03-Expressões Regulares

1. Utilizando as expressões regulares do slide anterior, apresente 5 strings representados a partir de cada expressão.

Expressão que representa as strings formados sobre o alfabeto {a,b}, que contenham o substring bb. L(E) = (a+b)\*bb(a+b)\*

R: abbb,bbaab,bbaaab,bbaabaab,bb

Obs: Essa expressão regular representa representa todas as strings que começam e terminam com zero ou mais ocorrências de 'a' ou 'b' e têm a sequência 'bb' em algum ponto no meio.

• Expressão que representa a linguagem das palavras que tenha aa como prefixo e bb como sufixo, sobre o alfabeto {a,b}. L(E) = aa(a+b)\*bb

R: aaabbb , aabb , aaaabababababbb, aabbaabb

Obs: Essa expressão regular representa todas as strings que começam com duas ocorrências de 'a', seguidas por zero ou mais ocorrências de 'a' ou 'b', e terminam com a sequência 'bb'.

• Expressão que representa todas as palavras que terminam com 00 ou 11, sobre o alfabeto {0,1}. L(E) = (0+1)\*(00+11)

R: 0011, 0101010011, 00001000010100010011 , 001000011 , 0010011

Obs: Essa expressão regular representa cadeias de bits que começam e terminam com os mesmos bits (00 ou 11) e podem conter qualquer combinação de 0s e 1s no meio, incluindo sequências vazias.

2.Crie a expressão regular para cada notação formal das linguagens:

* L = {w ∈ {a, b} | w inicia por b, seguido de zero ou mais a’s}

R: L(E) = ba\*

* L = {w ∈ {a, b} | w termina por aa ou bb}

R: L(E) = (a+b)\*(aa+bb)

* L = {w ∈ {a, b} | cada a em w é imediatamente precedido por um b}

R: L(E) = (b+ba)\*

* L = {w ∈ {a, b} | w tem abab como um substring}

R: L(E) = (a+b)\*(abab)(a+b)\*

* L = {w ∈ {a, b} | w não possui dois a’s consecutivos}

R: L(E) = (a+ )(b+ba)\*

* L = {w ∈ {a, b} | w tenha exatamente dois b’s}

R: L(E) = a\*ba\*ba\*

* L = {w ∈ {0, 1} | w termine com 1 e |w| >=4}

R: L(E) = (0+1)(0+1)(0+1)+ 1

* L = {w ∈ {0, 1} | w tenha o número de 1’s múltiplo de três}

R: L(E) = (0\*10\*10\*10\*)+

3. Reescreva o exercício n° 2 utilizando os metacaracteres padrão UNIX.

a) L = {w ∈ {a, b} | w inicia por b, seguido de zero ou mais a’s}

R: Expressão Regular UNIX: ˆb(a\*)

b) L = {w ∈ {a, b} | w termina por aa ou bb}

R: Expressão Regular UNIX: (a+b)\*(aa+bb)

c) L = {w ∈ {a, b} | cada a em w é imediatamente precedido por um b}

R: Expressão Regular UNIX: (b+ba)\*

d) L = {w ∈ {a, b} | w tem abab como um substring}

R: Expressão Regular UNIX: (a+b)\*(abab)(a+b)\*

e) L = {w ∈ {a, b} | w não possui dois a’s consecutivos}

R: Expressão Regular UNIX: (a+ )(b+ba)\*

f) L = {w ∈ {a, b} | w tenha exatamente dois b’s}

R: Expressão Regular UNIX: a\*ba\*ba\*

g) L = {w ∈ {0, 1} | w termine com 1 e |w| >=4}

R: Expressão Regular UNIX: (0+1)(0+1)(0+1)+1

h) L = {w ∈ {0, 1} | w tenha o número de 1’s múltiplo de três}

R: Expressão Regular UNIX: (0\*10\*10\*10\*)+

4) Continue a aprimorar as seguintes expressões regulares abaixo:

a) Data no formato dd/mm/aaaa

ER: primeira versão: ../../....

aprimorando: [0-9]{2} / [0-9]{2} / [0-9]{4}

R: ((0[1-9])|(1[0-9])|(2[0-9])|(3[0-1]))/((0[1-9])|(1[0-2]))/[0-9]{4}.

b) Hora no formato hh:mm

ER: primeira versão: ..:..

aprimorando: [0-9]{2}:[0-9]{2}

R: ([01][0-9]|2[0-3]):[0-5][0-9].

c) Telefone

ER: primeira versão: \+.. (..) . ....-....

R: ^\+(\d{1,3}) \((\d{1,3})\) \d{4,}-\d{4}.

5) Crie as expressões regulares (utilizando os metacaracteres padrão

UNIX) para os seguintes itens:

a) CPF

R: ^[0-9]{3}\.[0-9]{3}\.[0-9]{3}-[0-9]{2}.

b) e-mail

R: ^[a-zA-Z0-9.\_%+-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,}.

c) URL (endereços web com os domínios .com, .net, .edu, .gov, .nom,

todos eles do Brasil)

R: "^(http|https):\/\/[a-zA-Z0-9.-]+\.((com|net|edu|gov|nom)\.br)".

d) Nome de arquivo padrão MSDOS (começando por uma letra, com 8

caracteres e uma extensão de 3 caracteres)

R: ^[a-zA-Z][a-zA-Z0-9]{7}\.[a-zA-Z0-9]{3}.