CS323 Written Assignment 3

1 Requirements

You are expected to complete all required homework exercises. For submission, please put all your answers in a single PDF file and submit it via the assignment channel on BlackBoard. The name of the file should follow the format **"studentID_HW#"** (e.g., 30003554_HW1). **The submission deadline is 10:00 PM, November 2, 2025.** Late submissions are allowed within three days after the deadline (grace period). If you submit your assignment during the grace period, your score will be 60% of the score you could get if the submission was made in time. Assignments submitted after the grace period will not be graded.

2 Exercises (100 points)

Exercise 1: Given the alphabet $\Sigma = \{a, b\}$ and the regular language $L_1 = L((a|b) * ab)$:

- Construct a context-free grammar G to represent the language. [10 points]
- Given the string *ababab*, which is a valid sentence in the language $L(G)$, give the leftmost derivation and the rightmost derivation of the string? [10 points]
- Provide a parse tree for the above string. [5 points]

Exercise 2: The following grammar is known to be ambiguous:

```
S -> if E then S | if E then S else S | other
E -> id
```

Please demonstrate the ambiguity by drawing two different parse trees for the string: `if id then if id then other else other`. [10 points]

Exercise 3: Please design a grammar for the following language: the set of strings of 0's and 1's that are palindromes, that is, the strings read the same forwards and backwards (such as 0, 1, 010, 101). [20 points]

We provide the grammar for the language $L = \{a^n b^n, n \geq 0\}$ below to help you think of the production rules:

```
S -> ε
S -> aSb
```

Explanation: The first rule allows for the generation of the empty string, which trivially has an equal number of a's and b's. The second recursive rule ensures that for each a added to the string, a b is added somewhere after it. This guarantees that any string generated by the rule will have an equal number of a's and b's, with all a's preceding all b's.

Note: In addition to providing the grammar, please also explain why it makes sense.

Exercise 4: Consider the following grammar G:

```
E -> TX
X -> +E | ε
T -> FY
Y -> T | ε
F -> PZ
Z -> *Z | ε
P -> (E) | a | b
```

- Please list all non-terminals in the grammar and compute the FIRST and FOLLOW sets for each non-terminal. [20 points]
- Please construct the predictive parsing table for G. Is the grammar LL(1)? [20 points]
- Give the parsing steps for the input string $(a * +b) + b$. [5 points]