Trabalho prático 2 - Análise sintática e interpretador.

Construção de Compiladores I

Prof. Rodrigo Ribeiro

12-12-2023

1. Instruções importantes

Nesta seção são apresentadas diversas informações relevantes referentes a entrega do trabalho e orientações a serem seguidas durante a implementação do mesmo.

Leia atentamente antes de começá-lo.

1.1. Equipe de desenvolvimento

O trabalho será desenvolvido por grupos de até dois alunos. Não será permitido o trabalho ser realizado por três ou mais pessoas.

1.2. Escolha da linguagem de programação

O trabalho deverá ser desenvolvido na linguagem Haskell. No que diz respeito a implementação, todas as estruturas de dados referente a modelagem e desenvolvimento do trabalho deverão ser implementadas. Poderão ser usadas apenas bibliotecas de estruturas de dados elementares (listas, filas, pilhas, árvores, etc.), leitura de strings e arquivos ou ferramentas de uso geral, como o gerador de analisadores léxicos Alex e o gerador de analisadores sintáticos Happy.

1.3. Artefatos a serem entregues

Os artefatos a serem entregues são:

- Código fonte do programa;
- Relatório do trabalho
  - O repositório deverá conter o código fonte LaTeXde seu relatório e um makefile para produzir o respectivo arquivo pdf.

Antes de enviar seu trabalho para avaliação, assegure-se que:

- Seu código compila e executa em ambiente Unix / Linux. Programas que não compilam receberão nota zero;

- Todos os fontes a serem enviados têm, em comentário no início do arquivo, nome e matrícula do(s) autor(es) do trabalho;
- Arquivo do relatório tenha a identificação do(s) autor(es) do trabalho;

1.4. Critérios de avaliação

A avaliação será feita mediante análise do código fonte, relatório e estrutura de commits do repositório. Os seguintes fatores serão observados na avaliação do código fonte: corretude do programa, estrutura do código e legibilidade. A corretude se refere à implementação correta de todas as funcionalidades especificadas, i.e., se o programa desenvolvido está funcionando corretamente e não apresenta erros. Os demais fatores avaliados no código fonte são referentes a organização e escrita do trabalho.

Nesse quesito, será observado:

- Estruturas de dados bem planejadas;
- Código modularizado em nível físico (separação em arquivos) e lógico (arquitetura bem definida);
- Legibilidade do código, i.e., nomes adequados de variáveis, funções e comentários úteis no código;

O relatório do código deve conter informações relevantes para compilar, executar e auxiliar no entendimento da estratégia adotada para resolução do problema e do código fonte.

Ressalta-se que no relatório não deve conter cópias do fonte – afinal o seu fonte é um dos artefatos entregue –, porém trechos de códigos podem ser incluídos caso isso facilite a explicação e elucidação das estratégias utilizadas.

O relatório deve apresentar as decisões de projetos tomadas: modificações realizadas na gramática, estrutura da árvore de sintaxe, estratégia de implementação do analisador sintático, estratégia utilizada para implementação do interpretador, detalhes de leitura da entrada, dentre outras informações.

A utilização de LaTeX é **obrigatória**. Trabalhos que não utilizem LaTeX para elaboração do relatório não serão considerados para correção. Todo o código LaTeX construído para produzir o relatório deverá estar presente no repositório junto com um makefile para construção do pdf do relatório.

Tenha atenção ao prazo de entrega, pois não serão permitidas alterações nos repositórios após a data limite.

2. Especificação técnica do trabalho

Ao longo do semestre será construído um compilador para a linguagem lang. Assim, neste segundo trabalho pede-se que sejam implementados o analisador sintático e o interpretador de Lang. Como resultado deste trabalho, espera-se que seu interpretador seja capaz de executar diversos programas simples.

Espera-se que seu interpretador seja capaz de executar corretamente os seguintes algoritmos clássicos (que devem acompanhar a seu trabalho como casos de teste):

- Cálculo de fatorial.
- Cálculo do n-ésimo termo da sequência de Fibonnacci.
- Implementação de um algoritmo de ordenação de arrays.
- Implementação de uma pilha e de suas operações (criação de uma pilha vazia, empilhar, desempilhar, número de elementos).
- Implementação de uma árvore binária e de suas operações (criação de uma árvore vazia, inserir, pesquisar, remover).

Todos esses exemplos devem ser implementados utilizando Lang e possuir casos de teste que comprovem o correto funcionamento destas implementações.

A utilização de *Parsing Expression Grammars* para realização da análise sintática de *Lang* terá um acréscimo de 25% sobre a nota obtida neste trabalho.

3. Entrega do Trabalho

A data da entrega do trabalho será até o dia **21 de janeiro de 2024**.