Read all of Chapters 3 and 4.

0.1 Read and solve, but do not turn in: Book, 2.16. \[\text{CFLs closed under } \cup, \circ, \ast\]
Solve by using both CFLs and PDAs.

0.2 Read and solve, but do not turn in: Book, 2.18. \[\text{CFL } \cap \text{ regular } = \text{CFL}\]
You can check your solution with the one in the book.

0.3 Read and solve, but do not turn in: Book, 2.26. [Chomsky normal form]

1. Let \(\Sigma = \{0, 1\}\) and let \(C_2 = \{ztz \mid z \in 0^*\text{ and } t \in 0^*10^*10^*, \text{ where } |t| = |z|\}\).
   
   (a) Show that \(C_2\) is not a CFL.
   
   (b) Is \(C_2 \cup (\Sigma \Sigma \Sigma)^*\) a CFL? Why or why not?
   
   (c) Is \(C_2 \cup \Sigma (\Sigma \Sigma \Sigma)^*\) a CFL? Why or why not?

2. Let \(G = (V, \Sigma, R, \langle \text{stmt} \rangle)\) be the following grammar. \(\Sigma = \{\text{if, condition, then, else, a:=1}\}\), \(V = \{\langle \text{stmt} \rangle, \langle \text{if-then} \rangle, \langle \text{if-then-else} \rangle, \langle \text{assign} \rangle\}\) and the rules are:
   
   \[
   \begin{align*}
   \langle \text{stmt} \rangle & \rightarrow \langle \text{assign} \rangle \mid \langle \text{if-then} \rangle \mid \langle \text{if-then-else} \rangle \\
   \langle \text{if-then} \rangle & \rightarrow \text{if condition then } \langle \text{stmt} \rangle \\
   \langle \text{if-then-else} \rangle & \rightarrow \text{if condition then } \langle \text{stmt} \rangle \text{ else } \langle \text{stmt} \rangle \\
   \langle \text{assign} \rangle & \rightarrow \text{a:=1}
   \end{align*}
   \]
   
   (a) Show that \(G\) is ambiguous.
   
   (b) Give a new unambiguous grammar that generates \(L(G)\).
      (You do not need to prove that your grammar works or that it is unambiguous, but please add a few comments about why it does work to help the grader.)

3. A queue automaton is like a push-down automaton except that the stack is replaced by a queue. A queue is a tape allowing symbols to be written only on the left-hand end and read only at the right-hand end. Each write operation (we’ll call it a push) adds a symbol to the left-hand end of the queue and each read operation (we’ll call it a pull) reads and removes a symbol at the right-hand end. As with a PDA, the input is placed on a separate read-only input tape, and the head on the input tape can move only from left to right. The input tape contains a cell with a blank symbol following the input, so that the end of the input can be detected. A queue automaton accepts its input by entering a special accept state at any time. Show that a language can be recognized by a deterministic queue automaton iff the language is Turing-recognizable.

4. Show that a language is decidable iff some enumerator enumerates the language in string order. (String order is the standard length-increasing, lexicographic order, see text p 14).

5. Let \(C\) be a language. Prove that \(C\) is Turing-recognizable iff a decidable language \(D\) exists such that \(C = \{x \mid \exists y \in \{0, 1\}^* (\langle x, y \rangle \in D)\}\). (Hint: You must prove both directions of the “iff”. The \((\leftarrow)\) direction is easier. For the \((\rightarrow)\) direction, think of \(y\) as providing additional information that allows you to confirm when \(x \in C\), but without the possibility of looping.)

6. Consider the problem of testing whether a pushdown automaton ever uses its stack. Formally, let \(PUSHER = \{\langle P \rangle \mid P \text{ is a PDA that pushes a symbol on its stack on some (possibly non-accepting) branch of its computation at some point on some input } w \in \Sigma^*\}\). Show that \(PUSHER\) is decidable. (Hint: Use a theorem from lecture to give a short solution.)

7* Let the rotational closure of language \(A\) be \(RC(A) = \{yx \mid xy \in A \text{ where } x, y \in \Sigma^*\}\).
Show that the class of CFLs is closed under rotational closure.