Project 1: Card Game

NOTE: *If you turn in a solution to this problem based on something from a pay-for-answers website such as Chegg, CourseHero, etc., you will receive an F in this course (not just on this assignment). Start early and seek help from a lab assistant, the help room, or your professor if you need it. If you don't have time to do that, it is better to get a low grade on the project than to cheat and fail the course.*

This program involves one turn in a very simple card game. In this game, each player plays a card, and the card with the highest point value wins.

The input to your program will have the following form:

<rank1><suit1><space><rank2><suit2><space><rank3><suit3>

The rank should be either a number between 2 and 10, inclusive, a J for a Jack, a Q for a Queen, a K for a King, or an A for an Ace. If it is anything other than this, you should output a message saying "<rankX><suitX> has an invalid rank" and the program should terminate.

The suit should be either a C for clubs, an H for hearts, a D for diamonds, or an S for spades. Again, if the user enters anything other than this, you should output a message saying "<rankX><suitX> has an invalid suit" and the program should terminate. (If both the rank and suit are invalid, you only need to report one of the problems before terminating.)

For example, if the user enters `AH 3D 10S`, that equates to the ace of hearts, the three of diamonds, and the ten of spades.

Your program does not need to check for impossible situations, like the 4 of hearts appearing twice.

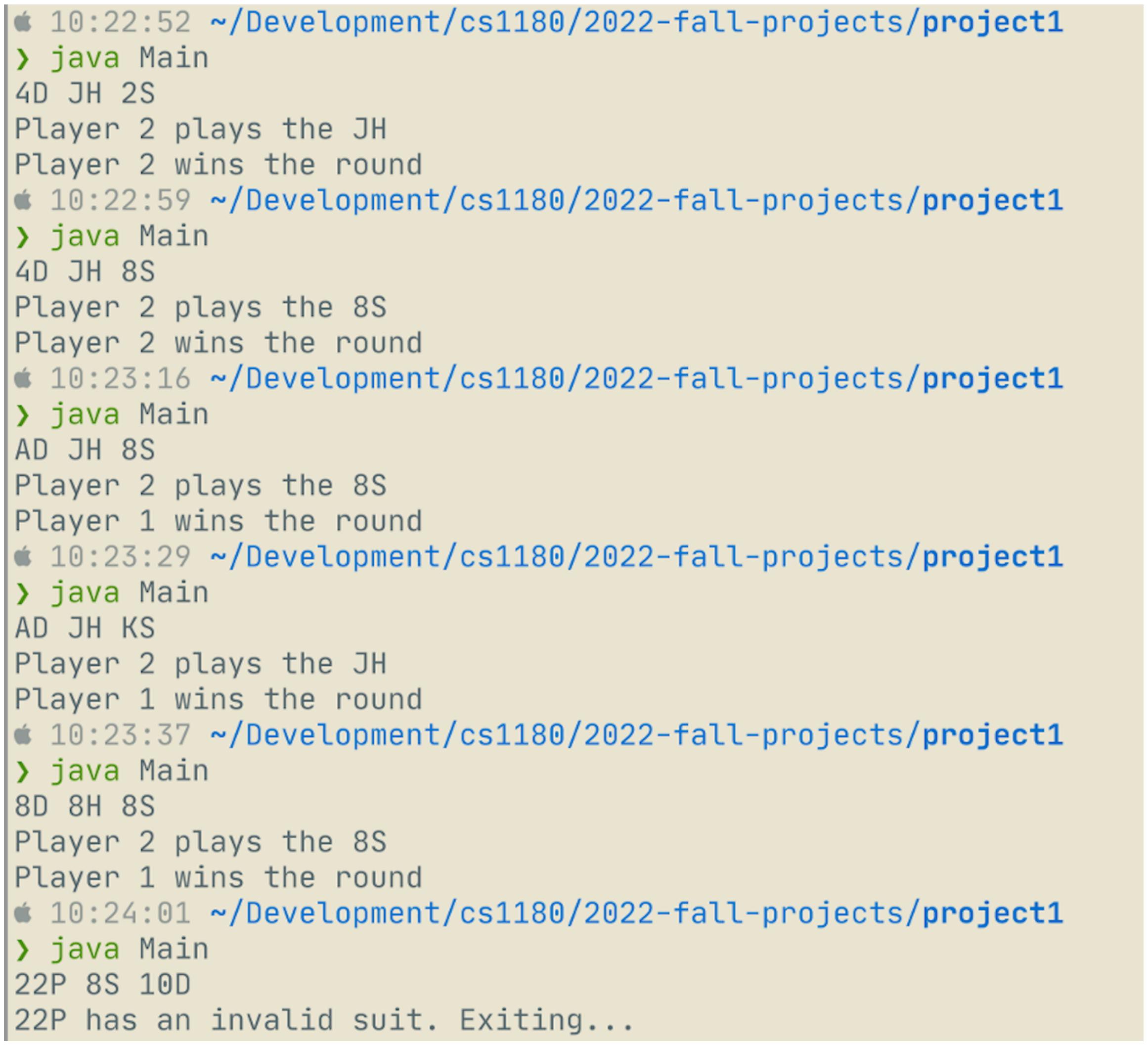
The first card from the input belongs to Player 1, and the second two cards belong to Player 2. A player wins by playing a card with a higher point value than their opponent. The point value for cards with rank 2 through 10 is just the rank itself. The point value of a jack is 10, a queen is 11, a king is 12, and an ace is 13.

Your task is to determine which card Player 2 should play in response to Player 1's card, according to the following strategy:

- Player 2 should win by the smallest amount whenever possible.
- If Player 2 cannot win, they should lose by the largest amount possible.

Your program should state which card Player 2 plays, and which player won the round. If the point value of both players is the same, Player 1 wins since they went first.

Examples:



```
 10:22:52 ~/Development/cs1180/2022-fall-projects/project1
❯ java Main
4D JH 2S
Player 2 plays the JH
Player 2 wins the round
 10:22:59 ~/Development/cs1180/2022-fall-projects/project1
❯ java Main
4D JH 8S
Player 2 plays the 8S
Player 2 wins the round
 10:23:16 ~/Development/cs1180/2022-fall-projects/project1
❯ java Main
AD JH 8S
Player 2 plays the 8S
Player 1 wins the round
 10:23:29 ~/Development/cs1180/2022-fall-projects/project1
❯ java Main
AD JH KS
Player 2 plays the JH
Player 1 wins the round
 10:23:37 ~/Development/cs1180/2022-fall-projects/project1
❯ java Main
8D 8H 8S
Player 2 plays the 8S
Player 1 wins the round
 10:24:01 ~/Development/cs1180/2022-fall-projects/project1
❯ java Main
22P 8S 10D
22P has an invalid suit. Exiting...
```

Rubric:

Note: Code that does not compile will receive a zero.

[15 pts] Correctly identifies invalid ranks for any of the three cards

[15 pts] Correctly identifies invalid suits for any of the three cards

[20 pts] Correctly establishes the point value for a particular rank (i.e. a ten is 10, a jack is 11, an ace is 14, etc.)

[20 pts] Correctly determines which card to play according to the described strategy.

[20 pts] Correctly reports which player won the round.

[10 pts] The program is clearly organized, commented, and follows standard coding practices, including using descriptive/meaningful variable and class names.