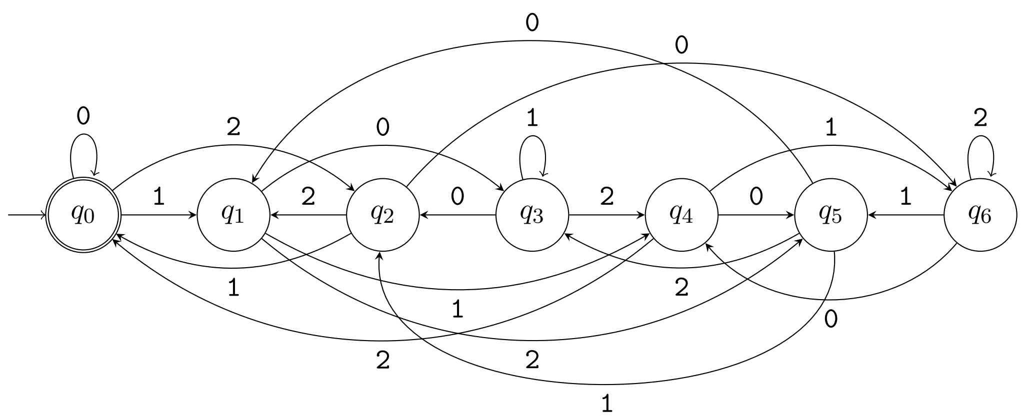
# PROBLEM SET 1 SOLUTIONS

MS. ASMA SANAM LARIK

**Note:**
**All questions will be graded out of 5.**
**Only 1 mark has been deducted for making a late submission.**
**No marks deducted for submissions not made in Word or LaTeX.**

1. Let $L = \{ w \in \{0, 1, 2\}^* \mid w \text{ represents an integer in ternary that is divisible by } 7 \}$. Draw a **DFA** for $L$. Also include the transition table / function.

   **Solution:**



| State | 0 | 1 | 2 |
|---|---|---|---|
| $q_0$ | $q_0$ | $q_1$ | $q_2$ |
| $q_1$ | $q_3$ | $q_4$ | $q_5$ |
| $q_2$ | $q_6$ | $q_0$ | $q_1$ |
| $q_3$ | $q_2$ | $q_3$ | $q_4$ |
| $q_4$ | $q_5$ | $q_6$ | $q_0$ |
| $q_5$ | $q_1$ | $q_2$ | $q_3$ |
| $q_6$ | $q_4$ | $q_5$ | $q_6$ |

   **Grading Scheme:**
   +1 mark - All strings in language accepted.
   +1 mark - All strings NOT in language rejected.
   +2 marks - Accurate transition table/function given.
   +1 mark - Accurate accept states.

2. Let $\Sigma = \{ \texttt{\#}, \texttt{!} \}$. Let $L = \{ \texttt{\#}^k u \texttt{\#}^k \mid u \in \Sigma^* \text{ and } k \geq 1 \}$. Show that $L$ is regular.

   **Solution:**
   u can be written as $\Sigma^*$ and given the condition that $k \geq 1$, we can easily write $\texttt{\#}^k$ as $\texttt{\#}^+$ on either side of the expression. This yields us the expression $\texttt{\#}^+\Sigma^*\texttt{\#}^+$. If the number of # symbols on either side is uneven, the remaining # symbols can be interpreted as part

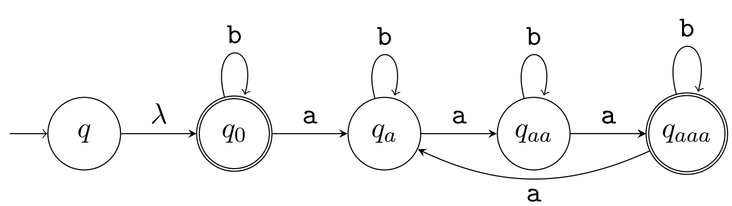
of u i.e. $\Sigma^*$. Since we can easily characterize this language L in the form of this regular expression, L is regular. Please note that **pumping lemma is NEVER used to prove a language is regular.** It is only used to prove a language is not regular. **There may be a non-regular language on which the pumping lemma holds. But if the lemma does not hold, the language is definitely non-regular.**

**Grading Scheme:**
+1 mark - Attempt at proving regularity using any DFA/NFA or regular expression etc.
+1 mark - Accepts strings starting and ending with `#`.
+1 mark - All strings in language accepted.
+1 mark - All strings NOT in language rejected.
+1 mark - Equivalence of given language and constructed DFA/NFA/regular expression implied.
(Only +2.5 marks awarded for using pumping lemma to prove regularity)

3. Construct an **NFA** that is **<u>NOT a DFA</u>** for the following language over $\Sigma = \{\, \texttt{a}, \texttt{b} \,\}$:

$$L = \{\, w \mid w \in \Sigma^* \text{ and } |a| \text{ in } w \text{ is a multiple of } 3 \,\}$$

**Solution:**
The given automaton should count the number of $a$'s in threes and accept, including the empty string and strings with no $a$'s. In addition, it should have at least one characteristic related to NFAs such as multiple transitions for same symbols and/or epsilon transitions etc. For example:




**Grading Scheme:**
+1 mark - Attempt at making an automaton.
+1 mark - All strings in language accepted.
+1 mark - All strings NOT in language rejected.
+2 marks - Automaton is clearly an NFA and not a DFA.

