EXA869 MI Processadores de Linguagens de Programação

Problema 3: Quem fez fez quem não fez não faz mais! Parte 3.

Cronograma

Sessão	Data	Assunto
	29/out	Apresentação Problema 3
1	31/out	Problema 3
2	05/nov	Problema 3
3	07/nov	Problema 3
4	12/nov	Problema 3
5	14/nov	Problema 3
6	19/nov	Problema 3
	21/nov	Ponto facultativo
7	26/nov	Problema 3
8	28/nov	Problema 3
9	03/dez	Problema 3
10	05/12	Problema 3

Analisador Sintático Preditivo Recursivo, método do Pânico, tokens de sincronização, ufa! Esse deu trabalho. E agora vamos para última etapa do *front-end* de um compilador, o analisador semântico.

Problema

Desenvolver um Analisador Semântico para a gramática definida pelo tutor. O analisador deverá ser capaz de identificar TODOS os erros semânticos presentes, alertando ao usuário o tipo do erro e onde ele ocorreu no código.

Regras semânticas a serem implementadas

- Verificação de tipos. A linguagem é fortemente tipada, ou seja, não há conversão de tipos de forma explícita ou implícita.
- Visibilidade de variáveis e constantes. Existem dois escopos: global e local. É permitido o uso de identificadores de mesmo nome, porém em escopos diferentes.
- Existência e duplicidade de variáveis e constantes.
- Verificação da função *main*. Deve existir apenas uma função *main*.

Produto

O analisador semântico poderá ser desenvolvido em dupla ou individualmente. Esta dupla deve ser a mesma do problema 2. O analisador semântico deverá ser implementado na linguagem Python. O código-fonte do analisador deve ser enviado para o seu tutor, pelo classroom da disciplina, até as **23h59m** do dia **07/12/2025**. Será descontado 02 (dois) pontos por um dia de atraso. Após esse prazo, o código não será mais aceito.

Observações

1. A entrada para este analisador é um conjunto de arquivos texto. Estes arquivos de entrada deverão estar em uma pasta na raiz do projeto chamada **files**.
2. A saída do analisador semântico deverá ser através de um conjunto de arquivos de saída (um para cada arquivo de entrada), denominados **X-saida.txt**, onde **X** é o nome do arquivo de entrada. Estes arquivos de saída deverão estar em uma pasta na raiz do projeto chamada **files**.
3. Os arquivos de saída deverão apresentar a lista de erros semânticos, caso existam. Se não houver erros, uma mensagem de "SUCESSO" deve ser gravada no arquivo de saída. O formato dos arquivos de saída será definido pelo tutor e todos devem segui-lo.

Recursos para aprendizagem

- AHO, A. V.; SETHI, S. & ULMMAN, J. D. **Compiladores: princípios, técnicas e ferramentas.** Rio de Janeiro: LTC, 1995.
- LOUDEN, K. C. **Compiladores – Princípios e Práticas.** São Paulo, Thomson, 2004.
- HOPCROFT, J. E. *et al*. **Introdução à Teoria dos Autômatos, Linguagens e Computação**. 1ª edição, Editora Campus, 2002.
- MENEZES, P. F. B. **Linguagens Formais e Autômatos**. 5ª edição, Editora Sagra-Luzzatto, 2005.