0. Write a regular expression for the language of symbolic Internet domain names, such as cs.washington.edu. Numbers, letters (upper and lowercase) and dash are legal, but starting or ending with a dash is not allowed.

1. Convert the following regular expression to a NDFA:
   \[(bcd|a*cd)* | x* | xaxcd\]

2. Convert the NFA into a DFA, following the algorithm from class. Be sure to label the NFA states and to label each of the DFA states with a set of NFA states.

3. In the first lecture an intermediate compiler representation called “three address code” was mentioned. For example,
   
   \[
   \begin{align*}
   t0 & := 1; \\
   t1 & := num < t0; \\
   \text{ifnonzero } t1 & \text{ goto L0;} \\
   t2 & := 1; \\
   t3 & := num - t2; \\
   t6 & := 4; \\
   t7 & := num * t6; \\
   \text{numAux} & := t7; \\
   \text{goto } L2;
   \end{align*}
   \]

   The language should be pretty obvious. There are temporary registers, e.g. \( t<integer> \), normal identifiers from the source language, e.g. \( \text{num} \), key words, e.g. \( \text{ifnonzero} \), and operators. Write the lexical grammar for this language (using the extended rules from class is OK). You only need to handle symbols occurring in this example.

Produce a hard-copy of your answers and turn them in by the start of class on the due date. Do these exercises individually, not with your project partner.