2FA3 - Assignment 1

Submitted via Avenue. Due Feb 16th, 11:59pm. Note: I indicate what each question if worth by square brackets, i.e. [k].

1. For each statement below, state if it is true of false, and explain why. The explanation does not need to be a formal proof, but the argument should be sound.[6]

   (a) If L_1 is regular and $|L_1| = k$ and L_2 is non-regular, then $L_1 \cap L_2$ is regular.

   (b) If L_1 is regular and L_2 is non-regular, then $L_1 \cup L_2$ is regular.

   (c) $\forall L_1$ such that L_1 is a non-regular language, $\exists L_2$ such that L_2 is regular and $L_1 \subseteq L_2$.

a) True. $L_1 \cap L_2 = L_3$ will give $L_3 \subseteq L_1$ because of the intersection. An example of when a subset of a regular language is non-regular is $(a^* b^*)$, the subset being L_2 $(a^n b^n \mid n \geq 0)$. However, $|L_1| = k$, meaning there is a fixed number of strings that are in L_1, while $(a^* b^*)$ has an infinite set of accepted strings.

Since L_1 must accept a fixed number of strings, the graph of the DFA M where $\mathcal{L}(M) = L_1$ must not contain a cycle. If $L_3 \subseteq L_1$, the DFA M_3 such that $\mathcal{L}(M_3) = L_3$ must be possible to create as it will contain logic from M, but any removals from the language can be dealt with by deleting paths or creating new branching paths.

b) False. If $L_1 = \{ab\}$, that is, it only accepts the string ab, and $L_2 = \{a^n b^n \mid n \geq 0\}$, $L_1 \cup L_2 = \{a^n b^n \mid n \geq 0\}$, which is known to be non-regular.

c) True. For all non-regular languages L_1, they can be seen as a subset of Σ^*. However, Σ^* is a regular language, and it can be modeled by a DFA which has one state, an accepting state, and loops to itself on all inputs. Thus, Σ^* is a regular language that is a superset for all non-regular languages.

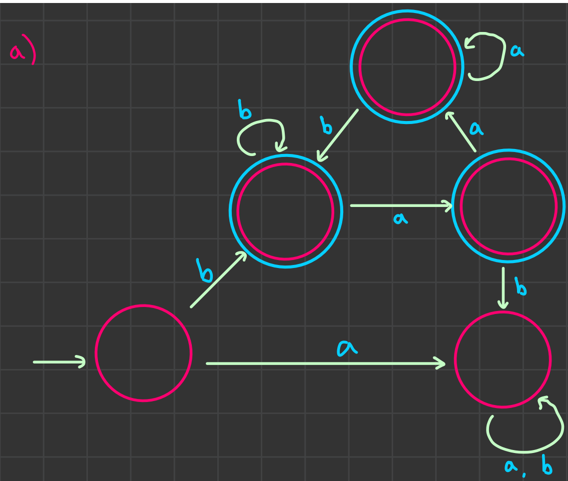
2. Create a DFA M, such that:

(a) M accepts all strings which begin with b but do not contain the substring bab. [2]

(b) $\mathcal{L}(M) = \{a^i b^j c^k \mid i + j + k \text{ is a multiple of } 3\}$, $\Sigma = \{a, b, c\}$ [3]