4. Convert your NFA from the previous question into a **DFA**. Make sure you show **each** step clearly.

**Solution:**
Straightforward solution with well-defined complete transition table from each visited state to next state for both characters $a$ and $b$. **DFA need not be a minimized DFA**. For example, for the NFA above:

| State | a | b |
|---|---|---|
| $q$ | $q_a$ | $q_0$ |
| $q_0$ | $q_a$ | $q_0$ |
| $q_a$ | $q_{aa}$ | $q_a$ |
| $q_{aa}$ | $q_{aaa}$ | $q_{aa}$ |
| $q_{aaa}$ | $q_a$ | $q_{aaa}$ |

**Grading Scheme:**
+5 marks - Complete DFA transition table made from correct NFA from Q3.
+2.5 marks - Complete DFA transition table given for incorrect automaton from Q3 *(error carried forward grace marks)*.

5. Let $\Sigma^* = \{\, \mathrm{a}, \mathrm{b} \,\}^*$. Find a regular expression for the following languages:

   a) $A = \{\, ab^k w \mid k \geq 3 \text{ and } w \in \{\, a, b \,\}^+ \,\}$.
   b) $B = \{\, vwv \mid v, w \in \{\, a, b \,\}^* \text{ and } |v| = 2 \,\}$.
   c) $C = \{\, vwv \mid v, w \in \{\, a, b \,\}^* \text{ and } |v| \ \leq 3 \,\}$.

   **Solution:**
   **a)** Observe all strings in A start with one $a$, are followed by at least three $b$'s and then by at least one other character.
   **b)** Observe all strings in B start and end with any one of these strings $\{aa, ab, ba, bb\}$, and can have any other characters including $\lambda$ in between.
   **c)** Observe all strings in C start and end with any one of these strings $\{\lambda$, *a, b, aa, ab, ba, bb, aaa, aab, aba, abb, baa, bab, bba, bbb*$\}$, and can have any other characters including $\lambda$ in between.

   **Grading Scheme:**
   +0.5 marks - Attempt at making any regular expression for **all** parts.
   +1.5 marks - $a.bbb.\{b\}^*.\Sigma^*$ or equivalent expression for **(a)**
   +1.5 marks - $\Sigma^2.\Sigma^*.\Sigma^2$ or equivalent expression for **(b)**
   +1.5 marks - $\{\lambda \cup \Sigma \cup \Sigma^2 \cup \Sigma^3\}.\Sigma^*.\{\lambda \cup \Sigma \cup \Sigma^2 \cup \Sigma^3\}$ or equivalent expression for **(c)**

6. The language *DIFFERENCE(A and B)* contains all strings in language A that are not in language B. Formulate this language as a combination of closure properties you have discussed in class ($A \cup B$, $A \cap B$, $A^*$, $\overline{A}$ etc) to show the class of regular languages is closed under *DIFFERENCE*.

   **Solution:**
   The language simply accepts all strings that are accepted by any automaton for A and are at once rejected by that for B. An automaton can be defined for *DIFFERENCE(A and B)* as 5-tuple definition similar to the 5-tuple definition of the union of A and B given in Sipser (Page 60) with the addition that final states for the automaton for *DIFFERENCE* will be the accept states of the automaton for A, and reject states for the automaton for B. For example, if N1 = $(Q_A, \Sigma, \delta_B, q_A, F_A)$ is the **DFA** for A, and N2 = $(Q_B, \Sigma, \delta_A, q_B, F_B)$ is the **DFA** for B, then N3 is the **NFA** that can be constructed for *DIFFERENCE(A and B)* such that N3 = $(Q, \Sigma, \delta, q_0, F)$ where all details are similar to the example given in the book, except that: $F = F_A \cap \{Q_B - F_B\}$

**Grading Scheme:**
+1 mark - $A \cap B'$ or equivalent expression given.
+1 mark - 5-tuple automaton definition, diagram or any indication of proof of regularity of language.
+1 mark - Accurate set of states.
+1 mark - Accurate transition function.
+1 mark - Accurate set of final states.
**Note: No marks deducted for proof by example or missing initial state transitions in transition function.**