- Start as early as possible, and contact the instructor if you get stuck.
- See the course outline for details about the course's marking policy and rules on collaboration.
- Submit your completed solutions to **Crowdmark**.

1. A pushdown automaton

[6] Let $\Sigma = \{0, 1\}$. Construct a pushdown automaton, P, which accepts exactly the language

$$L = \{w \in \Sigma^* \mid \text{every prefix of } w \text{ has at least as many 0s as 1s}\}.$$

State explicitly whether P accepts by final state or by empty stack. Argue informally why your P accepts exactly L.

2. Building a context-free grammar from a PDA
   Define the pushdown automaton, $P = (Q, T, \Gamma, \delta, q, X)$ (which accepts by empty stack), with
   - $Q = \{q\}$
   - $T = \{0, 1\}$
   - $\Gamma = \{0, 1, X\}$
   - q = start state for machine
   - X = stack start letter

   and transition function
   1. $\delta(q, \varepsilon, X) = \{(q, \varepsilon), (q, 0X1)\}$
   2. $\delta(q, 0, 0) = \{(q, \varepsilon)\}$
   3. $\delta(q, 1, 1) = \{(q, \varepsilon)\}$

[2] (a) Draw a diagram for P.

[4] (b) Use the technique described on slides 51-54 of Module 6 to construct a context-free grammar, G, such that $L(G) = N(P)$. In your final grammar, replace the non-terminals from the construction with single capital letters $A, B, C, \ldots$. Simplify your grammar as much as possible after you have completed the construction. You do **not** have to prove that $L(G) = N(P)$. But you should convince yourself that the equality holds once you have completed the construction and simplification of G.

3. The analog of Kleene's Theorem for CFLs and PDAs

[8] Let Σ be an alphabet. For any language L over Σ, Define

$$SUFFIX(L) = \{y \mid xy \in L, \text{ for some string } x \in \Sigma^*\}.$$

Prove that the class of context-free languages over Σ is closed under the $SUFFIX$ operation.

4. Deterministic PDAs and DCFLs

[6] Prove that the language

$$L = \left\{ x!y \mid x \neq y^R \text{ and } x, y \in \{0,1\}^* \right\}$$

is a DCFL, by giving a natural DPDA for it and explaining how that DPDA works. Your machine should accept by final state.

5. Recall from Slide 18 of Module 5, a grammar for words having a balanced number of 0s and 1s:

$$G : S \to \varepsilon \mid 0S1 \mid 1S0 \mid SS.$$

[8] (a) Use the technique on slides 6-17 of Module 7 to produce a grammar, G', in Chomsky Normal Form, such that $L(G) = L(G') \cup \{\varepsilon\}$. You do **not** need to prove this equality of languages: following the algorithm correctly will guarantee this fact.

[4] (b) Give an explicit derivation, in G', for the word: 0101.