Centro Universitário de Belo Horizonte – Uni-BH
Instituto de Engenharia e Tecnologia
Curso: Ciência da Computação
Disciplina: Compiladores
Professor: Moisés Henrique Ramos Pereira

Trabalho Prático II – Análise Sintática

Descrição do trabalho

Nesta etapa, você deverá implementar um analisador sintático descendente (top-down) para a linguagem *Pyscal*, cuja descrição encontra-se no enunciado do trabalho prático I.

Seu analisador sintático deverá ser um analisador de uma única passada. Dessa forma, ele deverá interagir com o analisador léxico para obter os tokens do arquivo-fonte. Como resultado, ele deverá retornar uma árvore sintática abstrata, representando o programa analisado. Você deve implementar seu analisar sintático utilizando o algoritmo de Parser Preditivo Recursivo (Funções) ou o algorimto do Parser Preditivo Não-Recursivo (Tabela de Parser).

Seu analisador sintático deve atualizar a tabela de símbolos com os identificadores reconhecidos pelo analisador léxico e seus respectivos tipos nas regras de declarações. Dessa forma, à medida que novos tokens ID forem reconhecidos no momento da declaração, esses deverão ser inseridos na tabela de símbolos.

O analisador sintático deverá reportar possíveis erros ocorridos no programa-fonte. O analisador deverá informar qual o erro encontrado e sua localização no arquivo-fonte. A recuperação de erros a ser adotada deverá ser o Método do Pânico. Contudo, se o número de erros ultrapassar o limite de 5 erros, o programa deverá abortar a análise. Os tokens de sincronismo podem ser ";", "." e ")".

Para implementar o analisador sintático, você deverá modificar a estrutura gramatical da linguagem. Você deverá adequá-la às restrições de precedência de operadores, eliminar a recursividade à esquerda e fatorar a gramática, ou seja, essa gramática não é LL(1). Portanto, você deverá verificar as regras que infringem as restrições das gramáticas LL(1) e adaptá-las para tornar a gramática LL(1). Antes disso, você deve tratar operações de gramáticas formais, ou seja, um trecho com * é o fecho de Kleene (de 0 até várias produções), um trecho sob + é o fecho de Kleene positivo (de 1 até várias produções) e trechos sob o símbolo ? ocorrem 1 vez ou podem nem ocorrer.

O que fazer?

1. Tratar a ambiguidade da gramática referente à precedência de operadores na variável Expressao;
2. Tratar os trechos descritos por expressões regulares da gramática formal;
3. Eliminar a recursividade à esquerdar e fatorar a gramática;
4. Implementar os algoritmos de Parser Preditivo Recursivo ou Não-Recursivo;
5. Modelar e implementar a árvore abstrata sintática.

O que entregar?

1. A nova versão da gramática;
2. Programa com todos os arquivos-fonte (será apresentado em laboratório);
3. Testes realizados com programas corretos e errados (no mínimo, 3 certos e 3 errados).

Para avaliar a correção, o programa deverá exibir a árvore sintática em pré-ordem e depois os erros sintáticos ocorridos, informando as respectivas linhas e colunas.

Pontuação extra (2 pontos)

Pode-se tratar o método do pânico na recuperação de erros sintáticos construindo um conjunto de tokens de sincronismo para cada variável (não-terminal) da gramática. Uma abordagem bastante usada é atribuir a esses conjuntos os respectivos conjuntos de tokens Follow das variáveis. Para isso, você deve computar os conjuntos First e Follow de cada não-terminal.