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

Trabalho Prático III – Análise Semântica

Descrição do trabalho

Nesta etapa, você deverá implementar um analisador semântico para a linguagem *JaCa+*, cuja descrição inicial encontra-se no enunciado do trabalho prático I (porém, você deve seguir a gramática que você alterou no trabalho prático anterior).

Deve-se usar uma implementação de tabela de símbolos e árvore sintática abstrata com nodos correspondentes às conversões implícitas de tipos.

Itens a verificar (10 pontos):

1. Declaração duplicada no mesmo escopo;
2. Utilização antes da declaração;
3. Verificação de compatibilidade de tipos entre *double* e *int* resultando em *double*;
4. Verificação do tipo resultante das expressões dos comandos *if, while* e das operações lógicas e relacionais, que deve ser *boolean*;
5. Verificação se o tipo do índice em um vetor é inteiro;
6. Em uma atribuição a expressão que recebe o valor deve ser um identificador (variável) ou uma posição de uma variável vetor (v[1]) compatível com o valor atribuído;
7. Número e tipos dos parâmetros das chamadas de métodos com a declaração desses métodos;
8. Verificação se o *return* de um método retorna uma expressão compatível com o tipo de retorno do método;
9. Verificação da não atribuição de métodos, cujo tipo de retorno seja *void*.

Além da verificação de tipos, seu compilador deverá gerar código assembly para a JVM (3 pontos) utilizando o software Jasmin. No SOL, durante a semana e a próxima, serão disponibilizados alguns exemplos desse assembly. Embora a maioria dos comandos necessários para a tradução serão contemplados, existem outros comandos que não serão apresentados, mas podem ser obtidos na especificação completa da linguagem em http://jasmin.sourceforge.net/instructions.html (inglês) e na apostila colocada no SOL (português).

Correção da Gramática no TP II

A gramática descrita no TP I e desenvolvida no TP II possui um erro de sintaxe: faltou o token “;” (ponto e vírgula) no final da regra da variável CmdFunc, ou seja, a regra de produção correta é:

CmdFunc → ID "(" (Expressao ("," Expressao)∗)? ")" ";"

Tratando as expressões regulares, conforme feito na etapa anterior, fica:

CmdFunc → ID "(" Parametro ")" ";"
Parametro → Expressao ListaExpressao | λ
ListaExpressao → "," Expressao ListaExpressao | λ

Ou, se foi corretamente observado a função sintática do trecho, simplesmente substituir por ListaParam o trecho com expressões regulares.

O que entregar nesse TP III?

Você deverá entregar nesta etapa:

1. Programa com todos os arquivos fonte (será apresentado em data previamente marcada) ;

2. Testes realizados com programas corretos e errados com relação a verificação de tipos (no mínimo, 3 certos e 3 errados), contendo todas as verificações especificadas.

Para avaliar a correção, o programa deverá exibir a árvore sintática em pré-ordem com o tipo anotado em cada nodo (tipo vazio, erro ou o tipo primitivo correspondente).