Some additional practice on CFG

(Based on Past Homeworks – Weeks 9-11)

50.051 Programming Language Concepts

Task 1: A simple Context-Free Grammar

Consider the following context-free grammar below, where ε denotes the empty string.

M → a M a

M → b M b

M → a

M → b

M → ε

1. What are acceptable strings according to this CFG?
2. Derive the "abba" using this grammar and build a parse tree for this derivation.

Task 2: Another Simple Context-Free Grammar.

Consider the CFG below, where S, T and U are non-terminal symbols and numbers are terminal symbols. For this CFG, S is the start symbol.

S → S - T | T

T → T + U | U

U → (any number)

A. This CFG defines a precedence order between the addition and subtraction operations. Which operation has higher priority? Would this CFG be correct mathematically-speaking?
B. Prove the precedence order you have identified for the CFG above, by establishing the derivation for a well-chosen expression of your choice.

Task 3: Some more Context-Free Grammars.

A. Consider the Context-Free Grammar (CFG) below, where S is the only non-terminal (and start) symbol, and terminal symbols are {x, y}. Here, ϵ denotes the empty string. How would you describe the strings accepted by this CFG?

$$S \rightarrow xSyS \mid ySxS \mid \epsilon$$

B. The CFG above cannot be replaced with a simple RegEx. Briefly explain why (no extensive proof required).

Task 4: A Simple Context-Free Grammars for Boolean Expressions.

Consider the Context-Free Grammar for Booleans in Python, below.

$$E \rightarrow B$$

$$B \rightarrow B \text{ and } B$$

$$B \rightarrow B \text{ or } B$$

$$B \rightarrow \text{not } B$$

$$B \rightarrow \text{True}$$

$$B \rightarrow \text{False}$$

1. According to this CFG, is the expression “False or not True” a valid syntax? If so, what is the parse tree for the derivation?

2. In Python, what is the precedence order between “and”, “or” and “not” operations?

3. In Python, what is the associativity for the “not”, “and” and “or” operations?

4. ~~Your friend Chris suggests using a rightmost top-down parsing algorithm, relying on rightmost BFS. He claims this will work 100% of the time, for any valid syntax string. Is he right to think so? If not, how would you explain his mistake to him?~~ (something for later!)

5. This CFG has a few problems, as highlighted above. What would you suggest doing to fix this CFG? Show your proposed CFG that fixes those issues.