Interpretador BC

Emanoel Aurélio Vianna Fabiano, Jessica Arruda Ferreira de Santana, Matheus Cosme Britzke

Draft

Faculdade de Informática – Pontifícia Universidade Católica do Rio Grande do Sul (PUCRS)
Av. Ipiranga, 6681 - Partenon, RS, 90619-900 – Porto Alegre – RS – Brazil

31 de outubro de 2016

{emanoel.fabiano, jessica.santana, matheus.britzke}@acad.pucrs.br

Resumo. *Este artigo tem como objetivo apresentar os resultados do trabalho da disciplina de Compiladores do segundo semestre de 2016. O trabalho foi realizado com o intuito de realizar o desenvolvimento de um interpretador. Utilizando as ferramentas JFlex e ByaccJ.*

1. Introdução

Dentro do escopo da disciplina de Compiladores, o trabalho pode ser resumido assim: Utilize como base a calculadora básica conhecida como BC (basic calculator) existente nos sistema Unix desde a versão 6 de 1975 desenvolvida inicialmente por Robert Morris e Lorinda Cherry da Bell Labs para o desenvolvimento de um interpretador como objetivo do trabalho da disciplina de Compiladores. Pelo curto período para o desenvolvimento deste trabalho algumas funções presentes na calculadora original não estão presentes no escopo deste interpretador.

Para resolver o problema proposto utilizamos os conhecimentos adquiridos em aula sobre a análise léxica, sintática e semântica, além de conhecimentos adquiridos em outras disciplinas passadas para auxiliar no desenvolvimento e melhor entendimento de tópicos deste trabalho proposto. Em seguida os resultados obtidos serão apresentados, bem como as conclusões obtidas no decorrer do trabalho.

2. Desenvolvimento

A calculadora trabalha apenas com os tipos numérico (double) e lógicos (boolean), permite três modos básicos de operações, conhecidos como imediato, atribuição e declaração de funções:

1. Imediato, onde a expressão escrita pelo o usuário em tela é compilada e executada. Exemplo do modo imediato: “2*3+1”. A expressão é avaliada, compilada, deve gerar uma árvore de símbolos conhecida como ASA com o objetivo de auxiliar na interpretação da expressão matemática, o resultado é mostrado em tela para o usuário.
2. Atribuição, onde se torna possível atribuir expressões, valores e variáveis para um identificador, deste modo a expressão é compilada, executada e armazenada no identificador. Exemplo: “f=2^2+2”, onde o identificador “f” está armazenando o resultado da expressão matemática.
3. Declaração de funções, a expressão é compilada e armazenada em uma tabela de funções para uma utilização posterior. Exemplo: define d (n) { return (2*n); }.

Ainda sobre as funcionalidades do interpretador foi requisitado algumas funções para controle com o objetivo de mostrar um pequeno auxílio ao usuário, permitir realizar o carregamento em memória de um conjunto de declarações de funções e gravar todo o conteúdo da tabela de símbolos ou sobre um identificador informado pelo o usuário.

2.1. Desenvolvimento da análise léxica

Dentro do escopo da disciplina a análise léxica em poucas palavras tem como objetivo identificar sequências significativas na entrada: Lexemas, quando um lexema é identificado, gera um token. A análise léxica também utiliza conceitos de linguagens formais já estudado em outras disciplinas, a já conhecida gramática regular para a criação dos autômatos finitos.

Para o desenvolvimento da análise léxica deste trabalho o primeiro passo foi identificar qual conjunto de tokens que existem no escopo deste interpretador. Para isto começamos com o conjunto básico já conhecido e informado no enunciado do trabalho, os operadores matemáticos, todo o universo de operadores matemáticos está presente nesse interpretador.