1. Provide two valid strings in the languages described by each of the following regular expressions, with alphabet $\Sigma = \{0,1,2\}$.

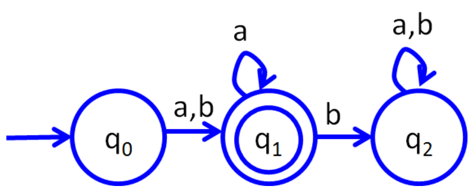
(a) $0(010)^*1$

(b) $(21 \cup 10)^*0012^*$

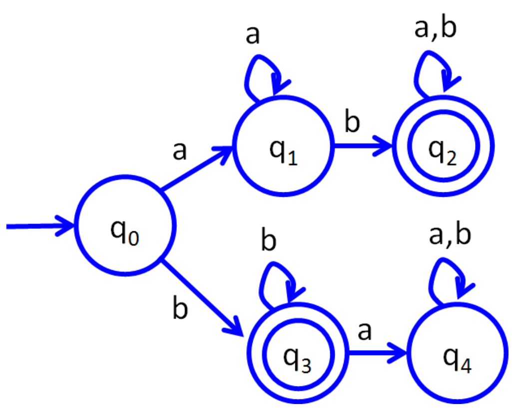
(c) $1^*(200)^* \cup 100^*01$
Examples: 1, 200, 111, 11200200, 111200200200, 1001, 1000001, 10000001

2. For each of the following DFAs, provide a Regular Expression to describe the language, with alphabet $\Sigma = \{a, b\}$.

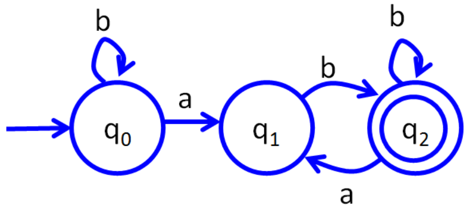
(a) RED QUESTION



(b) BLUE QUESTION



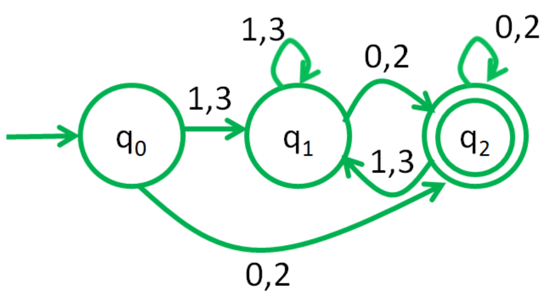
(c) GREEN QUESTION



b*abb*(abb*)*

3. Create a DFA to accept each of the following languages.

A={w | last number in w is even} , given alphabet $\Sigma = \{0,1,2,3\}$

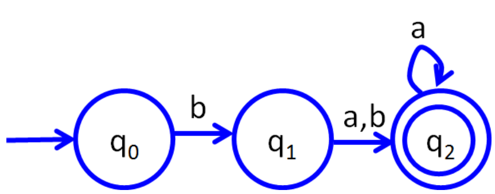


B={w | at least three symbols in w} , given alphabet $\Sigma = \{a, b, c\}$

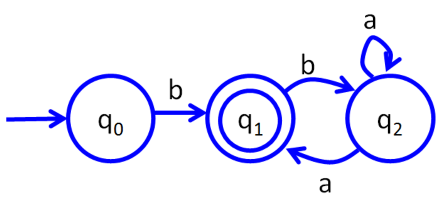
C={w | sum of digits in w equals 2} , given alphabet $\Sigma = \{0,1,2\}$

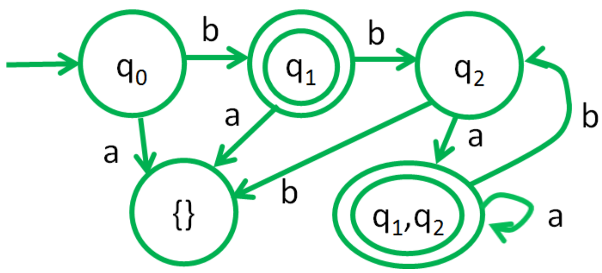
4. Convert each of the following NFAs to a DFA, with alphabet $\Sigma = \{a, b\}$.

