Individual Assignment COMP 3004

Due: September 20th 2018 at 6AM on culearn

Blackjack (see https://en.wikipedia.org/wiki/Blackjack and the video https://www.youtube.com/watch?v=-9YGKFdP6sY) is a relatively simple card game and several source code versions of it in Java exist on the net.
The objective of the game is to beat the dealer in one of the following ways:

- Get a score of 21 points with the player's first two cards (called a "blackjack") *without* the dealer also obtaining a blackjack; or

- Reach a final score higher than the dealer without exceeding 21; or

- Let the dealer draw additional cards until the dealer's hand exceeds 21.

We will play with a single deck of 52 cards and only some of the rules.

Cards 2 to 10 are worth their face value in points. Aces are worth 1 or 11 points (whichever is best without *busting*, see below), and face cards (J, Q, K) are all worth 10 points. In our version, a card is to be designated by a one-letter suit (S for spades, C for clubs, D for diamonds, H for hearts) and a rank (K, Q, J, 10… 2, A). So 3 of clubs is denoted as C3, king of spades as SK.

We address one single game between a single player and an *automated* dealer.

The interface of the game is to be textual (unless you attempt the Bonus).

1) The game first prompts the player to select either console (c) or file (f) input.

If using console input, then the value of all cards used by the player and the dealer are assigned randomly out of the remaining cards of the deck (which starts complete and randomly shuffled).

If using file input, then the value of all cards used by the player *and* the dealer are read from a file, as well as all the commands of the player (as illustrated below).

2) Then the player and the dealer are each 'dealt' two cards (in this order).

Both cards of the player are visible (i.e., reported by the interface). However, only one of the two cards of the dealer is visible. (The other is "face down".)

3) Then the player plays:

At that point, in the simplest form of our limited version of blackjack, if neither the player or dealer has already won, then the player will have two options: Hit (H) or Stand (S).

If the player chooses to hit, s/he is dealt a new card and will then be allowed to choose between the hit and stand options again if their current score is 21 or less.

If the player *busts* (i.e., has a score over 21 with Aces worth 1) during their turn, the dealer automatically wins.

If a player chooses to stand their turn ends and all the cards of the players are visible.

4) The dealer's turn

The card of the dealer that was face down is then shown in the UI. The dealer will then play using typical dealer rules: if it has a score of 16 or less, or a soft 17, it will hit. A "soft 17" is a score of 17 in which 11 of the points come from an Ace card. Otherwise, it stands.

If console input has been selected, all cards received by the dealer are automatically and randomly chosen and displayed. Conversely, if file input has been chosen, the input file must contain the correct number of cards that the dealer's algorithm dictates based on the two initial cards specified in this file for the dealer. Cards read from file are to be displayed.

At the end of the turn of the dealer, all its cards are visible.

5) Determining the winner

Blackjack occurs when a player has an Ace with i) a face card (J, Q, K) or ii) a 10 card. If either the player or the dealer has a Blackjack upon receiving their initial two cards, they win. If both have a Blackjack, the dealer wins.

If the player or the dealer busts, the other participant wins.

Otherwise, the winner is the participant who has the highest score without going over 21. If both have the same score, the dealer wins.

Once a winner has been determined, it is announced via the interface (i.e., on the console) along with the score (i.e., value of the hand) of each participant.

Marking scheme

This first version of our simplified game of BlackJack is worth a maximum of 84 marks out of 100 for this assignment. If and only if this first version is functional and, most importantly, is developed using a TDD approach (see below), then:

Up to 10 additional marks are given for supporting Splitting (for the player *and* for the dealer): If the two initial cards of the player are identical (e.g., two Aces, two kings, two fives), then this player can Split (D), which gets him a second card on each of these two cards. Then that player can Hit or Stand on each of these two stacks of cards. More precisely, the player plays until standing or busting on the first identical card, then and only then plays on the second identical card. Effectively, the player gets two turns instead of one, playing on the first identical card, then on the second identical one. The game must determine which of these two hands results in the best score and compare that better hand against the hand of the dealer. If the dealer has two identical cards that add to 17 or less, the dealer necessarily splits dealing with one card, then the other (as the player does).

Up to 6 additional points are given for robustness (ie dealing with invalid inputs): 2 marks for console input, 4 marks for file input.

