Question 1

Consider ternary (base-3) numbers over the alphabet $\Sigma = \{0, 1, 2\}$. Give a regular expression for each of the following languages:

1. All 3-digit numbers with no leading zeros.

2. All numbers in which the digits are non-decreasing. For example, "002" is valid, but "102" is not.

3. All numbers with no repeated digits. For example, 012 and 20 are valid, but 010 and 122 are invalid.

4. All numbers with at least one repeated digit. For example, 010 and 1122 are valid, but 0 and 20 are invalid.
**Question 2**

The following DFAs define three languages over $\Sigma = \{0, 1\}$. Describe each of the languages and give a regular expression for it.

Figure 1: (DFA-1)

Figure 2: (DFA-2)

Figure 3: (DFA-3)
Question 3

Give a context-free grammar for each of the following languages over $\Sigma = \{a, b\}$.

1. Strings that contain more a’s than b’s. For example, $a$, $aa$, and $baa$ are valid, but $b$, $ab$, and $aabb$ are invalid.

2. Strings that are palindromes. For example, $a$, $b$, and $abba$ are valid.

3. $\{a^n b^{2n} : n \geq 0\}$.

4. Strings that contain twice as many a’s as b’s. For example, $aba$ is valid.
Question 4

Consider expressions defined by the following CFG:

\[ E \rightarrow (E + E) \mid (E * E) \mid 0 \mid 1 \mid 2 \mid 3 \mid 4 \mid 5 \mid 6 \mid 7 \mid 8 \mid 9 \]

1. Is the grammar ambiguous? Why?

2. Give a left-most derivation for the expression \((2 * (3 + 4))\).

3. Draw a parse tree for the expression \((2 * (3 + 4))\).

4. Implement an evaluator in a programming language of your choice for the language based on the recursive-descent parsing framework. The top function takes an expression as a string and returns the evaluated value. For example, for "\((2 * (3 + 4))\)" , it returns 14. If the string is not a valid expression, the function throws an exception.