|  |  |
| --- | --- |
|  | **Sub Title: Automata Theory and Introduction to Compilers** |
| **Sub Code: 21CST501** |
| **Assignment-3** |

1. Consider the language L over the alphabet {0, 1,2,…,9} defined as follows:

L={w∈ {0, 1,2,…,9}∗∣ the Decimal representation of the decimal value of w is a leap year}

*(Consider both the condition of millennium and divisibility rule)*

**Solution:**

**Transitions:**

* From *q*0​, transitions for digits 0-9 looping back to *q*0​.
* From *q*0​ to *q*1​ on seeing the digit 1.
* From *q*0​ to *q*2​ on seeing the digit 2.
* From *q*0​ to *q*reject​ on seeing the digits 3-9.
* From *q*1​, transitions for digits 0-9 looping back to *q*1​.
* From *q*1​ to *q*reject​ on seeing digits other than 0-9.
* From *q*2​, transitions for digits 0-9 looping back to *q*2​.
* From *q*2​ to *q*3​ on seeing any digit.
* From *q*3​, transitions for digits 0-9 looping back to *q*3​.
* *q*1​, *q*3​: Indicate these as accepting states.

1. Consider the language *L* over the alphabet *a*,*b*,*c*} defined as follows:

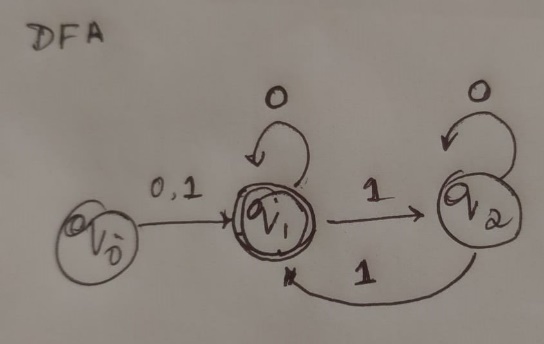
L={w∣ w is a palindrome and the number of ’a’s in w is a prime number}.

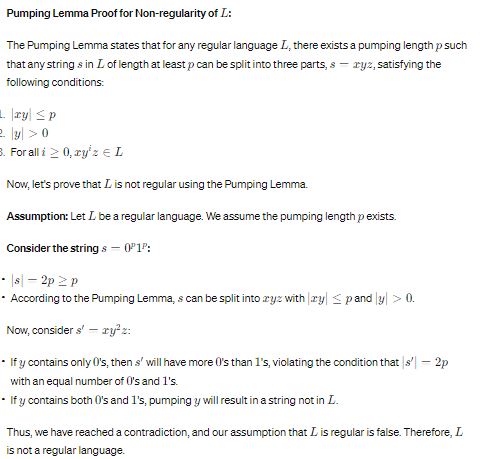
*D*esign a regular expression that represents the language *L*. Explain the components of your regular expression and how it ensures the given conditions for the language.

Solution: (a(bb\*|cc\*)a|(ba(bb\*|cc\*)ab)|(ca(bb\*|cc\*)ac)|b|c)\*

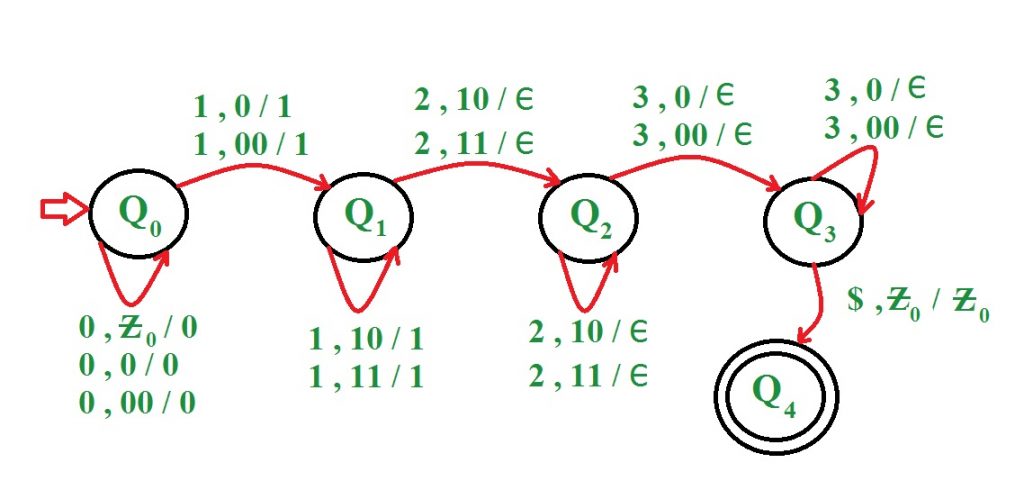
1. Design a DFA that recognizes the language *L* over the alphabet {0,1}where *L* consists of all binary strings that are palindromes and have an odd number of '1's. Provide a detailed explanation of the states, transitions, and the reasoning behind your design choices.

Prove that the language L you defined in Part 1 is not regular by using the Pumping Lemma for regular languages. Provide a step-by-step proof, explicitly stating the assumptions made and justifying each step.





1. Convert the Below PDA to CFG



1. The PDA has the following transition rules:

R1: δ(q, ε, S) = {(q, SX) | (q, 1SY) | (q, ε)}

R2: δ(q, ε, X) = {(q, 1)}

R3: δ(q, ε, Y) = {(q, )}

R4: δ(q, , ) = {(q, ε)}

R5: δ(q, 1, 1) = {(q, ε)}

Give ID for the following strings

1. 10111011101
2. 0104
3. ε
4. 100011101
5. 000111
6. Convert the below grammar to Turing Machine

S → X | ε

X → aY | Y

Y → aY | bZ | ε

Z → bZ | ε

(Hint: number of occurrences of 'a' is a perfect square)

