Exercício-Programa: Implementação do algoritmo CYK para Gramáticas Livres do Contexto

Objetivo:

Desenvolver um programa para processamento de Gramáticas Livres do Contexto (GLCs). Dada as especificações de GLCs na Forma Normal de Chomsky e um conjunto de cadeias $w \in \Sigma^*$, seu programa deve determinar, para cada gramática *G* e cada cadeia de entrada *w*, se *G* gera *w*. O programa deverá ser desenvolvido em C, Python, R ou Java, e deverá ser executável via linha de comando do terminal.

A chamada do programa será feita através de linha de comando, pelo nome:

Implementação em C:

glc.exe

Implementação em Java:

java glc

Implementação em Python:

py glc.py

Implementação em R:

Rscript glc.r

Entrada:

O programa terá os seguintes arquivos de entrada:

- inp-glc.txt:
  Conterá a especificação da GLC (na Forma Normal de Chomsky).

  A primeira linha conterá o campo:
  k
  no qual k é o número de gramáticas a serem testadas

  A segunda linha conterá os campos:

$$q_1 \ t_1 \ s_1$$

onde q_1 é o número de variáveis, t_1 é o número de símbolos terminais e s_1 é o número de regras de substituição referentes à gramática G_1. Note que todos esses parâmetros devem ser estritamente positivos.

A terceira linha conterá a lista de variáveis da gramática G_1, separadas por espaços. A variável inicial deve aparecer em primeiro lugar na lista.

A quarta linha conterá a lista de terminais da gramática G_1, separados por espaços.

As linhas 5 a 5 + s_1 - 1 conterão as regras de substituição, em uma das formas abaixo:
Variável => &
Variável => terminal
Variável => Variável Variável

As linhas 5 + s_1 adiante conterão as especificações da próxima gramática, e assim por diante até o fim da especificação da gramática G_k.

Observações:

– Nas regras de substituição, as variáveis e símbolos serão separados por espaços;

– O símbolo reservado & representará cadeia vazia.

O exemplo a seguir ilustra a especificação para duas GLCs.

Primeira GLC:

$S_0 \rightarrow$ & | *A T* | *B U* | *S S* | *A B* | *B A*
S → *A T* | *B U* | *S S* | *A B* | *B A*
T → *S B*
U → *S A*
A → *a*
B → *b*

Segunda GLC:

$S_1 \rightarrow$ AB
A → *0*
B → *1*

Arquivo inp-glc.txt correspondente a essas duas gramáticas:

2
6 2 15
S0 S T U A B
a b

S0 => &
S0 => A T
S0 => B U
S0 => S S
S0 => A B
S0 => B A
S => A T
S => B U
S => S S
S => A B
S => B A
T => S B
U => S A
A => a
B => b
3 2 3
S1 A B
0 1
S1 => AB
A => 0
B => 1

- inp-cadeias.txt:
  Conterá as cadeias a serem avaliadas.

  A primeira linha conterá um inteiro n_1 representando o número de cadeias a serem processadas pela gramática G_1.

  As $n_1$linhas conterão uma cadeia, onde os símbolos serão separados por espaços.

  A linha $n_1 + 1$ conterá o inteiro n_2, indicando o número de cadeias a serem processadas pela gramática G_2. E assim por diante até o inteiro n_k, onde k é o número de gramáticas de teste.

  Ex:

  3
  &
  a
  a b a a b b
  3
  &
  0 1
  0 0 1

Saída:

O arquivo de saída será:

- out-status.txt:
  Conterá k linhas, sendo que cada linha *i* conterá o vetor de status (0 = rejeita, 1 = aceita) das cadeias de entrada segundo a gramática G_i. Esses dígitos serão separados por espaços.

  Para o exemplo anterior, o arquivo conteria as linhas:

  1 0 1
  0 1 0
  (Note que o tamanho de cada linha pode variar, pois depende da quantidade de cadeias de entrada para cada gramática)

Entrega do trabalho:

- O trabalho poderá ser feito em grupos de ATÉ dois alunos, devidamente identificados na primeira linha do código-fonte.

- Deverá ser entregue um diretório compactado (formato .zip) contendo o arquivo fonte e o executável (ou classe Java compilada). O diretório deve ser nomeado na forma *d_numerousp1_numerousp2.zip*
  Exemplo: d_1234567_7654321.zip

- O módulo principal deve ter o nome glc.c, glc.java, glc.py ou glc.r. Nas 1as linhas do código fonte deverá constar os nomes e números USP dos membros do grupo.

- O código-fonte deverá ser compilável via comando gcc, javac ou py. Se desenvolver seu programa em IDEs como Eclipse ou Netbeans, certifique-se de que seu programa seja compilável sob as condições aqui expostas.

- Inclua também, no diretório de seu ep, um arquivo chamado LEIAME.TXT contendo
  - A linguagem utilizada
  - A versão utilizada do compilador ou ambiente
  - IDE utilizada (se for o caso) e sua versão
  - Sistema operacional em que desenvolveu e compilou seus programas
  - a linha de comando exata a ser digitada no prompt do sistema operacional para compilar seu programa

- O trabalho deverá ser submetido via E-DISCIPLINAS.

  Não é necessário que os dois alunos do grupo enviem o código-fonte, basta uma entrega por grupo.

- Além da correção do programa, será considerada a qualidade da documentação do código fonte.

- Se houver evidência de plágio entre trabalhos de grupos distintos, os mesmos serão desconsiderados.