Processamento de linguagens

Construção semanticamente sustentada de um interpretador e de um compilador para a linguagem *Natrix*

Simão Melo de Sousa, DI-UBI

Este trabalho é parte constituinte da avaliação prática da Unidade Curricular de código 11567 designada por Processamento de Linguagens, na sua edição de 2019-2020.

Entrega do trabalho

O enunciado introduz várias metas intermédias que devem ser entendidas como momentos em que as parte relacionadas devem ser concluídas para uma boa realização do trabalho ao longo do semestre.

O trabalho como um todo deve ser entregue na data final. A modalidade de entrega toma a forma de um arquivo tar comprimido (*nome.tgz*) em que *nome* é o identificador do vosso grupo. este arquivo deve conter todos os ficheiros fontes necessários à compilação assim como um `Makefile` completo (as entradas `all` e `clean` devem estar presentes).

Este arquivo deverá igualmente conter o relatório que descreve o trabalho feito, as escolhas (de desenho, etc.) tomadas, a documentação do código e o manual do utilizador. É igualmente esperada que seja preparada uma apresentação para a respectiva defesa.

Introdução

O objectivo deste trabalho é implementar de forma incremental:

- uma semântica operacional;
- uma gramática (LR) com os respectivos *lexer* e *parser*;
- um pequeno interpretador (com base na semântica operacional proposta);
- um pequeno compilador;

para uma linguagem de programação elementar, chamada *Natrix*. .

A linguagem *Natrix*

Apresentamos a linguagem *Natrix* pelo exemplo. É esperado que sejam capazes de propôr complementos/ajustes (justificados!) caso entendam que a linguagem precise de ser estendida ou alterada.

o ficheiro natrix.nx introduz tais exemplos.

Exercício 1: (*Programar* Natrix)

- Escrever vários programas "clássicos" em *Natrix* (factorial, procura do maior elemento, etc.). O objectivo é obter um conjunto pequeno mas representativos de programas para testar o projecto a medida que este for completado.
- Numa pasta diferente, escrever programas (ou alterar do ponto anterior) de tal forma que sejam propositadamente introduzidos erros (de sintaxe, de léxico, de tipagem, problemas de intervalo etc.). Estes programas servirão para testar os casos negativos que se espera que o resultado do projeto detecte.

Prazo: 29/10/2019

1 Análise léxica, sintáctica e árvore de sintaxe abstracta

Exercício 2: (*Gramática*)
Defina uma gramática formal para a linguagem *Natrix*. Poderá usar a sintaxe EBNF caso seja conveniente.

Exercício 3: (*Parsing and Lexing*)
Defina um analisador léxico e um analisador sintáctico para a linguagem *Natrix*. Os analisadores construídos deverão ter em conta uma gestão apropriada dos erros que possam surgir (por enquanto léxicos e sintácticos).

Exercício 4: (*Árvore de sintaxe abstracta*)
Defina o tipo das árvore de sintaxe para a linguagem *Natrix*.

Exercício 5: (*Lexing e Parsing (bis)*)
Modifique os analisadores léxicos e sintácticos por forma a que seja construída uma árvore de sintaxe abstracta no caso de uma análise bem sucedida. O parser e o lexer construídos deverão, mais uma vez, assinalar de forma clara e precisa os erros.

Prazo: 19/11/2019

2 Análise Semântica

Exercício 6: (*Semântica Operacional*)
Defina uma semântica operacional para o subconjunto básico da linguagem *Natrix* (sem os tipos intervalo, por exemplo). É deixado ao vosso critério a definição deste subconjunto. É no entanto necessário comunicar ao doente a escolha.

Exercício 7: (*Semântica Operacional (opcional)*)
Integre na semântica definida no exercício anterior o resto da linguagem *Natrix*.

Exercício 8: (*Interpretador*)
Implemente um interpretador com base na semântica definida nos pontos anteriores.

Prazo: 10/12/2019

3 Geração de Código

Exercício 9: (*Compilador*)
Implemente o gerador de código para o maior núcleo possível da linguagem *Natrix*.

Prazo: última semana de aulas