

Teoria da Computação

2023/1 - Trabalho da disciplina

Enunciado

Escreva um programa em C ou C++ que lê a descrição de um autômato à pilha (PDA) e uma sequência de palavras e, para cada palavra dada, determina se ela é ou não é aceita pelo PDA.

A entrada inicia com uma linha contendo dois inteiros Q e T , sendo Q o número de estados e T o número de transições. Considere que os estados são $\{q_0, q_1, \dots, q_{Q-1}\}$; que q_0 é o estado inicial; que o alfabeto de entrada são as letras minúsculas (isto é, $\Sigma = \{a, b, c, \dots, z\}$); que o alfabeto da pilha são as letras maiúsculas (isto é, $\Gamma = \{A, B, C, \dots, Z\}$); e que a letra Z é o símbolo inicialmente na pilha.

As próximas T linhas da entrada descrevem as transições do PDA. Cada transição é dada por uma linha contendo $i \ c \ T \ X \dots \ j$, indicando a transição (do estado q_i para o estado q_j) $c, T/X \dots$ (isto é, que consome a letra c da entrada, desempilha T da pilha e empilha $X \dots$ na pilha, sendo X seu novo topo). Por exemplo, a linha $0 \ a \ Z \ AZ \ 1$ indica a transição $a, Z/AZ$ de q_0 para q_1 .

O símbolo $\&$ indica a palavra vazia ε . Desta forma, se $c = \&$, a transição não consome da entrada; se $T = \&$, a transição não desempilha; e se $XY \dots = \&$, a transição não empilha.

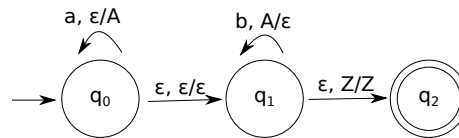
Por fim, a próxima linha contém F , o número de estados finais, seguida de uma linha contendo F inteiros (entre 0 e $Q - 1$) indicando quais são os estados finais do PDA.

Após a descrição do PDA, a entrada conterá várias palavras, uma por linha. Para cada palavra w dada, seu programa deve imprimir w : **sim** se ela é aceita pelo PDA, ou **w: nao** caso contrário.

A entrada termina com uma linha contendo $*$.

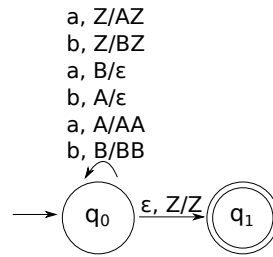
Exemplos

Como exemplo, o seguinte PDA reconhece a linguagem $\{a^n b^n \mid n \in \mathbb{N}\}$:



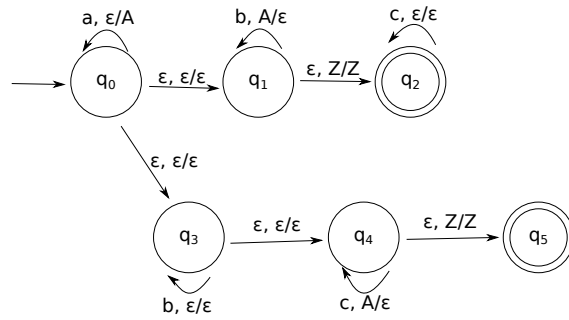
Exemplo de entrada	Exemplo de saída
3 4	ab: sim
0 a & A 0	aabb: sim
0 & & & 1	aabba: nao
1 b A & 1	abb: nao
1 & Z Z 2	aab: nao
1	aaabbb: sim
2	aaaaabbbb: nao
ab	aaaaabbbb: sim
aabb	aaaaabbbb: nao
aabba	
abb	
aab	
aaabbb	
aaaaabbbb	
aaaaabbbb	
aaaaabbbb	
*	

O seguinte PDA reconhece a linguagem $\{w \in \{a, b\}^* \mid w \text{ contém o mesmo número de } as \text{ e } bs \}$:



