1. A context-free grammar is linear if every rule has at most one variable on the right hand side. A language is linear if it is generated by a linear grammar.
   (a) Show that the regular languages form a proper subset of the linear languages.
   (b) Is every linear language a DCFL? Answer yes or no, and explain. (If you answer yes, show why. If you answer no, give an example.)
2. I modify the notion of PDA to provide one more ‘stack’ operation. As well as pushing and popping, a transition is permitted to turn the stack upside down (it cannot push in the same transition if it does so). Call this a flip-PDA.
   (a) Give a formal definition of a flip-PDA, particularly its transition function. (This is asking for a general definition, not an example.)
   (b) Show that there is a language that is recognized by a flip-PDA but not recognized by a PDA.
3. Describe in detail a deterministic Turing machine to recognize \{0^a1^b2^c : c is the remainder when a is divided by b\}.
4. A regular expression pattern matcher needs to check whether a string matches a regular expression and answer yes or no. Suppose that we represent a problem of this type as \(R \# w\), where \(R\) is a regular expression, \(\#\) is a symbol not used in the alphabet of the regular expression, and \(w\) is a finite string. Then define the language \(L_{re} = \{R \# w : w \text{ is in the language described by } R\}\). Give a description of a (deterministic or nondeterministic) Turing machine that recognizes \(L_{re}\) and always halts (so that it decides \(L_{re}\)). You do not need to detail the individual transitions, but provide enough detail so that a not-too-clever programmer could write out the detailed specification of the TM.
5. (not to be graded) Think of the poor grader of the homework assignments. S/he has to look at 82 different DFAs and for each to decide whether it accepts the language that it should. S/he has a correct DFA, but most of the student submissions look quite different. S/he wants to automate the process of deciding whether a student submission recognizes the same language as the sample solution. Can this be done by a Turing machine?

   More precisely, we encode each DFA \(M\) as a string \(\langle M \rangle\). The details of the encoding are not important, but we suppose that we can find each of \((Q, \Sigma, \delta, q_0, F)\) in the encoding, and can find individual states, input symbols, and transitions within these.

   Then we want to decide membership in the language \{\(\langle M \rangle \# \langle M' \rangle : M \text{ and } M' \text{ encode DFAs and } L(M) = L(M')\}\}. Can a deterministic TM decide this? (Hint: \(L(M) = L(M')\) if and only if \((L(M) \cap \overline{L(M')}))) \cup (L(M) \cap L(M')) = \emptyset\). Can the TM construct a DFA \(M''\) whose language is \((L(M) \cap \overline{L(M')}) \cup (L(M) \cap L(M'))\)? Can it then decide whether \(L(M'') = \emptyset\)?