Na gramática a seguir, os não-terminais estão em maiúsculas e os terminais em minúsculas, todos separados por espaços (estilo BNF).

1. Considere a gramática para um trecho de código em Pascal:

   LISTA -> LISTA ; CMD | CMD
   CMD -> id := EXP
   EXP -> id | num

   (a) Reescreva essa gramática para eliminar a recursão à esquerda. Se desejar, simplifique os não terminais para L, C e E.
   (b) Construa as funções PRIMEIRO e SEGUINTE.
   (c) Monte a tabela de análise preditiva.
   (d) Verifique se a sentença **id := num; id := id** é válida para a linguagem, mostrando os passos de um analisador preditivo tabular.
   (e) Mostre a árvore de derivação conforme os passos do item d.

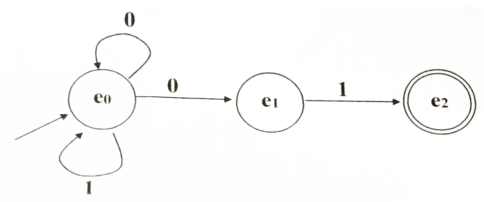
2. Construa um autômato finito para cada uma das seguintes expressões regulares:

   (a) $0\ (0 \mid 1)^*0^+$
   (b) x^+y^*xy

3. Considere a gramática R -> R = R | R R | R * | **a** | **b**

   (a) A gramática é ambígua? Utilize a sentença **b * a = b** para justificar sua resposta.
   (b) Verifique se a gramática é recursiva à esquerda. Caso afirmativo, elimine a recursividade.
   (c) Elimine o vazio das produções em (b).
   (d) Utilizando as regras de produção criadas por você em (c), construa uma árvore de derivação para a sentença **b * a = b**.

4. Informe a expressão regular aceita pelo autômato finito abaixo. Construa a tabela de transição de estados para o autômato.



5. Considere o autômato finito M = (e0, e1, e2, a, b, c, d, δ, e0, e2), onde:

   δ(e0, a) = e0 δ(e0, b) = e0 δ(e0, c) = e1 δ(e1, d) = e1 δ(e1, a) = e2

   (a) Represente o autômato finito por meio de uma diagrama (grafo).
   (b) Apresente a expressão regular que representa a linguagem reconhecida pelo autômato acima.
   (c) Verifique se a cadeia **abaabcdda** é aceita, mostrando a sequência de movimentos executados pelo autômato.