Exemplo de entrada	Exemplo de saída
2 7 0 a Z AZ 0 0 b Z BZ 0 0 a B & 0 0 b A & 0 0 a A AA 0 0 b B BB 0 0 & Z Z 1 1 1 aabb abb abba aabbbbbaaab aabbbbabab *	aabb: sim abb: nao abba: sim aabbbbbaaab: sim aabbbbabab: nao

O seguinte PDA reconhece a linguagem $\{a^i b^j c^k \mid i = j \text{ ou } i = k\}$:



Exemplo de entrada	Exemplo de saída
6 10	aabbccc: sim
0 a & A 0	aabcc: sim
0 & & & 1	aabbcc: sim
1 b A & 1	abbccc: nao
1 & Z Z 2	abbcc: nao
2 c & & 2	
0 & & & 3	
3 b & & 3	
3 & & & 4	
4 c A & 4	
4 & Z Z 5	
2	
2 5	
aabbccc	
aabcc	
aabbcc	
abbccc	
abbcc	
*	

Implementação

- O trabalho deve ser feito em C ou em C++;
- Você pode assumir que o PDA terá no máximo 100 estados, 100 transições e 100 empilhamentos por transição, e que cada palavra dada terá no mínimo 1 e no máximo 100 letras;
- O tempo de execução da sua solução não será levado em consideração na correção do trabalho. Entretanto, seu programa deve terminar em tempo “razoável” (< 1 min) para os exemplos dados neste documento;
- O trabalho poderá ganhar até 10 *pontos extras* (e valer ao todo 110 pontos) se, para cada palavra da entrada aceita pelo PDA, sua computação também seja impressa, no formato dado no seguinte exemplo:

Exemplo de entrada	Exemplo de saída
<pre>2 7 0 a Z AZ 0 0 b Z BZ 0 0 a B & 0 0 b A & 0 0 a A AA 0 0 b B BB 0 0 & Z Z 1 1 1 aabb abb abba aabbbbbaaab aabbbbabab *</pre>	<pre>aabb: sim (q0, aabb, Z) - (q0, abb, AZ) - (q0, bb, AAZ) - (q0, b, AZ) - (q0, &, Z) - (q1, &, Z). abb: nao abba: sim (q0, abba, Z) - (q0, bba, AZ) - (q0, ba, Z) - (q0, a, BZ) - (q0, &, Z) - (q1, &, Z). aabbbbbaaab: sim (q0, aabbbbbaaab, Z) - (q0, abbbbbaaab, AZ) - (q0, bbbbbaaab, AAZ) - (q0, bbbbaaab, AZ) - (q0, bbbaaab, Z) - (q0, bbaaab, BZ) - (q0, aaab, BBZ) - (q0, aab, BZ) - (q0, ab, Z) - (q0, b, AZ) - (q0, &, Z) - (q1, &, Z). aabbbbabab: nao</pre>

Orientações

- O trabalho pode ser feito por equipes de *até* 2 (dois) estudantes;
- Submeta, via *Moodle*, um pacote (zip ou tar.gz) contendo todo o código-fonte do trabalho, além de um arquivo de texto (txt) onde conste:
 - O nome de todos os integrantes da equipe;
 - Toda informação que a equipe julgar relevante para a correção (como *bugs* conhecidos, detalhes de implementação, escolhas de projeto, etc.)
- Comente adequadamente seus códigos para facilitar a correção.
- Atenção: a correção será parcialmente automatizada, e a saída do programa será testada com outras entradas além das fornecidas como exemplo. *Siga **fielmente** o formato de saída dado nos exemplos*, sob pena de grande redução da nota;
- Certifique-se que seu programa compila e funciona antes de submetê-lo;
- O trabalho deve ser entregue até **23:59 de 21 de Maio de 2023**, apenas via *Moodle*. Trabalhos entregues por outros meios ou fora do prazo não serão aceitos. É suficiente que o trabalho seja submetido por apenas um estudante da equipe;
- Trabalhos detectados como cópia, plágio ou comprados receberão **todos** a nota 0 (**ZERO**) e estarão sujeitos a abertura de Processo Administrativo Disciplinar Discente.