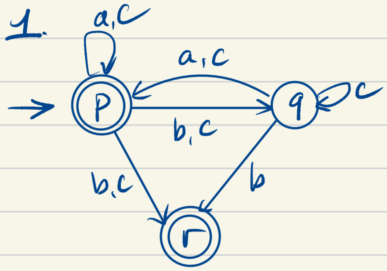
Vu Nguyen

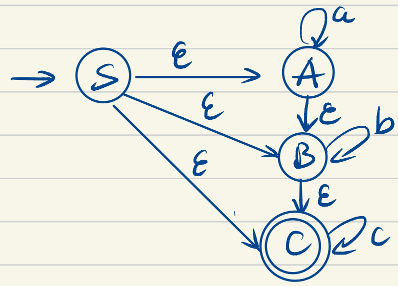
1.



Tabular form

	a	b	c
p	{p}	{q,r}	{p,q,r}
q	{r}	{r}	{p,q}
r	$\emptyset$	$\emptyset$	$\emptyset$

2.



states	ε	a	b	c
→S	{A,B,C}	$\emptyset$	$\emptyset$	$\emptyset$
A	{B}	{A}	$\emptyset$	$\emptyset$
B	{C}	$\emptyset$	{B}	$\emptyset$
*C	$\emptyset$	$\emptyset$	$\emptyset$	{C}

3\. 1. $w = \varepsilon$ because
- By def., the string with length 0 is the empty string ε
- (C) Reasoning about properties of binary strings

2\. $\delta(A, \varepsilon) = A$ because
- Reason: By def., the extended transition function DFA states that $\delta(q, \varepsilon) = q$, meaning that if no input is given, the state remain the same
- (B)

3\. ε has an even # of 0's because:
- Reason: the empty string contains 0 occurrences of any symbol, which is an even #
- (C)

4\. there are 2 cases: a. when $w = x1$ & b. when $w = x0$ because:
- Reason: Any non-empty binary string ends in either a '1' or a '0'
- (C)

Case (a)

5\. In case (a), w has an odd # of 1's iff x has an even # of 1's, because
- Reason: Appending a '1' to a string changes the parity of the count of '1's. If x has an even #, now w has an odd #
- (C)

6\. In case (a), $\delta(A, x) = A$ iff w has an odd # of '1's because:
- Reason: the induction hypothesis states if x has an even # of '1's (& hence w has an odd #), we are in state A
- (A)

7\. In case (a), $\delta(A, w) = B$ iff w has an odd # of '1's because:
- Reason: Given $\delta(A, x) = A$ & $w = x1$, the transition takes us from A to B on input '1'
- (B)

In case (b)

8\. w has an odd # of '1's iff x has an odd # of '1'
- Reason: Appending a '0' to a string does not change the count of '1'. If x has an odd # of '1's, w will also have an odd number
- (C)

9. $\delta(A,x)=B$ iff w has an odd # of '1's because:
- Reason: The induction hyp. states that if x has an odd # of '1', we're in B
- (A)

10. $\delta(A,w)=B$ iff w has an odd # of '1' because:
- Reason: Given $\delta(A,x)=B$ & $w=x0$, the transition function keeps us in B on input 0
- (B)

4. (a) 1. Start with a DFA for regular language since every regular language can be represented by a DFA
 2. Transform the DFA → ε-NFA by
 - Adding new start state with a single ε-transition to the original start state
 - For each state, ensure that for each input symbol, there is only 1 transition out of that state. If a state has >1 transitions for the same input, introduce intermediate states w ε-transitions to separate these transitions
 - If there are multiple accept states, add ε-transitions from each to a new, single accept one

