Homework 2

1. Write regular expressions to capture the following regular languages:

   (a) The set of binary strings which have an even number of 0's or which contain three consecutive 1's somewhere in the string (or both). Hint: this is an OR, so build the two regular expressions separately and then put an OR between them!

   (b) All strings of lower case letters which contain the following 5 letters (in this order): a b c d e. Note that any other letters can appear before, after, or between (or whatever); I just want at least one of each of these letters, appearing in this relative order somewhere when scanning across the input string. You're also welcome to use ASCII shorthands; for example, you can use [m-x] to represent any lowercase letter between m and x (inclusive).

   (c) Comments as in Python: a # followed by any character up to (and including) the next newline, or triple double quotes with any number of lines in between:

   ```
   """ This is a comment block.

   It takes up multiple lines and can be used

   in any place in your code. """
   ```

   Again, feel free to use shortcuts for letters or sets of characters if it will simplify - just define them carefully so I know what you mean.

2. Write a DFA or NFA to recognize each of the languages described in each part of problem 1.

3. Note: for this problem, I'm NOT asking for just any NFA! I want the one that results from that standard algorithmic construction, so you have to go review this in chapter 2. Random NFAs will not receive credit, even if they accept the correct set of strings.

   (a) Show the NFA that results from applying the standard construction we saw in class (or you can find in the book in Figures 2.7, and 2.8) to the regular expression:
   letter (*digit* | *letter*)* *letter*.

   (b) Convert your NFA to a DFA (see Figure 2.9 in the book). Note that you are NOT required to minimize (as in Figure 2.10), although I suppose you are welcome to if you want!