1) (18 points) We wish to compute the remainder of an integer (represented in binary) when divided by 3. Create a Moore machine (deterministic finite state machine) that parses a binary number left to right and outputs the remainder (mod 3) after reading each bit. For example, 1010 (the number 10) would result in the following output: 01221, where the last ‘1’ indicates 10 divided by 3 has a remainder of 1.

(Approximately 2 points off for each missing path in solution)
2) (18 points) Construct a deterministic finite-state automaton that accepts the same language as the following nondeterministic finite-state automaton. Note: your deterministic FSA need not have a transition defined for every (state, input) pair.

(Approximately 4 points off for each misplaced path in solution. I tried not to penalize anyone twice for the same mistake. 2 points off for unnecessary edges. Failure to mark final states also resulted in a couple points off)
3. (6 points) Minimize the following finite-state automaton by reducing the number of states to the minimum required. The minimized automaton must recognize the exact same language as the original automaton.

(1 point off for failure to mark final state. Partial credit for incremental solutions)
4. (18 points) Construct a regular grammar \( G = (V, T, S, P) \) that generates the language recognized by the following finite state machine:

\[
\begin{align*}
V &= \{0, 1, S_0, S_1, S_2, S_3\} \\
T &= \{0, 1\} \\
S &= \{S_0\} \\
P &= \{ \\
S_0 \rightarrow 1S_1, & \quad S_0 \rightarrow OS_3, & \quad S_0 \rightarrow 1 \\
S_1 \rightarrow 1, & \quad S_1 \rightarrow 1S_1, & \quad S_1 \rightarrow OS_3 \\
S_3 \rightarrow OS_3, & \quad S_3 \rightarrow 1, & \quad S_3 \rightarrow 1S_2 \\
S_2 \rightarrow 0_1, & \quad S_2 \rightarrow 1, & \quad S_2 \rightarrow OS_2, & \quad S_2 \rightarrow 1S_2 \\
\} 
\end{align*}
\]
5. (6 points) Construct a deterministic finite-state automaton $M_1 = (S, I, f, s_0, F)$ that recognizes the regular set $L_1 = a^*(ca)^*$

(2 points off for failure to mark final state. 3 or 4 points off for partial solutions)

6. (6 points) Construct a deterministic finite-state automaton $M_2 = (S', I', f', s'_0, F')$ that recognizes the regular set $L_2 = bb^*a$

(2 points off for failure to mark final state. 3 or 4 points off for partial solutions)

7. (10 points) Construct a non-deterministic or non-deterministic finite-state automaton $M_3 = (S'', I'', f'', s''_0, F'')$ that recognizes $L_1L_2$ (the concatenation of $L_1$ and $L_2$ from problems (5) and (6)) by combining your answers from (5) and (6).

(2 points off for failure to mark final state. 2 points off for missing links between two FSA. 8 points off for completely broken FSAs that don’t reuse the solutions from problems (5) and (6))
8. (18 points) Construct a Turing machine to double a positive unary number. Initially, the Turing machine head will be at the beginning of the input. At halt, the Turing machine head must be at the beginning of your doubled unary number (with B's at both ends). For example, the unary number 2 ...B11B... would be transformed into the unary number 4 ...B1111B... The final position of the tape head should be at the left-most '1' of the unary number.

The following are hints that you may or may not wish to use in your solution:

For each 1 encountered on original unary string, add two 1's to the end of the string (and delete the original '1')

Use temporary symbols as placekeepers.

Label your tuples with English descriptions of what role they play in your algorithm.

(4 points off for significant logical errors – how to move to the left or right of a row of ones. 2 points off for mistakes with Turing machine rules)

\[
\begin{align*}
\text{Ex 1} & : (s_0, B, s_{10}, B, R) \\
&s_0, 1, s_{11}, X, L \\
&s_1, B, s_2, 1, L \\
&s_2, B, s_3, 1, R \\
&s_3, 1, s_1, 1, L \\
&s_1, X, s_{11}, X, L \\
&s_3, 1, s_3, 1, R \\
&s_3, X, s_3, X, R \\
&s_3, B, s_4, B, L \\
&s_4, X, s_9, B, L \\
&s_4, 1, s_{11}, B, L \\
&s_{19}, 1, s_{19}, 1, L \\
&s_{19}, B, s_{20}, B, R \\
\end{align*}
\]

\[
\begin{align*}
\text{Ex 2} & : (s_0, 1, s_1, m, L) \\
&s_1, 1, s_{11}, 1, L \\
&s_1, B, s_2, 1, R \\
&s_2, 1, s_2, 1, R \\
&s_2, m, s_0, 1, R \\
&s_0, B, s_3, B, L \\
&s_3, 1, s_3, 1, L \\
&s_3, B, s_4, B, R \\
\end{align*}
\]