

## Manual de Usuário: Autômato com Pilha - linguagem $\{a^n b^n \mid n \geq 0\}$

Este manual descreve a implementação e utilização de um autômato com pilha (PDA - Pushdown Automaton) em Python. A documentação descreve a funcionalidade do programa, explica cada passo do código e apresenta exemplos e resultados.

### 1. Dependências

Certifique-se de ter as seguintes bibliotecas instaladas: *graphviz*, *time*, *os*.

### 2. Função grafo

A função *grafo* cria e exibe um diagrama de estados do autômato usando *graphviz*. Cria um grafo dirigido *Digraph*, define a direção do grafo de esquerda para direita e o tamanho do diagrama, adiciona os estados (*q0*, *q1*, *qf*), onde *qf* é um estado final (representado com dupla borda), adiciona as transições com etiquetas no formato leitura, desempilha, empilha e gera/exibe o gráfico em formato PNG.

### 3. Classe PDA

A classe PDA representa um autômato com pilha que reconhece a linguagem  $\{a^n b^n \mid n \geq 0\}$ . Em seus atributos são verificados:

- Pilha: Lista que representa a pilha, inicializada com o símbolo inicial ?.
- Estado: Estado atual do autômato, inicializado para *q0*.
- Transições: Dicionário que define as transições do autômato. Cada chave é uma tupla (estado, entrada, topo da pilha) e o valor é uma tupla (novo estado, símbolos para empilhar).

Seu funcionamento opera com o construtor *"init"* que inicializa a pilha, o estado e define as transições do autômato. A seguir tem a função *"verificação\_final"* que verifica se a palavra é aceita pelo autômato. Para próxima função faz o *"teste\_caractere"* que processa cada caractere da entrada, atualizando o estado e a pilha conforme as transições. E finalmente têm-se a função *"str"* que retorna uma *string* que representa o estado atual e a pilha do autômato.

### 4. Função *"test\_pda"*

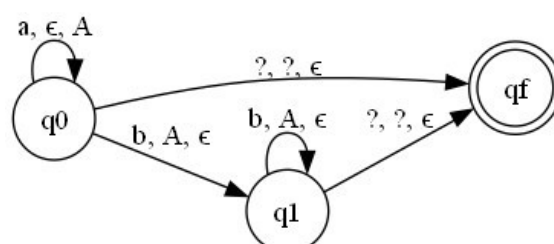
A função *"test\_pda"* anima a operação do PDA ao processar uma palavra de entrada. Seu funcionamento conta o número de *a*'s e *b*'s na palavra e para cada caractere, atualiza o estado e a pilha do PDA, exibindo o processo no terminal. Após processar a palavra, verifica a aceitação. Se aceito, gera o grafo do autômato.

### 5. Exemplo de Uso

Para exemplo temos a função as opções abaixo.

```
palavra = "ab" --- pda = PDA()
```

Inicializa-se a função *test\_pda(pda, palavra)*. Conforme descrito ela recebe a palavra e ser testada e sua classe. Caso se chegue a um estado de aceitação gera/exibe o grafo do autômato (segue abaixo). Caso contrário usuário recebe apenas a informação de rejeitado.



### Considerações Finais

Este código implementa um PDA básico para a linguagem  $\{a^n b^n \mid n \geq 0\}$  com visualização do processo e do grafo de estados. É uma boa base para entender a teoria dos autômatos e a manipulação de pilhas. Para uma aplicação mais complexa, pode-se expandir as transições e estados conforme necessário.