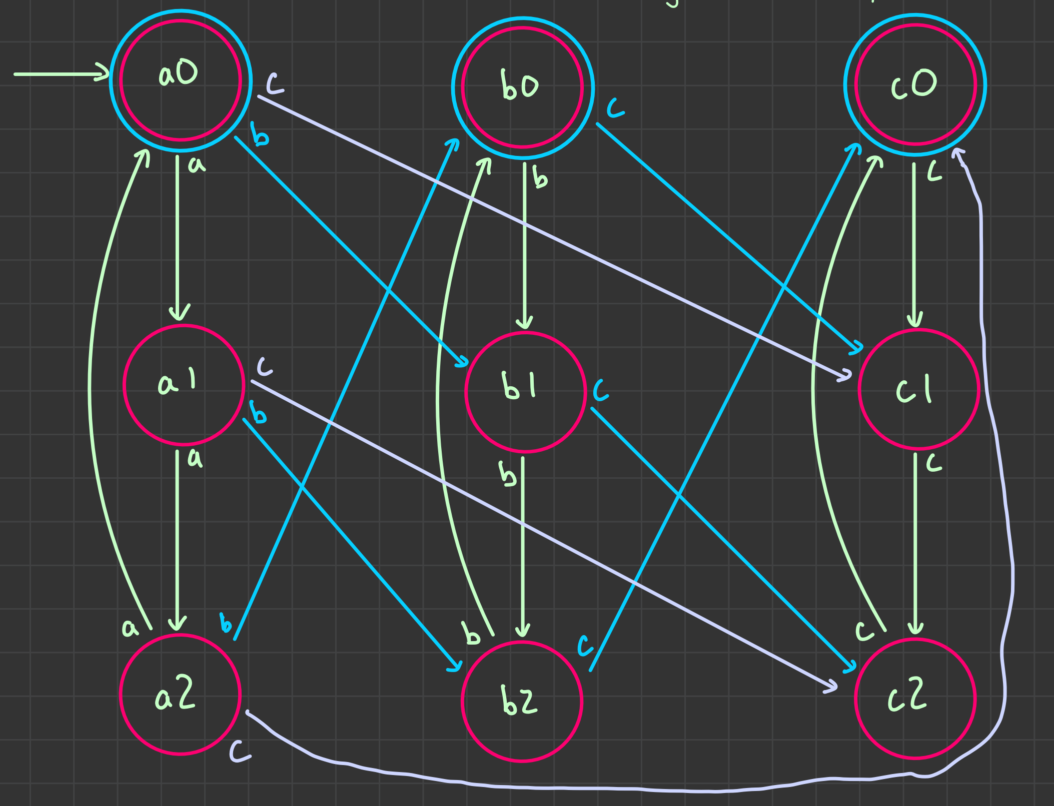
(c) $\mathcal{L}(M) = \{x \mid \text{There are at least two a's in the last three characters of } x\}$

a)



Note: transitions not shown go to the drain state.
eg. at state c0, read in a or b

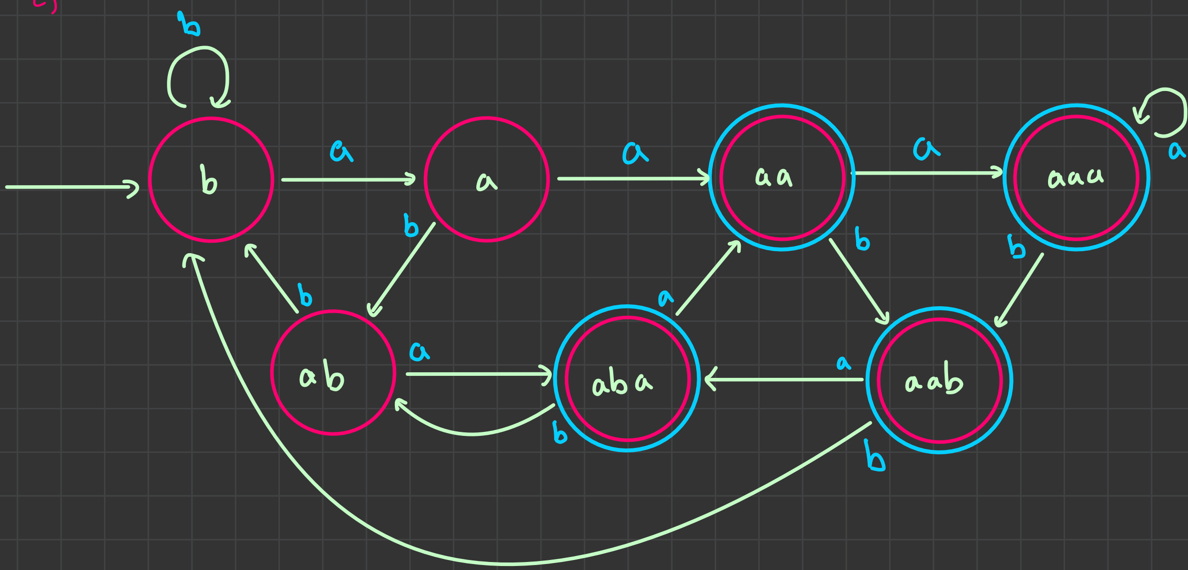
b)



2. Create a DFA M, such that:

(c) $\mathcal{L}(M) = \{x \mid \text{There are at least two a's in the last three characters of } x\}$

c)



