Question #1 - What do machines need to understand language [20]
Find a short dialog involving a robot from popular culture. This can be a movie, tv show, or book.

Describe in as much detail as you can, what capabilities the robot needs to have to 1) understand its conversational partner and 2) generate its responses.

Note that this question is worth 20% of the assignment so be thorough.

Question #2 - Regular Expression (J&M 2.1) [30 - 5pts each]
1. Write regular expressions for the following languages. By ‘word’, we mean an alphabetic string separated from other words by white space, any relevant punctuation, line breaks, etc.

2. the set of all alphabetic strings.

3. the set of all lowercase alphabetic strings ending in a b.

4. the set of all strings with two consecutive repeated words (for example ‘Humbert Humbert’ and ‘the the’ but not ‘the bug’ or ‘the big bug’).

5. the set of all strings from the alphabet a b such that each a is immediately preceded and immediately followed by a b.

6. all strings which start at the beginning of the line with an integer (i.e. 1,2,3...10...10000...) and which end at the end of the line with a word.

Question #3 - Regular Expression (J&M 2.8) [10]
Write a regular expression for the language accepted by the NFSA below.
Question #4 - ELIZA (J&M 2.2) [40]
Implement an ELIZA-like program, using substitutions such as those described in Chapter 2. You may choose a different domain than a Rogerian psychologist, if you wish, although keep in mind that you would need a domain in which your program can legitimately do a lot of simple repeating-back. Give your version of ELIZA a distinct, recognizable and describable personality. Some examples would be "sarcastic" or "insulting", "silly", "polite", etc.

This program must be callable from the command line.

User input must be read from standard in.

Machine output must be written to standard out.

The implementation must follow the guidelines on the course website. The submission must include a README file which outlines 1) how to run the assignment, 2) how to compile the assignment (if necessary), 3) any required packages, 4) any features that you would like the grader to be made aware of.

The grading breakdown of this section is as follows:

* (10) Correctness - does the program appropriately respond to user input?

* (10) Functionality - does the program work as described? Does the program compile correctly and without error?

* (5) README - Does the attached README file include all appropriate documentation.

* (5) Within code documentation - Are all functions commented to describe what they do and appropriate inputs and outputs? Are non-trivial sections of code appropriately documented?

* (5) Style - is the code well written? Do functions and variable names have meaningful names? are spacing and other stylistic aspects consistent?

* (5) Grader's Discretion - Does the assignment go "above and beyond" in some way, either by effectively expressing the intended personality, especially well written and designed code, or some other feature not required by the assignment.