1 T/F Questions (4 pts)

Directions: For each of the following, select True or False.

1. All NFA's can be converted to an equivalent DFA.

   Solution. True □

2. All regular expressions can be converted to an equivalent NFA.

   Solution. True □

3. DFA's can have epsilon transitions.

   Solution. False □

4. DFA's can have multiple final states.

   Solution. True □

5. Using an epsilon transition while evaluating a string on an NFA consumes a character.

   Solution. False □

6. For certain NFA's, an epsilon-closure on one of the states in the NFA may return the empty list.

   Solution. False □

7. You can consider DFA's as behaving like they are in multiple states at once.

   Solution. False □

8. You can consider NFA's as behaving like they are in multiple states at once.

   Solution. True □

2 Multiple Choice Questions (3 pts)

Directions: Select all of the strings that will be accepted by the provided NFA (there may or may not be more than one correct answer).

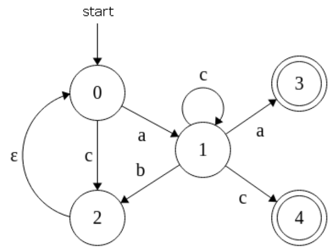


Figure 1: NFA 1

1. Which strings will be accepted by the provided NFA:

   (a) aba

   (b) abcca

   (c) abccac

   Solution. (c) abccac □

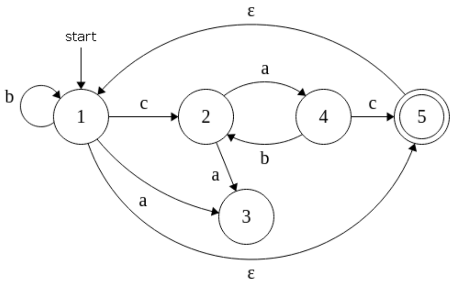


Figure 2: NFA 2

2. Which strings will be accepted by the provided NFA:

   (a) aba

   (b) caac

   (c) caccac

Solution. (c) caccac □

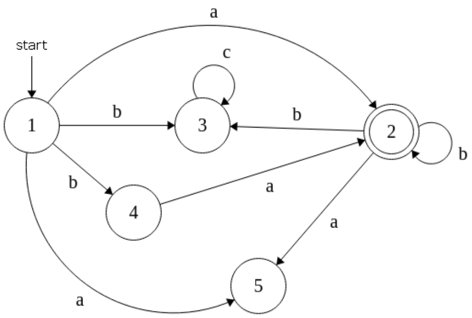


Figure 3: NFA 3

3. Which strings will be accepted by the provided NFA:

   (a) aba

   (b) abbb

   (c) bab

Solution. (b) abbb, (c) bab □

3 E-closure Questions (2 pts):

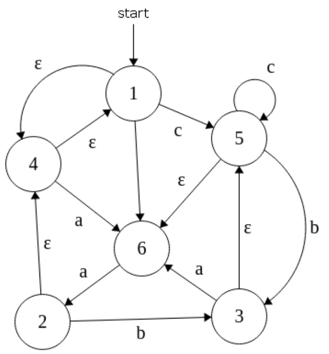


Figure 4: NFA 4

1. Write all the states that are in the ϵclosure of state 1 in the provided NFA.

   Solution. {1,4} Due to the typo in the above FA, {1,4, 6} also received full credit. □

2. Write all the states that are in the ϵ closure of state 2 in the provided NFA.

   Solution. {1,2,4} Due to the typo in the above FA, {1,2,4,.6} also received full credit. □