3. Via product construction, create a DFA M, such that

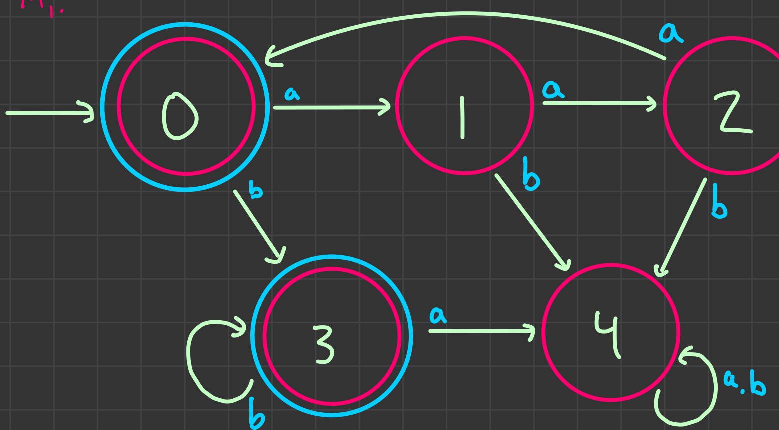
$$\mathcal{L}(M) = \{a^n b^m \mid n \text{ or } m \text{ is a multiple of } 3\}$$

First create two machine: one where n is a multiple and one where m is a multiple of three. Then create the "union" machine. When I say create two machines, I mean an M_1 and M_2 such

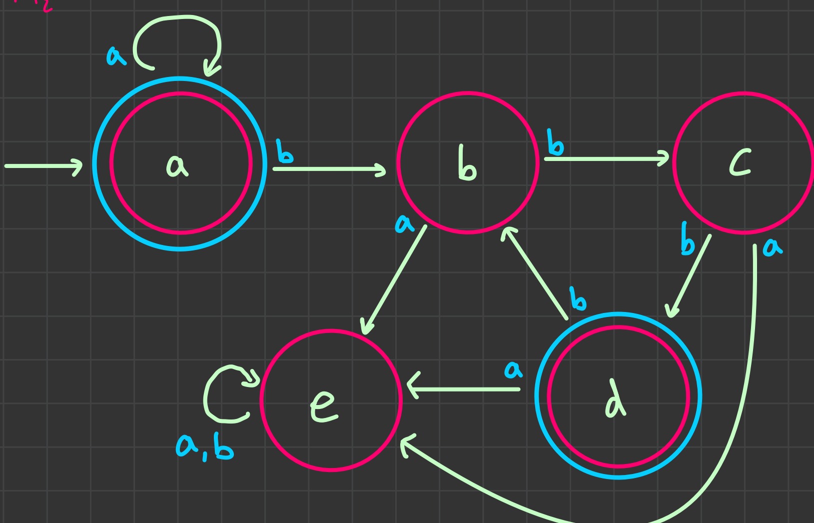
$$\mathcal{L}(M_1) = \{a^n b^m \mid n \text{ is a multiple of } 3\}$$
$$\mathcal{L}(M_2) = \{a^n b^m \mid m \text{ is a multiple of } 3\}$$

[5]

M_1:



