

There are a total of four problems. You have to solve them all.

**Problem 1: Finite Automata and the Regular Operations (10 points)**

Let  $\Sigma = \{0, 1, \#\}$ . Consider the following two languages.

$L_1 = \{w \in \Sigma^* : w \text{ does not contain } \# \text{ and the number of 0s in } w \text{ is not a multiple of } 3\}$

$L_2 = \{w \in \Sigma^* : \text{the substring between any two successive occurrences of } \# \text{ in } w \text{ is in } L_1\}$

Now solve the following problems.

- (a) Write down a string  $w \in L_2$  such that the length of  $w$  is ten. (1 point)
- (b) Give the state diagram for a DFA that recognizes  $L_1$ . (4 points)
- (c) Give the state diagram for a DFA that recognizes  $L_2$ . (3 points)
- (d) If you use the “cross product” construction shown in class to obtain a DFA for  $L_1 \cap L_2$ , how many states will it have? (1 point)
- (e) Give an upper bound on the number of states in the smallest DFA that recognizes  $L_1 \cap L_2$ . (1 point)

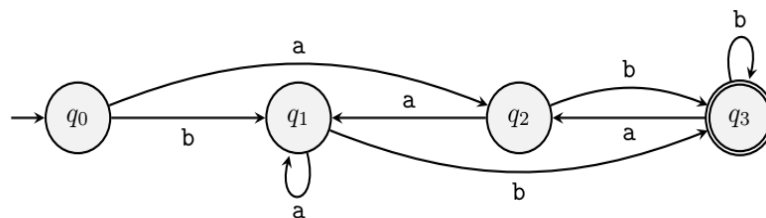
**Problem 2: Regular Expressions (10 points)**

Mike and Willy recently learned how to write regular expressions. Mike wrote the regular expression  $10^*1^*$  for a language  $L_1$  on the board and Willy wrote the regular expression  $1^*01^*$  for another language  $L_2$  below that.

- (a) Write down a string that is present in the language  $L_1$  but not in the language  $L_2$ . (2 points)
- (b) Write down a string that is not present in the language  $L_1$  but present in the language  $L_2$ . (2 points)
- (c) Write down a string that is neither present in the language  $L_1$  nor in the language  $L_2$ . (2 points)
- (d) Mike and Willy asked their friend Dustin to write a regular expression for the language  $L_1 \cap L_2$ . Dustin came up with  $1^*0^*1^*$ . Is Dustin’s regular expression correct? If you think it’s not correct, then write down a correct regular expression for  $L_1 \cap L_2$ . (4 points)

**Problem 3: Converting DFAs to Regular Expressions (10 points)**

Convert the following DFA into an equivalent regular expression using the state elimination method. First eliminate  $q_1$ , then  $q_2$  and so on. You must show work.

**Problem 4: Converting Regular Expressions to NFAs (10 points)**

Convert the following regular expression into an equivalent NFA.

$$(a^*b^* + (ac + b^*c)b)^*$$