Bonus

If *and only if* you do follow a TDD approach (see below) *and* have a working and tested game that supports splitting, then providing a working Java/FX interface for it is worth an additional 25 marks. In other words, you can score up to 125 out of 100.

Evaluation

- You are to develop your code in a private Git repository and use Maven to set up dependencies. Your *assigned TA* must have access to this repository **as soon as** you create it in order to monitor your submissions. That is, you must send your name and Git credentials to your assigned TA upon creating your repository.
- You MUST follow a TDD approach:
    - We will consider that a method of a class is non-trivial if it captures *several* paths of execution. More precisely, any method that has a loop or an if statement is to be considered to be non-trivial.
    - Each execution *path* of each non-trivial method requires one or more JUnit tests to be submitted to the repository BEFORE the code for that path is submitted.
    - Failure to submit JUnit tests *before* submitting a non-trivial method will result in a significant deduction. Repeated failure to follow a TDD approach over several non-trivial methods will result in a mark of 0.
- You will submit a single zip file to culearn. It *must* be named <yourFullNameA1.zip> and must include all source and compiled code (including all tests) of your latest version of the game, as well as a filled out correction grid (to be posted by the instructor by Sept. 14th) that captures what functionality you do support and whether you wrote your code yourself from scratch or adapted it from code found on the Web (which *is* allowed in the context of this assignment).
- You are to contact your assigned TA *before* **September 15th** 6AM and schedule with him/her a short (max 10 minute) demo of your assignment on September 20th or 21st. You will receive a mark of zero for this assignment if

you do not schedule or do not attend your demo. **Do not** contact your assigned TA before I have posted his/her available times on those days.

Important advice: given the quick submission date, DO manage your time carefully!

Example input files

File1: SK HA HQ CA

Player receives the King of Spades and Ace of Hearts and thus has a blackjack

Dealer receives Queen of Hearts and Ace of Clubs and wins with a blackjack

File 2: SK HQ SQ C5 S DJ

Player receives the King of Spades and Queen of Hearts

Dealer receives the Queen of Spades and 5 of Clubs

Player stands

Dealer has 15 and thus must hit (notice no command in input file for the dealer)

Dealer gets Jack of Diamonds and busts: Player wins

File 3: S10 D3 SQ C5 H H5 H SA S CA D2

Player receives the 10 of Spades and 3 of Diamonds: hand value is 13

Dealer receives the Queen of Spades and 5 of Clubs: hand value is 15

Player hits and gets 5 of Hearts: hand value is 18

Player hits again and gets Ace of Spades, which must count as 1 otherwise the player would bust: hand value is 19. Player stands.

Dealer must hit and gets Ace of Clubs, which must count as 1 otherwise the dealer would bust: hand value is 16. Dealer must hit again and gets 2 of Diamonds. Hand value is 18 and dealer stands (being over 17).

Player wins.

File 4: Player splits
SK HK CQ D9 D H6 H D3 S C5 H D5 S
Player gets king of Spades and king of Hearts.
Dealer gets queen of Clubs and 9 of Diamonds: hand value is 19.
Player splits (D command) and gets 6 of hearts on king of spades.
Notice no hit required to get this second card.
Player hits (H command) gets 3 of diamonds: hand value is now 19. Player stands (S command).
Player gets 5 of clubs on king of hearts.
Again notice no Hit command necessary here: hand value is 15.
Player hits and gets 5 of diamonds: hand value is 20 for king of hearts.
Player stands. Best of its two hands is the one on king of hearts at a value of 20.
Dealer stands.
Player wins.

File 5: Dealer splits (Notice absence of H commands)
SK H9 C5 D5 S H7 CQ SA SQ D2
Player gets king of Spades and 9 of Hearts.
Dealer gets 5 of clubs and 5 of diamonds.
Player stands with a hand at 19.
Dealer gets 7 of hearts then queen of clubs and busts on the 5 of clubs hand
Then the dealer gets ace of spades on 5 of diamonds: hand value is 6 or 16
Then the dealer gets queen of spades: hand value is 16 (with ace necessarily at 1)
Then the dealer gets 2 of diamonds for a hand value of 18 and stops.
Player wins