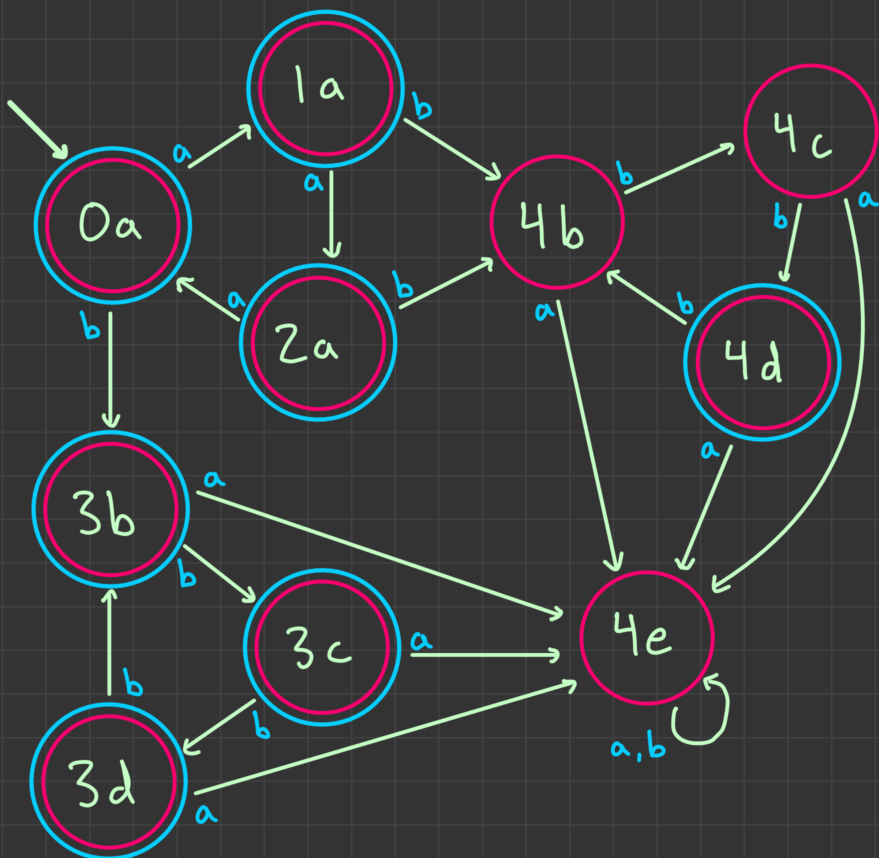
M_2:



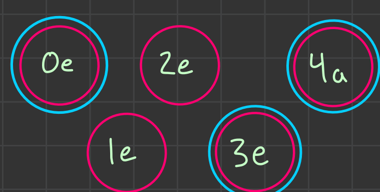
	next state given input	
state	a	b
0a	1a	3b
0b	1e	3c
0c	1e	3d
0d	1e	3b
0e	1e	3e
1a	2a	4b
1b	2e	4c
1c	2e	4d
1d	2e	4b
1e	2e	4e
2a	0a	4b
2b	0e	4c
2c	0e	4d
2d	0e	4b
2e	0e	4e
3a	4a	3b
3b	4e	3c
3c	4e	3d
3d	4e	3b
3e	4e	3e
4a	4a	4b
4b	4e	4c
4c	4e	4d
4d	4e	4b
4e	4e	4e

states which are never reached are crossed out

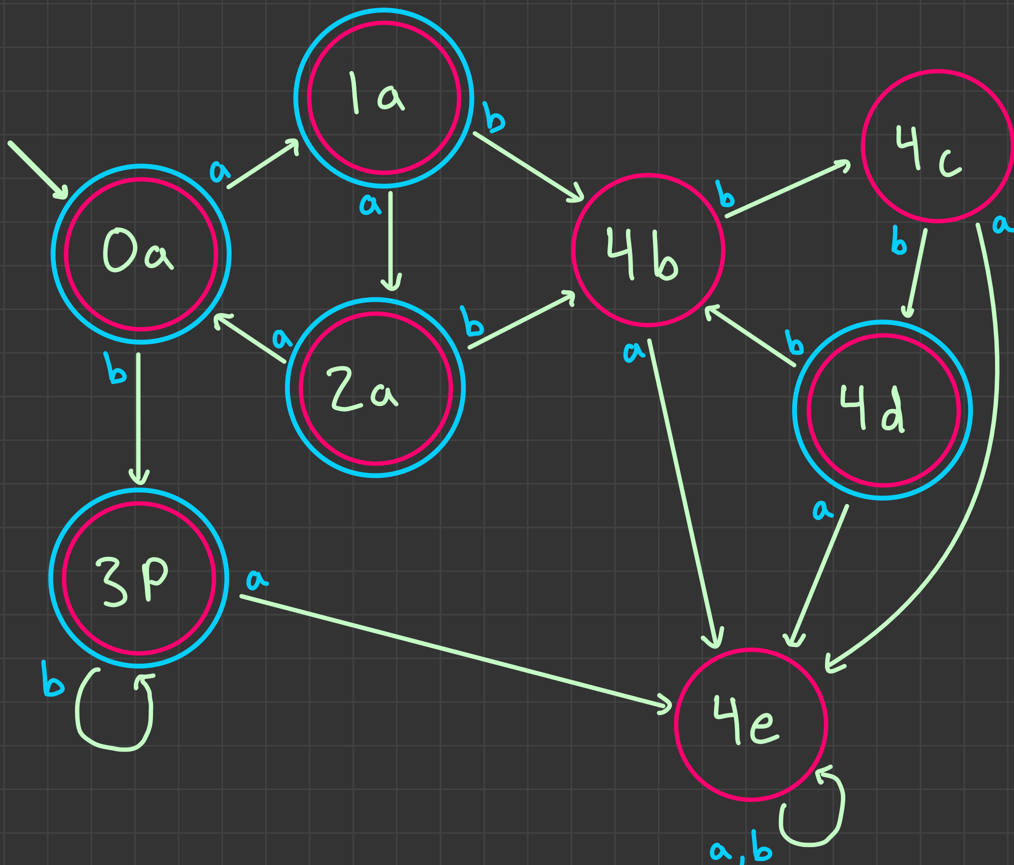
state	next state given input: a	b
0a	1a	3b
~~0b~~	~~1e~~	~~3c~~
~~0c~~	~~1e~~	~~3d~~
~~0d~~	~~1e~~	~~3b~~
0e	1e	3e
1a	2a	4b
~~1b~~	~~2e~~	~~4c~~
~~1c~~	~~2e~~	~~4d~~
~~1d~~	~~2e~~	~~4b~~
1e	2e	4e
2a	0a	4b
~~2b~~	~~0e~~	~~4c~~
~~2c~~	~~0e~~	~~4d~~
~~2d~~	~~0e~~	~~4b~~
2e	0e	4e
~~3a~~	~~4a~~	~~3b~~
3b	4e	3c
3c	4e	3d
3d	4e	3b
3e	4e	3e
4a	4a	4b
4b	4e	4c
4c	4e	4d
4d	4e	4b
4e	4e	4e



