

Construção de um Compilador para Pascal Standard

Projeto de Processamento de Linguagens 2025

2025-03-20 by jcr

Objetivo

O objetivo deste projeto é que os alunos desenvolvam um compilador para a linguagem Pascal standard.

O compilador deverá ser capaz de analisar, interpretar e traduzir código Pascal para um formato intermediário e deste para código máquina ou diretamente para código máquina, neste caso da [VM disponibilizada aos alunos](#).

Etapas do Projeto

Análise Léxica

1. Implementar um analisador léxico (lexer) para converter código Pascal numa lista de tokens;
2. Usar a ferramenta `ply.lex`, na implementação do analisador léxico;
3. Identificar palavras-chave, identificadores, números, operadores e símbolos especiais.

Análise Sintática

1. Construir um analisador sintático (parser) para validar a estrutura gramatical do código;
2. Usar a ferramenta `ply.yacc`, na implementação do analisador sintático;

Análise Semântica

- Verificar tipos de dados, declaração de variáveis e coerência do código;

Geração de Código

1. Converter o código Pascal em código da [máquina virtual](#), caso opte por não realizar a etapa seguinte;
2. Ou, gerar uma representação intermédia do programa reconhecido para passar à etapa seguinte.

Otimização de Código (!extra!)

- Melhorar a eficiência do código gerado, eliminando redundâncias e aplicando otimizações locais e globais.

Geração de Código Final (!extra!)

- Converter a representação intermédia para código da VM.

Testes

1. Criar uma série de testes para verificar a correção do compilador;
2. Testar programas Pascal de exemplo e validar a saída.

Exemplos de Programas Pascal

Exemplo 1: Olá, Mundo!

```
program HelloWorld;
begin
  writeln('Ola, Mundo!');
end.
```

Exemplo 2: Fatorial

```
program Fatorial;
var
  n, i, fat: integer;
begin
  writeln('Introduza um número inteiro positivo:');
  readln(n);
  fat := 1;
  for i := 1 to n do
    fat := fat * i;
  writeln('Fatorial de ', n, ': ', fat);
end.
```

Exemplo 3: Verificação de Número Primo

```
program NumeroPrimo;
var
  num, i: integer;
  primo: boolean;
begin
  writeln('Introduza um número inteiro positivo:');
  readln(num);
  primo := true;
  i := 2;
  while (i <= (num div 2)) and primo do
    begin
      if (num mod i) = 0 then
        primo := false;
      i := i + 1;
    end;
  if primo then
    writeln(num, ' é um número primo')
  else
    writeln(num, ' não é um número primo')
end.
```

Exemplo4: Soma de uma lista de inteiros

```
program SomaArray;
var
  numeros: array[1..5] of integer;
  i, soma: integer;
begin
  soma := 0;
  writeln('Introduza 5 números inteiros:');
  for i := 1 to 5 do
  begin
    readln(numeros[i]);
    soma := soma + numeros[i];
  end;

  writeln('A soma dos números é: ', soma);
end.
```

Exemplo5: Conversão binário-decimal

```
program BinarioParaInteiro;

function BinToInt(bin: string): integer;
var
  i, valor, potencia: integer;
begin
  valor := 0;
  potencia := 1;

  for i := length(bin) downto 1 do
  begin
    if bin[i] = '1' then
      valor := valor + potencia;
    potencia := potencia * 2;
  end;

  BinToInt := valor;
end;

var
  bin: string;
  valor: integer;
begin
  writeln('Introduza uma string binária:');
  readln(bin);

  valor := BinToInt(bin);

  writeln('O valor inteiro correspondente é: ', valor);
end.
```

Requisitos

O compilador deve ser capaz de processar programas Pascal standard, incluindo declaração de variáveis, expressões aritméticas, comandos de controle de fluxo (if, while, for), e, opcionalmente, subprogramas (procedure e function).

Resultados a entregar

1. Código-fonte do compilador;
2. Relatório técnico, em LaTeX ou Markdown, explicando a implementação;
3. Testes e exemplos de entrada e saída.

Critérios de Avaliação

1. Correção: O compilador deve processar corretamente programas Pascal;
2. Estrutura: Organização do código e documentação;
3. Funcionalidade: Suporte às principais construções da linguagem;
4. Eficiência: Desempenho do compilador;
5. Demonstração realizada durante a defesa.

Boa sorte e bom trabalho!