Universidade Federal do Rio Grande do Norte
Programa de Pós-Graduação em Tecnologia da Informação
Disciplina de Linguagens de Domínio Específico

Primeiro Trabalho de Implementação

Versão 1.10, de 29/01/2021

1 Objetivo

O objetivo deste trabalho é implementar *List*, uma linguagem de manipulação de listas.

Para essa implementação, recomenda-se a geração de código para a linguagem Python. Caso o aluno queira, pode usar uma abordagem diferente. Em caso de dúvidas, converse com o professor.

2 Sintaxe de *List*

A gramática a seguir descreve a sintaxe de *List*:

$$
\begin{aligned}
programa \rightarrow\ & comando+ \\
comando \rightarrow\ & NOME\ \text{`=`}\ exp\ /\ \text{`print`}\ exp \\
exp \rightarrow\ & exp\ \text{`.`}\ exp\ /\ exp\ \text{`+`}\ exp\ /\ NOME \\
& /\ lista\ /\ INT\ /\ \text{`false`}\ /\ \text{`true`} \\
lista \rightarrow\ & \text{`[`}\ exp\ (\text{`,`}\ exp) *\ \text{`]`} \\
NOME \rightarrow\ & [a-zA-Z]+ \\
INT \rightarrow\ & [0-9]+
\end{aligned}
$$

3 Semântica de *List*

A seguir, discutimos a semântica dos operadores da linguagem.

O operador **=** é usado para atribuir o valor da expressão à direita do operador para o identificador à esquerda dele.

O operador **+** é usado para somar valores. Caso essa operação envolva valores inteiros, o resultado deve ser a soma desses valores. Se a soma envolve valores booleanos, o resultado dessa operação é dependente da implementação.

Dadas duas listas l_1 e l_2, a operação $l_1 + l_2$ deve resultar em uma lista l_3 com o mesmo número de elementos de l_1, onde cada elemento na posição i de l_3 é igual à soma dos elementos de l_1 e l_2 na posição i. Caso não haja

elemento em l_2 na posição i, o elemento na posição i de l_3 será igual ao elemento na posição i de l_1.

Abaixo, temos alguns exemplos de somas em um programa *List*, onde o texto entre parênteses denota o resultado de uma operação e não faz parte do programa:

```
01  a = [1, 2]
02  b = [3, 4]
03  print a + b      (imprime [4, 6])
04  print 3 + 5      (imprime 8)
05  c = [1, b, 10]
06  d = [2, a]
07  print c + d      (imprime [3, [4, 6], 10])
08  print 1 + false  (não especificado)
```

O operador **.** é usado para concatenar listas. Caso essa operação envolva valores inteiros ou booleanos, o resultado dessa operação é dependente da implementação.

Dadas duas listas l_1 e l_2, a operação $l_1.l_2$ deve resultar em uma lista que consiste dos elementos de l_1 seguidos pelos elementos de l_2.

Abaixo, temos alguns exemplos do uso da operador **.** em um programa *List*, onde o texto entre parênteses denota o resultado de uma operação e não faz parte do programa:

```
01  a = [1, 2]
02  b = [3, 4]
03  print a . b      (imprime [1, 2, 3, 4])
04  print 3 . 5      (não especificado)
05  c = [1, b, 10]
06  d = [2, a]
07  print c . d      (imprime [1, [3, 4], 10, 2, [1, 2]])
08  print 1 . false  (não especificado)
```

4 Requisitos da Implementação

O compilador a ser desenvolvido deve ler programas *List* a partir de um arquivo e gerar um arquivo de saída na linguagem de alto nível de destino.

Em caso de erro sintático no arquivo *List*, o compilador deve indicar o erro e não deve ser gerado arquivo de saída.

A execução do arquivo de saída deve produzir um resultado de acordo com a semântica da linguagem *List*.

5 Envio do Trabalho

O código fonte, os arquivos de teste, e um breve relatório do trabalho devem ser enviados até às 14h00 do dia 05/02. Envie um arquivo *.zip* com todos os

arquivos relevantes do projeto.

No SIGAA, será permitido o reenvio das tarefas, sendo considerado somente o último arquivo enviado.

O relatório deve descrever a estratégia de implementação utilizada, qual semântica foi implementada para as operações cujo significado não foi especificado, como foram realizados os testes, bem como outros detalhes que o aluno julgar relevantes.

6 Outras Questões

Em caso de dúvidas, entre em contato com o professor da disciplina.