2. **Implementação**

   Criar um **programa** que, dada uma GLUD G que gera uma linguagem regular L, execute as seguintes operações:

   (a) Construa o **autômato finito determinístico (AFD)** M equivalente a G usando o algoritmo visto em aula;

   (b) Dada uma **lista de palavras**, apresente todas as palavras w da lista tal que $w \in ACEITA(M)$;

   (c) Determina se L é **finita ou infinita**.

2 Instruções

- O programa do item 2 pode ser implementado usando qualquer linguagem de programação, desde que o código fonte seja bem documentado;

- O **formato do arquivo de entrada** contendo a definição da GR deve seguir o seguinte padrão:

```
<G>=({<var1,...,varN>},{<ter1>,...,<terN>},<prod>,<ini>)
<prod>
<var1> -> <ter1> <var2>
<var2> -> <ter2>
<var3> ->
...
<varN> -> <terN> <varN>
```

onde:

$< G >$: nome dado à GLUD;

$< vari >$: para $1 \leq i \leq N$, com $N \in \mathbb{N}$ e $N \geq 1$, descreve um nome de variável da GLUD;

$< teri >$: para $1 \leq i \leq N$, com $N \in \mathbb{N}$ e $N \geq 1$, descreve um nome de terminal da GLUD;

$< prod >$: nome do conjunto de produções;

$< ini >$: indica o nome do símbolo inicial da GLUD.

Observe que a variável v_3 pode derivar a palavra vazia

Exemplo:

```
MinhaGLUD=({A,B,C},{a,b,c},prod,A)
prod
A -> aB
B -> bC
A -> b
C ->
C -> cA
```

- Todas as operações do programa devem seguir os algoritmos vistos em aula para garantir o resultado correto. **Qualquer otimização ou alteração deve ser devidamente documentada e associada a uma argumentação de correção**;

- Todas as entradas (GLUD e lista de palavras) devem ser fornecidas **via seleção de arquivo**, por teclado ou interface gráfica.