Problem: A typical deck of cards has 52 different cards. The cards are divided among four suits and each suit has numbered cards 2 through 10 along with a Jack, Queen, King, and Ace. The numbered cards are worth their face value. Jack, Queen, and King will be considered worth 11, 12, and 13 points respectively. The Ace is only one point.

Given two numbers in the range of 1 to 52 determine whether the first card, possessed by player #1 is greater than, less than, or equal to the card of player #2. If there is a winner then that player receives the point total of the two cards combined, otherwise, each player receives the point total of their own card.

The first 13 cards are of the Spades suit where 1 is the Ace, 2-10 are the numbered cards, and 11-13 are the Jack, Queen, and King. The next 13 cards belong to the Diamonds suit, the third 13 to the Clubs suit, and the final 13 to the Hearts suit.

You must validate that the user has entered a valid card (1 to 52) and that the second player's card is not the same as the card selected by the first player.

Example Execution #1:

Enter Player 1 card: 5
Enter Player 2 card: 9

Player 1's card 5 of Spades is less than Player 2's card 9 of Spades
Score: Player 1 - 0, Player 2 - 14

Example Execution #2:

Enter Player 1 card: 25
Enter Player 2 card: 24

Player 1's card Queen of Diamonds is greater than Player 2's card Jack of Diamonds
Score: Player 1 - 23, Player 2 - 0

Example Execution #3:

Enter Player 1 card: 50
Enter Player 2 card: 11

Player 1's card Jack of Hearts is equal to Player 2's card Jack of Spades
Score: Player 1 - 11, Player 2 - 11

Example Execution #4 (input validation demonstrated):

Enter Player 1 card: 55

Error - please enter a value between 1 and 52!

Enter Player 1 card: 27
Enter Player 2 card: 26

Player 1's card Ace of Clubs is less than Player 2's card King of Diamonds
Score: Player 1 - 0, Player 2 - 14

Academic Integrity Reminder:

- Please review the policies of the course as they relate to academic integrity. The assignment you submit should be your own original work. You are to be consulting only course staff regarding your specific algorithm for assistance. Collaboration is not permitted on individual homework assignments.
**Example Execution #5 (more input validation demonstrated):**

Enter Player 1 card: 38  
Enter Player 2 card: 38  

Error - the two players cannot have the same card!  

Enter Player 2 card: 0  

Error - please enter a value between 1 and 52!  

Enter Player 2 card: 40  

Player 1's card Queen of Clubs is greater than Player 2's card Ace of Hearts  
Score: Player 1 - 13, Player 2 - 0  

**Additional Requirements:**

1. Add the `head_hw` assignment header file to the top of your program. A description of your program will need to be included in the assignment header. This particular header can be added to your file by entering `hhw` while in command mode in `vi`.

2. For this assignment you will be **required** to implement the user-defined functions (from chapter 4).  
   - Failing to follow course standards as they relate to good user-defined function use will result in a **zero for this assignment**.  
   - Revisit **course standards** as it relates what makes for good use of user-defined functions, what is acceptable to retain in the main function, and when passing parameters by address is appropriate.

3. **The example executions provided for your reference each represent a single execution of the program.**  
   - Your program must **accept AND validate input** and **produce output exactly** as demonstrated in each example execution. The user should be given an **unlimited number of opportunities** to enter valid data.  
   - Your program will be tested with the data seen in the example execution and an unknown number of additional tests making use of reasonable data.

4. **Course standards prohibit the use of programming concepts not yet introduced in lecture.** For this assignment you can consider all material in the first SIX chapters of the book, notes, and lectures to be acceptable for use.

5. A program **MUST** compile to be considered for partial credit. The submission script will reject the submission of any file that does not compile.

**Course Programming and Documentation Standards Reminders:**

- Use the course function header (`head_fx vi shortcut hfx while in command mode`) for every user-defined function in your program.  
  - List and comment **all parameters** to a function, one per line, in the course function header.  
  - **All function declarations** will appear in the global declaration section of your program.  
  - The user-defined function definitions will appear in your program after the main function.

- At no point during the semester should the two sections of a function (local declarations, executables) overlap.  
- Place a **single space** between all operators and operands.  
- Comment **all** variables, declared in the local declaration section of a function, to the right of each declaration statement. Declare only one variable per line.

- The use of `{ and }` are required for all relevant selection and repetition constructs.  
- Indent all code two additional spaces that is found inside the body of relevant selection and repetition constructs.
When you submit... only the final attempt of a submission is kept for grading. All other submissions are over-written and cannot be recovered. You may make multiple submissions but only the last attempt is retained and graded.

- Verify in the confirmation e-mail sent to you by the course that you have submitted the correct file, to the correct assignment (hw04), and to the correct lab section.
- Leave time prior to the due date to seek assistance should you experience difficulties completing or submitting this assignment.
- All attempts to submit via a method other than through the guru server as set up during the first week of the semester will be denied consideration.

Assignment deadlines... are firm and the electronic submission will disable promptly as advertised. We can only grade what you submit as expected prior to the assignment deadline.

All course programming and documentation standards are in effect for this and each assignment this semester. Please review this document!