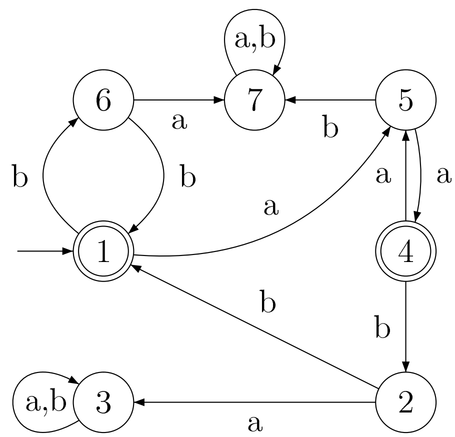
CS 273: Intro to Theory of Computation, Spring 2008
Problem Set 7
Due Monday, March 3rd, 4pm.

This homework contains four problems. Please submit each on a **separate sheet of paper**. This will help us grade your homeworks more quickly. Turn in your homework at Elaine Wilson's office (3229 Siebel).

1. Suffix languages.

   Consider the following DFA:



   (a) Write down the suffix language for each state.

   (b) Draw a DFA that has the same language as the one above, but has the minimal number of states.

2. Context-free grammar design

   Give context-free grammars generating the following languages:

   (a) $L_1 = \left\{ \mathtt{a}^n \mathtt{b}^p \,\middle|\, 0 < p < n \right\}$.

   (b) $L_2 = \{\ a^n b^n c^m d^m \mid n, m \in \mathbb{N}\ \}$

   (c) $L_3 = \left\{ \mathtt{a}^n \mathtt{b}^m \mathtt{c}^p \,\middle|\, n = m \text{ or } m = p \right\}$.

   (d) $L_4 = \mathtt{a}(\mathtt{ab}^*)^*$.

3. Context-free grammar interpretation.

(a) What is the language of this grammar? The alphabet is $\{\mathtt{a}, \mathtt{b}, \mathtt{c}, \mathtt{d}\}$ and start symbol is T.

$$S \rightarrow \mathtt{a}S\mathtt{b} \mid \epsilon$$
$$T \rightarrow S \mid \mathtt{c}\,T \mid T\,\mathtt{d}$$

(b) Answer the same question for this grammar, with same alphabet and start symbol.

$$S \rightarrow \mathtt{a}S\mathtt{b} \mid \epsilon$$
$$T \rightarrow S \mid \mathtt{c}S \mid S\mathtt{d}$$

(c) Answer the same question for this grammar, with same alphabet and start symbol.

$$S \rightarrow T\,\mathtt{b}$$
$$T \rightarrow \mathtt{aa}S \mid \mathtt{cd}$$

4. NFA pattern matching.

Pattern-search programs take two inputs: a pattern given by the user and a file of text. The program determines whether the text file contains a match to the pattern, typically using some variation on NFA/DFA technology. Fully developed programs, such as `grep`, accept patterns containing regular-expression operators (e.g. union) and also other convenient shorthands. Our patterns will be much simpler.

Let's fix an alphabet $\Sigma = \{\mathtt{a}, \mathtt{b}, \ldots.\mathtt{z}, \sqcup\}$. Let $\Gamma = \Sigma \cup \{?, [,], *\}$. A **pattern** will be any string in Γ^*.

A string w matches a pattern p if you can line up the characters in the two strings such that:

- When p contains a character from Σ, it must be paired with an identical character in w.
- The character ? in p can match any substring x in w, where x contains at least one character.
- When p contains a substring of the form $[w]*$, this can match zero or more repetitions of whatever w matches.

For example, the pattern "`fleck`" matches only the string "`fleck`". The pattern "`margaret?fleck`" will match anything containing "`margaret`" and "`fleck`", separated by at least one character. The pattern "$\mathtt{i} \sqcup \mathtt{ate} \sqcup [\mathtt{many}\sqcup] * \mathtt{donuts}$" matches strings like

"$\mathtt{i} \sqcup \mathtt{ate} \sqcup \mathtt{donuts}$" and

"$\mathtt{i} \sqcup \mathtt{ate} \sqcup \mathtt{many} \sqcup \mathtt{many} \sqcup \mathtt{donuts}$"

Instances of []* can be nested. So the pattern $\texttt{cc[bb[a]} * \texttt{bb]} * \texttt{dd}$ matches strings like `ccdd` or `ccbbaaaaabbdd` or `ccbbabbbbabbdd`.

A text file t contains a match to a pattern p if t contains some substring w such that w matches p.

Design an algorithm which converts a pattern p to an NFA N_p that searches for matches to p. That is, the NFA N_p will read an input text file t and accept t if and only if t contains a match to p. N_p searches for only one fixed pattern p. However you must describe a general method of constructing N_p from any input pattern p.

You can assume that your input pattern p has been checked to ensure that it's well-formed and that we have a function m which matches open and close brackets. For example, you can assume that an open bracket (]) at position i in the pattern is immediately followed by a star (*). You can also assume that there is a matching open bracket ([) at position $m(i)$ in the pattern. The function m is a bijection, so if there is an open bracket at position j in the pattern, $m^{-1}(j)$ returns the corresponding close bracket.