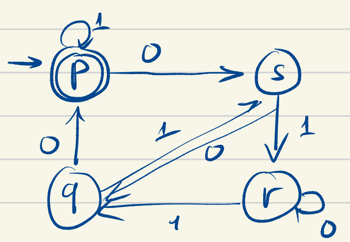
 (b) 1. from a DFA of regular lang, create a chain of states in the ε-NFA where each state has an ε-tran. to the next state
 2. At each state, include 1 non-ε-tran that corresponds to the transition in the DFA for the current input symbol
 3. When the DFA would transition to a new state on an input symbol, the ε-NFA transitions on that symbol to a new state in the chain, which then ε-transitions to the next state in the sequence, simulating DFA's computation
 4. The accept states of the DFA correspond to states in the ε-NFA chain that will ε-tran to a new accepting state

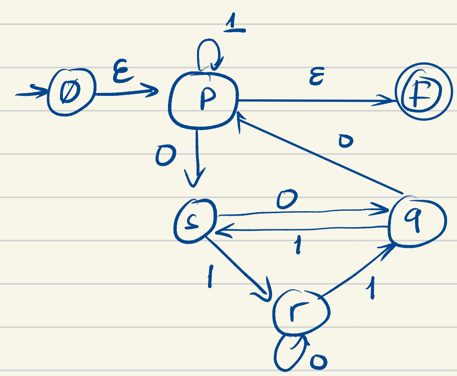
5. b. (0|1)*1(0|1){9}$
 c. ^(0*10*1)?0*$

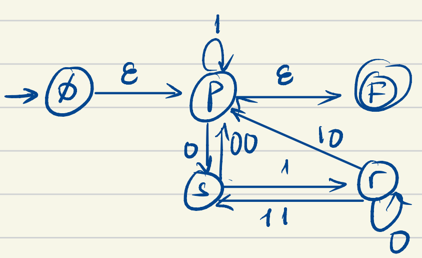
6. a. ^0*10*10*1*$
 b. ^(1*01*01*01*01*01*)*$

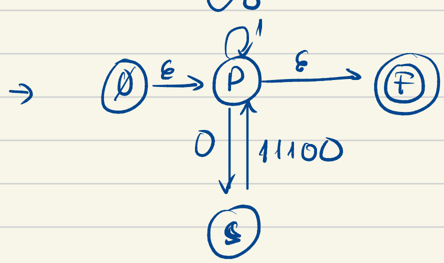
7. $a^*(b+c)^*$

8.





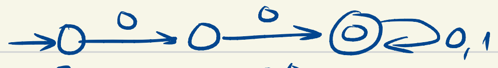




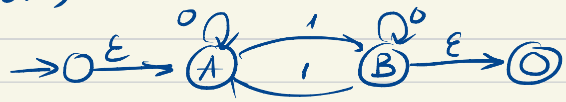
$1+0((0+10^*1)1)^*(0+10^*1)0$

answer

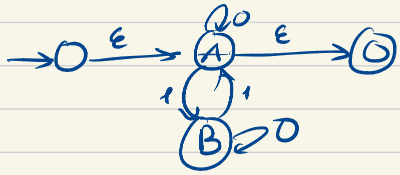
9. $10(0+1)^*$



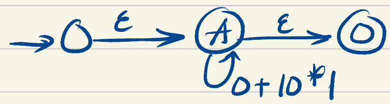
10. a.



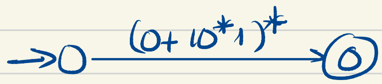
11. a



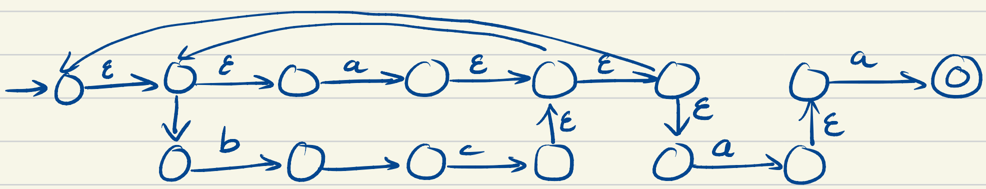
b.

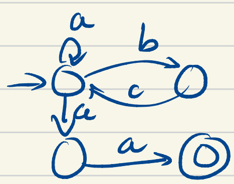


$(0+10^*1)^*$



13.





14. $a^*(b+c)^*$