 by running two instances of players or one instance of player and one instance of the server

depending on the student's implementation) make sure the game is handled through networking. The output should be something similar to one below: **(0.5% + 0.5% for the output on each player's console)**

For Player 1 console when it joins the game:

Joins the game through the stand-alone program.

For Player 2 console you should see the following once Player 2 joins:

Joins the game through the stand-alone program.

C. Follow A.A through D from Part 1 with a bet amount of 70. check both consoles for both players to see if they have the printouts below **(0.5% no credits will be awarded if the game does not function the way it was supposed to)**

In case there is not a war (The card values would be different because the game is pseudo random):

Player 1	Player 2
Joins the game through the stand-alone program	
	Joins the game through the stand-alone program
Bets $70	
	Resigns from the game
	Joins the game through the stand-alone program
Bets $70	
	Accepts $70 bet
Draws J	
	Draws 8
Wins $70	Loses $70

In case there is a war (The card values would be different because the game is pseudo random):

Player 1	Player 2
Joins the game through the stand-alone program	
	Joins the game through the stand-alone program
Bets $70	
	Accepts $70 bet
Draws J	
	Draws J
Ready to continue?	Ready to continue?
yes	yes
Draws Q	
	Draws Q
Draws 2	
	Draws 3
Loses $70	Wins $70

D. Follow E from Part 1. Make sure there is a "Ready to continue?" output for both players in case of a war and put yes as input for both players. **(0.5%)**

Part 3 (4.0%)

A. Look for `war.html`. Student will lose points if they don't have a `war.html` **(0.25%)**

B. There should be a README file for this part. **(0.25%)**

C. Run an instance of Player 1 through the browser and run the second player through the stand alone program. Make sure they both join the game. Terminate both programs **(0.25%)**

D. Run an instance of Player 1 through the stand alone program and run the second player through the browser. Make sure they both join the game. **(0.5%)**

E. Now, the Player 1 should be prompted to enter a bet. Enter 40 as the bet amount. **(0.75%)**

F. Make Player 2 not agree with the bet amount, and the program should terminate. **(0.5%)**

G. Follow C through D from Part 1 make sure the bet amount is $40 this time and that the player 1 is run from the stand alone program and the Player 2 is from the browser **(1.0%)**

In case there is no war going on:

H. Player 1	Player 2
Joins the game through the stand_alone program	
	Joins the game through the browser
Bets $40	
	Accepts $40 bet
Draws J	
	Draws 8
Wins $10	Loses $10

In case ther is a war going on:

Player 1	Player 2
Joins the game through the stand-alone program	
	Joins the game through the web
Bets $40	
	Accepts $40 bet
Draws J	
	Draws J
Ready to continue?	Ready to continue?
yes	yes
Ready to continue?	Ready to continue?
yes	yes
Draws 2	
	Draws 3
Loses $40	Wins $40

I. Follow E from Part 1. Make sure there is a "Ready to continue?" output for both players in case of a war and put yes as input for both players. **(0.5%)**