(a) RED QUESTION

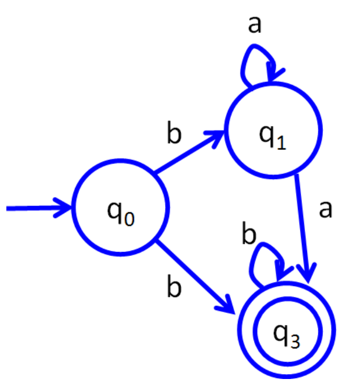


(b) GREEN QUESTION





(c) BLUE QUESTION



5. Prove the following languages are **not** regular.

(a) $A=\{b^{k!}a \mid k>0\}$

(b) $B=\{0^k1^{2k}0^k \mid k>0\}$

7. Provide two valid strings for each of the following CFGs.

(a) G1:

S -> A | B
A -> DC | C
B -> EF | F
C -> dog | cat | mouse
D -> big | small | red | white
E -> quickly | slowly
F -> runs | swims | jumps | barks

A -> DC -> **big mouse**
A -> C -> **cat**
B -> EF -> **slowly runs**
F -> **barks**

(b) G2:

S -> BA | B
B -> xBx | ε
A -> c | de | f

(c) G3:

S -> CaC | C
C -> yCy | y

8. Convert the following CFGs to CNF (same as Q7).

(a) G1: (for G1, each word is a terminal)

S -> A | B
A -> DC | C
B -> EF | F
C -> dog | cat | mouse
D -> big | small | red | white
E -> quickly | slowly

F -> runs | swims | jumps | barks

(b) G2:

S -> BA | B
B -> xBx | ε
A -> c | de | f

S -> BA | A | B | ε *Distribute ε*
B -> xBx | xx
A -> c | de | f

S -> BA | c | de | f | xBx | xx | ε *Use single variable substitutious*
B -> xBx | xx
A -> c | de | f

S -> BA | c | DE | f | XBX | XX | ε *Substitute literals with variables*
D -> d
E -> e
X -> x
B -> XBX | XX
A -> c | DE | f

S -> BA | c | DE | f | XQ | XX | ε
Q -> BX
D -> d
E -> e
X -> x
B -> XQ | XX
A -> c | DE | f

(c) G3:

