You have 70 minutes.

When drawing an automaton, adopt the following convention, to reduce clutter: If there is no transition out of a given state for a given input or stack symbol, then the machine rejects the input string, when it is in that state and sees that input or stack symbol.

You may cite without proof any theorem proved in class, in the assigned sections of the textbook, or in the assigned homework. You may not cite other results without proof.

If your solution to a problem is a straightforward application of a procedure discussed in class, then it may need no explanation. On the other hand, if it is novel, difficult to understand, or incomplete then it probably requires explanation. If you are unsure, then err on the side of explaining too much, or ask for clarification.

When there are multiple answers to a problem, it is understood that a simpler or more efficient answer may earn more credit.

Good luck.

A. If the language is regular, then draw a DFA for it. If the language is not regular, then prove so; it will be context-free, so draw a PDA for it and write a CFG for it. The language is $A = \{w : \text{there are more } a\text{s than } b\text{s in } w\} \subseteq \{a, b\}^*$.

B. If the language is regular, then draw a DFA for it. If the language is not regular, then prove so; it will be context-free, so draw a PDA for it and write a CFG for it. The language is $A = \{w : |w| \geq 2 \text{ and } w \text{ ends in } ab\} \subseteq \{a, b\}^*$.

C. In this problem, you will design a regular expression (RE), in the syntax of our textbook, to match URLs. The alphabet is $\Sigma = \{\mathtt{a}, \ldots, \mathtt{z}, \mathtt{A}, \ldots, \mathtt{Z}, \mathtt{0}, \ldots, \mathtt{9}, \mathtt{\_}, \mathtt{.}, \mathtt{/}, \mathtt{:}, \mathtt{\#}\}$. Here is a typical URL:

`http://www2.w3c.org:1080/apps/scrapple/index.html#apes`

The `http` is a *protocol*, which consists of one or more letters. It is followed by `://`. The `www2.w3c.org` is a *hostname*, which consists of a sequence of at least two nonempty alphanumeric strings, separated by periods, with the last string purely letters (no digits). Write an RE H for hostnames. (You may define auxiliary REs, if doing so aids clarity.)

The `:1080` is a *port*, which consists of a colon followed by one or more digits. The string `/apps/scrapple/index.html` is a *path*, which consists of letters, digits, `_`s, `.`s, and `/`s. The path must begin with a `/`, and no two `/`s can touch. Write an RE P for paths.

The `#apes` is a *fragment*, which consists of `#` followed by a nonempty string of letters, digits, periods, and underscores. In any URL, the port and path are optional; the fragment is optional, but may occur only if a path occurs. The other parts of the URL are required. Write an RE for URLs, in terms of H, P, and any other auxiliary regular expressions that you care to define.

D. Let $A = \{a^i b^j :$ exactly one of i, j is a multiple of 2, and exactly one of i, j is a multiple of $3\} \subseteq \{a, b\}^*$. Prove that A is regular, or prove that A is not regular.

E. Prove that no infinite subset of $\{a^n b^n : n \geq 0\} \subseteq \{a, b\}^*$ is regular.