3. Write all the states that are in the ϵ closure of state 3 in the provided NFA.

   Solution. {3,5,6} □

4 Basic NFA→Regex (3 pts)

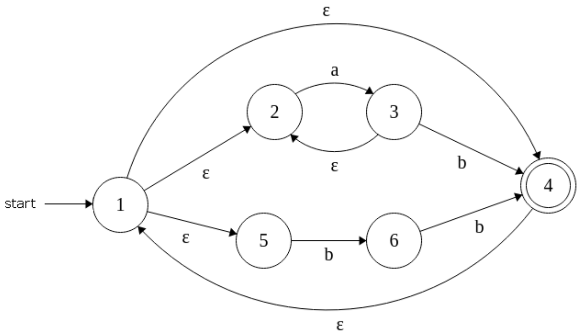


Figure 5: nfa05.png

1. Write down the regular expression equivalent to the provided NFA.

   Solution. ((a+b)|bb)* □

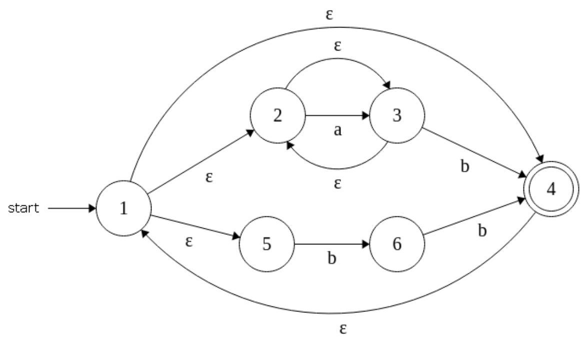


Figure 6: nfa06.png

2. Write down the regular expression equivalent to the provided NFA.

Solution. ((a*b)|bb)* □

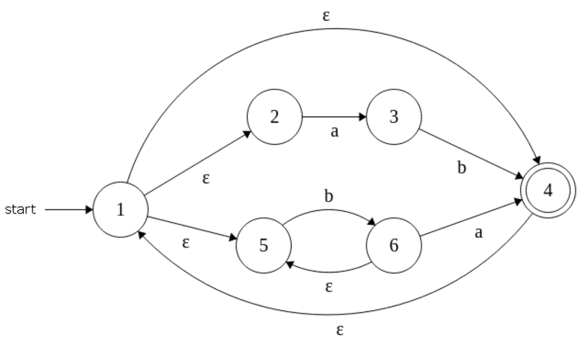


Figure 7: nfa07.png

3. Write down the regular expression equivalent to the provided NFA.

Solution. ((ab)|b+a)* □

5 NFA→DFA (8 pts)

Directions: Convert the provided NFA to an equivalent DFA using the algorithm taught in class. To represent your solution DFA, use the same syntax that was used in the project (provided below). We will be grading these by hand.

```
type ('q, 's) transition = 'q * 's option * 'q
type ('q, 's) nfa_t = {
sigma : 's list; (* alphabet *)
qs : 'q list; (* list of states *)
q0 : 'q; (* initial state *)
fs : 'q list; (* final states *)
delta : ('q, 's) transition list; (* transitions *)
}
```

1. NFA Image:

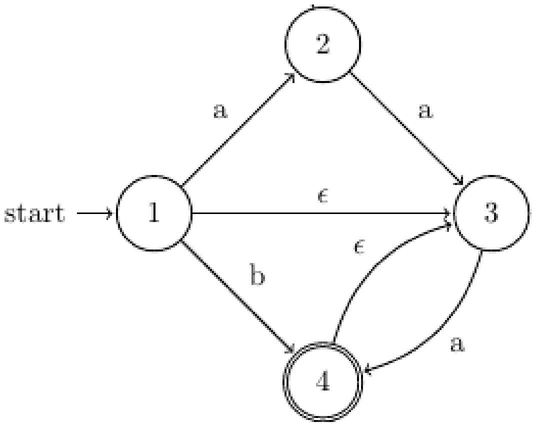
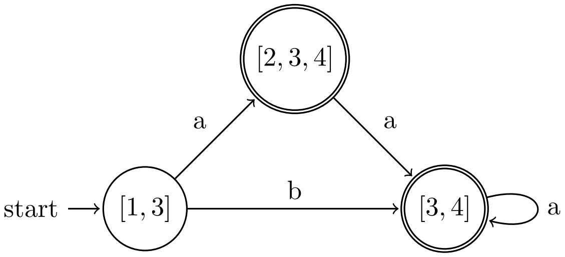


Figure 8: nfa08.png

Solution. DFA Image:



□

2. NFA Image:

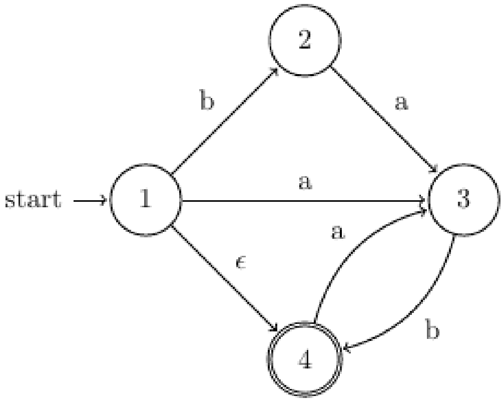
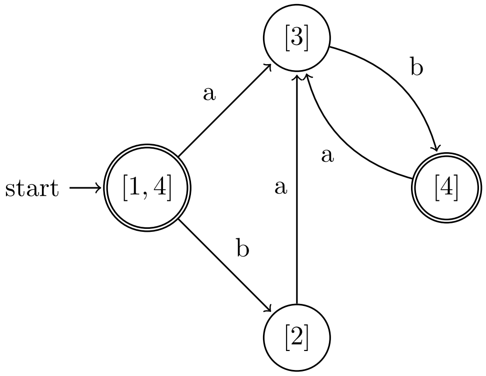


Figure 9: nfa09.png

***Solution*.** DFA Image:



□

3. NFA Image:

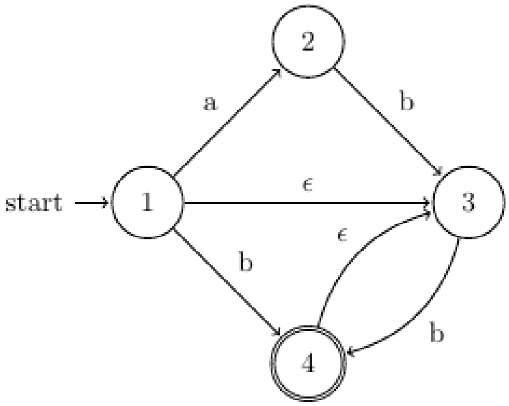
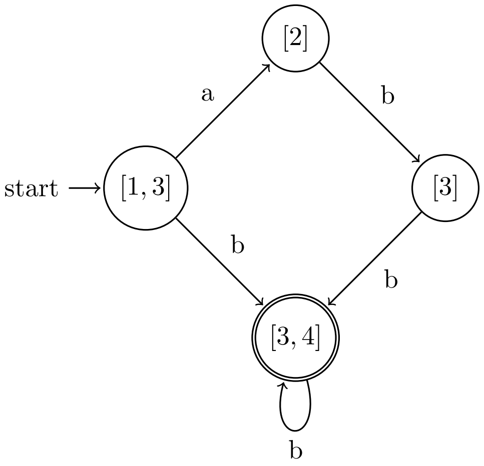


Figure 10: nfa10.png

Solution. DFA Image:



□