S -> CaC | C
C -> yBy | y

9. Express each of the following languages as a **CFG**.

(a) $A = \{x^k y^{2k} z\}$

(b) B = {w | w is described by (ab)*ba }

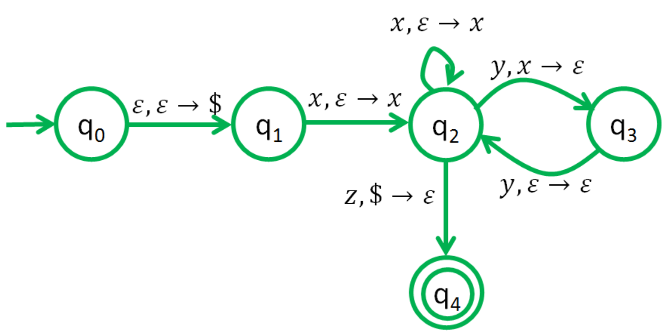
S -> Cba

C -> abC | ε

(c) $C = \{ 010^k 101^{k+2} \mid k > 0 \}$

10. Describe the PDA to accept each of the following languages (languages from Q9).

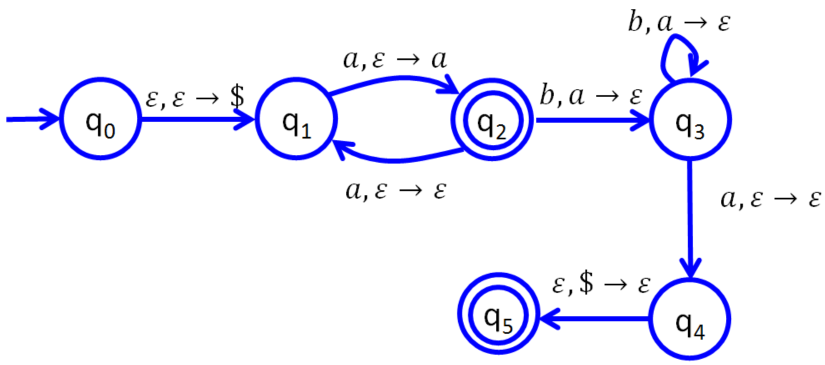
(a) $A = \{x^k y^{2k} z\}$



(b) B = {w | w is described by (ab)*ba }

(c) $C = \{ 010^k 101^{k+2} \mid k > 0 \}$

11. What is the response of PDA P1 to each input: **i.e., does it reach an accept state?**



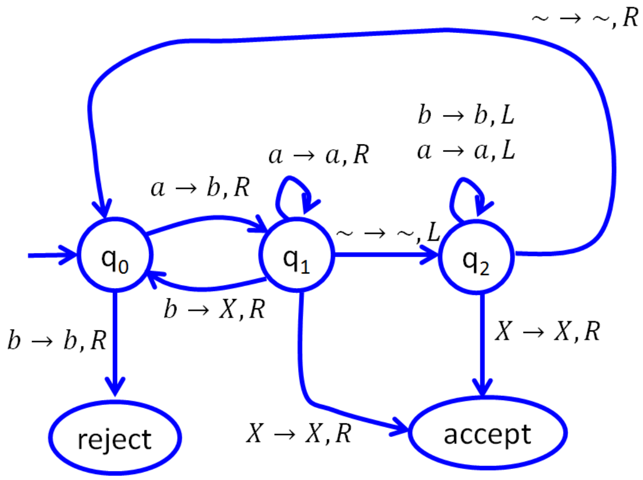
Input 1: bbaa

Input 2: aaa
Reaches accept. Gets to q_2 state.

Input 3: abb

Input 4: aaaaabbba
Reaches accept! 3 a's put on stack, popped with 3 b's.

12. Describe the configurations resulting from each of the input tapes specified below for the following Turing Machine.



(a) aabb

(b) abaaa

(c) aaaba

q_0 $\underline{a}aaba$
q_1 $b\underline{a}aba$
q_1 $ba\underline{a}ba$
q_1 $baa\underline{b}a$
q_0 $baaX\underline{a}$
q_1 $baaXb\underline{\sim}$
q_2 $baaX\underline{b}$
q_2 $baa\underline{X}b$
accept $baaX\underline{b}$

13. Express the following problems as languages.

(a) Determine if two specified CFG's accept complementary inputs – every accepted input for the first CFG is rejected by the second CFG and vice versa.

(b) Determine if a specified DFA accepts a specified string repeated zero or more times.

$L = \{<D,w> \mid L(D) = w^*\}$

(c) Determine if a specified Turing machine accepts the same language as a specified PDA.

14. Prove the follow languages are decidable.

(a) Determine if a specified DFA accepts a specified string repeated zero or more times.

By construction: We can construct a DFA to accept a specified string once by defining a DFA with a single edge corresponding to each symbol in the string. To accept arbitrary numbers of times, we add a loop back from the final state to the first state. This construction takes finite time.

We can test if any input DFA is equivalent to the constructed DFA. This problem is decidable through the language EQ_{DFA}.

(b) Determine if a specified CFG is in Chomsky Normal Form.

(c) Determine if a specified CFG does not accept a specified word.

15. Provide a big-O and a little-o complexity for each function.

(a) $f(n) = 20\, n \log n + 5n + 2$

(b) $f(n) = 30\ n^3 + 6\ n^5 + \log n$
Smallest: $O(n^5)$; also $O(n^{10})$, $O(2^n)$
Near-smallest: $o(n^5 \log n)$ or $o(n^6)$ also, $o(n^{20})$, $o(2^n)$

(c) $f(n) = 5\ n^2 + n^3 \log n + 4^n + 8$

16. Compute the complexity for each algorithm described below.

(a) Algorithm 1: (State the complexity based on *r* and *c*)
Start with a table of *r* rows and *c* columns
1. Sum the elements in each row
   - Use a running sum with a loop across all columns
2. Find the row with the maximum sum
   - Loop through all rows, saving biggest sum and its row in two separate variables

(b) Algorithm 2: (State the complexity based on *n*)
Start with a list of *n* elements
1. While list is longer than 1 element long
   - Replace each pair of elements with the product of the two elements (elements 1 and 2 replaced by single product, elements 3 and 4 replaced by single product, elements 5 and 6 replaced by single product, etc.)

17. Determine if the following problems are in P and/or NP.

(a) Given a directed graph and two nodes a and b, determine if there are at least two different paths to get from node a to node b. Paths are "different" if they differ by at least one edge.

We established how to find a single path in polynomial time using edge labeling, starting at node a and attempting to arrive at node b. After completing the process once, we can repeat this process a second time, with a loop removing from the graph one of the edges from the first solution each time and checking whether a path still exists between a and b. This solution for each altered graph will be polynomial, and the number of edges removed will be O(m) in the number of edges in the graph m (and even O(n) in the number of nodes in the graph), so the resulting algorithm is polynomial.
In the end, **this problem is in P (which means it also is in NP).**

(b) In an undirected graph, determine if every node is attached to every other node.

(c) Determine if the language of a DFA is empty.