Not connected to the rest of the DFA:

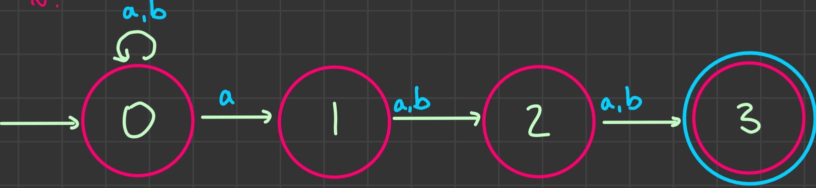


For anyone who wants to fully minimize M



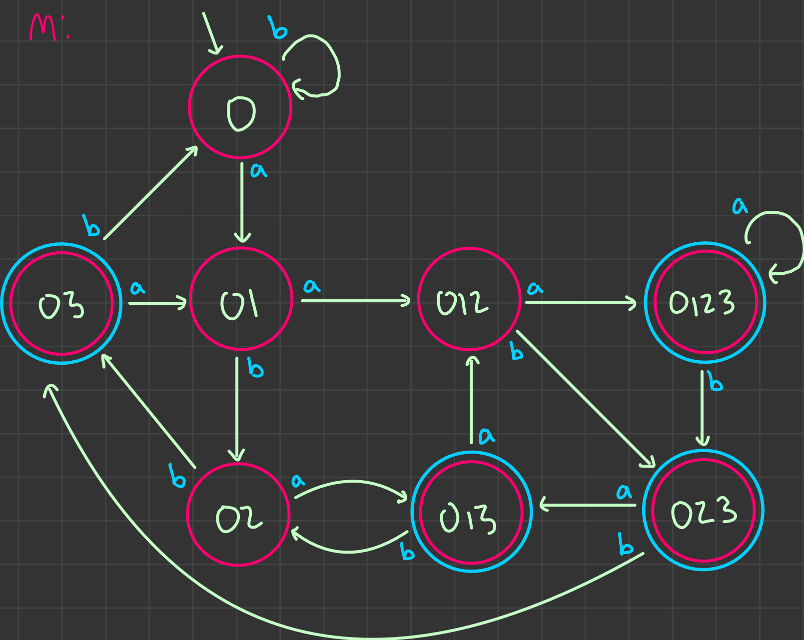
4. Create an NFA which accepts all strings in which the third last character is an a. Then via subset construction, create an equivalent DFA. Show all your work. [5]

N:



	next state given input	
state	a	b
0	01	0
1	2	2
2	3	3
3	∅	∅
01	012	02
02	013	03
03	01	0
12	23	23
13	2	2
23	3	3
012	0123	023
013	012	02
023	013	03
123	23	23
0123	0123	023

M:



Not connected to the rest of the DFA: