UNIVERSIDADE FEDERAL DE ITAJUBÁ
Compiladores – ECOI26
Prof. Paulo José Lage Alvarenga
Instituto de Ciências Tecnológicas

Trabalho Prático

Analisador Léxico e Tabela de Símbolos

Nome: Guilbert Wilkerson Marques Oliveira ***RA: 2019004349***
Nome: Guilherme Henrique Dias Costa ***RA: 2019009908***

Forma de uso do compilador

O uso desse compilador se baseia na seguinte ordem de funcionamento descrita pelo arquivo “test.sh”:

```
#!/bin/bash

# Loop de 1 a 9 para processar os arquivos de entrada
for i in {1..9}; do
  input_file="Test/Entradas/Entrada${i}.txt"
  output_file="Test/Saidas/Saida${i}.txt"

  # Compila o código Java
  javac Yylex.java

  # Executa o programa Java com o arquivo de entrada correspondente e redireciona a saída
  java Yylex "$input_file" > "$output_file"

  # Limpa os arquivos .class gerados durante a compilação
  rm *.class
done
```

O funcionamento de forma geral se resume a compilar o aquivo jflex que gera um yylex e compilar esse arquivo com as entradas de teste(está no arquivo test.sh o código).

Definição da Linguagem:

A partir da gramática da sua linguagem fictícia. Foram especificadas as regras léxicas usando expressões regulares no JFlex para identificar tokens.

Implementação do Analisador Léxico (JFlex):

As regras léxicas foram escritas em um arquivo .flex usando a sintaxe do JFlex. Este arquivo contém especificações para reconhecer padrões léxicos na entrada. Usa expressões regulares para definir os tokens da linguagem.

Execute o JFlex para gerar o código Java do analisador léxico. Isso pode ser feito usando o comando jflex nome-do-arquivo.flex.

Entrada do Compilador:

Ao usar o compilador, forneça o código-fonte da linguagem fictícia como entrada. Isso pode ser feito através de um arquivo de texto ou por meio de entrada padrão.

Análise Léxica:

O analisador léxico lê o código-fonte e gera uma sequência de tokens. Ele fornece esses tokens ao analisador sintático armazenando para o uso no analisador sintático A tabela de símbolos é uma estrutura de dados essencial para o analisador

léxico, pois é usada para armazenar informações sobre os identificadores (como variáveis e palavras-chave) encontrados durante a análise do código-fonte.

O analisador sintático verifica se a sequência de tokens segue as regras definidas na gramática. Se houver erros, mensagens apropriadas são geradas.

Saída do Compilador:

O compilador deve fornecer mensagens informativas sobre o sucesso ou falha da compilação. Se houver erros, indique a localização e natureza desses erros.

Testes:

Realize testes extensivos usando diferentes casos de entrada para garantir que o compilador funciona corretamente.

Descrição da abordagem utilizada na implementação

Compilação e Geração de Analisadores:

Podemos utilizar os comandos jflex MiniLexer.flex e java java_cup.Main -parser MiniParser -symbols MiniSym para gerar os analisadores léxico.

Implementação do Analisador Lexico (yylex.java):

Utilizamos a gramática Mini fornecida no formato .flex para desenvolver o analisador lexico. Este arquivo gerará um arquivo Java correspondente ao analisador lexico.

Integração dos Analisadores e Implementação do Compilador:

Importado os analisadores gerados para o projeto Java é criada uma classe principal que inicializa o analisador léxico e criar a classe Token / TS, depois devemos instruir o jflex criar um novo token e o inserir na TS.
Dessa forma estabelecemos a lógica para conectar a tabela e o lexer, permitindo que todos funcionem conforme necessário.

Entrada do Compilador:

Ao utilizar o compilador, é forneçido as entradas para que os testes sejam executados da linguagem "Mini".

Saída do Compilador e Testes:

Configurado o compilador para fornecer mensagens informativas sobre o sucesso ou falha da compilação. È realize testes extensivos utilizando diferentes casos de entrada para garantir o correto funcionamento do compilador.

Testes

Teste 1:

```
program teste1
declare
   integer a, b, c;
   integer result;
begin
   read (a);
   read (c);
   b := 10;
   result := (a * c)/(b + 5 % 345 -3) ;
   write(result)
end
```

Resultado Saida:

line: 1 char: 0 match: --program--

```
action [33] { Yytoken token = new Yytoken(0, yytext(), yyline, yychar, yychar + 7, "Palavra reservada: program");
   if (!tabelaSimbolos.contemToken(token.getText())) {
      tabelaSimbolos.adicionarEntrada(token);
   }
   return token; }
Text: program
Index: 0
Name: Palavra reservada: program
Line: 0
Begin: 0
End: 7
line: 1 char: 7 match: -- --
action [233] { // Ignorar espaços em branco e quebras de linha }
line: 1 char: 8 match: --teste1--
action [193] { Yytoken token = new Yytoken(20, yytext(), yyline, yychar, yychar + yylength(), "Identifier");
   if (!tabelaSimbolos.contemToken(token.getText())) {
      tabelaSimbolos.adicionarEntrada(token);
   }
   return token; }
Text: teste1
Index: 20
Name: Identifier
Line: 0
Begin: 8
End: 14
line: 1 char: 14 match: --\u000A--
action [233] { // Ignorar espaços em branco e quebras de linha }
line: 2 char: 15 match: --\u000A--
action [233] { // Ignorar espaços em branco e quebras de linha }
line: 3 char: 16 match: --declare--
action [41] { Yytoken token = new Yytoken(1, yytext(), yyline, yychar, yychar + 7, "Palavra reservada: declare");
   if (!tabelaSimbolos.contemToken(token.getText())) {
      tabelaSimbolos.adicionarEntrada(token);
   }
   return token; }
Text: declare
Index: 1
Name: Palavra reservada: declare
Line: 2
Begin: 16
End: 23
line: 3 char: 23 match: --\u000A--
action [233] { // Ignorar espaços em branco e quebras de linha }
line: 4 char: 24 match: -- --
action [233] { // Ignorar espaços em branco e quebras de linha }
line: 4 char: 25 match: -- --
action [233] { // Ignorar espaços em branco e quebras de linha }
line: 4 char: 26 match: -- --
action [233] { // Ignorar espaços em branco e quebras de linha }
line: 4 char: 27 match: -- --
action [233] { // Ignorar espaços em branco e quebras de linha }
line: 4 char: 28 match: --integer--
action [57] { Yytoken token = new Yytoken(3, yytext(), yyline, yychar, yychar + 7, "Palavra reservada: integer");
   if (!tabelaSimbolos.contemToken(token.getText())) {
      tabelaSimbolos.adicionarEntrada(token);
   }
   return token; }
Text: integer
Index: 3
Name: Palavra reservada: integer
Line: 3
Begin: 28
End: 35
line: 4 char: 35 match: -- --
action [233] { // Ignorar espaços em branco e quebras de linha }
line: 4 char: 36 match: --a--
action [193] { Yytoken token = new Yytoken(20, yytext(), yyline, yychar, yychar + yylength(), "Identifier");
   if (!tabelaSimbolos.contemToken(token.getText())) {
      tabelaSimbolos.adicionarEntrada(token);
```

```
    }
    return token; }
Text: a
Index: 20
Name: Identifier
Line: 3
Begin: 36
End: 37
line: 4 char: 37 match: --,--
action [260] { Yytoken token = new Yytoken(29, yytext(), yyline, yychar, yychar + yylength(), "Caractere");
    if (!tabelaSimbolos.contemToken(token.getText())) {
        tabelaSimbolos.adicionarEntrada(token);
    }
    return token; }
Text: ,
Index: 29
Name: Caractere
Line: 3
Begin: 37
End: 38
line: 4 char: 38 match: -- --
action [233] { // Ignorar espaços em branco e quebras de linha }
line: 4 char: 39 match: --b--
action [193] { Yytoken token = new Yytoken(20, yytext(), yyline, yychar, yychar + yylength(), "Identifier");
    if (!tabelaSimbolos.contemToken(token.getText())) {
        tabelaSimbolos.adicionarEntrada(token);
    }
    return token; }
Text: b
Index: 20
Name: Identifier
Line: 3
Begin: 39
End: 40
line: 4 char: 40 match: --,--
action [260] { Yytoken token = new Yytoken(29, yytext(), yyline, yychar, yychar + yylength(), "Caractere");
    if (!tabelaSimbolos.contemToken(token.getText())) {
        tabelaSimbolos.adicionarEntrada(token);
    }
    return token; }
Text: ,
Index: 29
Name: Caractere
Line: 3
Begin: 40
End: 41
line: 4 char: 41 match: -- --
action [233] { // Ignorar espaços em branco e quebras de linha }
line: 4 char: 42 match: --c--
action [193] { Yytoken token = new Yytoken(20, yytext(), yyline, yychar, yychar + yylength(), "Identifier");
    if (!tabelaSimbolos.contemToken(token.getText())) {
        tabelaSimbolos.adicionarEntrada(token);
    }
    return token; }
Text: c
Index: 20
Name: Identifier
Line: 3
Begin: 42
End: 43
line: 4 char: 43 match: --;--
action [260] { Yytoken token = new Yytoken(29, yytext(), yyline, yychar, yychar + yylength(), "Caractere");
    if (!tabelaSimbolos.contemToken(token.getText())) {
        tabelaSimbolos.adicionarEntrada(token);
    }
    return token; }
Text: ;
Index: 29
Name: Caractere
```

```
Line: 3
Begin: 43
End: 44
line: 4 char: 44 match: --\u000A--
action [233] { // Ignorar espaços em branco e quebras de linha }
line: 5 char: 45 match: -- --
action [233] { // Ignorar espaços em branco e quebras de linha }
line: 5 char: 46 match: -- --
action [233] { // Ignorar espaços em branco e quebras de linha }
line: 5 char: 47 match: -- --
action [233] { // Ignorar espaços em branco e quebras de linha }
line: 5 char: 48 match: -- --
action [233] { // Ignorar espaços em branco e quebras de linha }
line: 5 char: 49 match: --integer--
action [57] { Yytoken token = new Yytoken(3, yytext(), yyline, yychar, yychar + 7, "Palavra reservada: integer");
   if (!tabelaSimbolos.contemToken(token.getText())) {
      tabelaSimbolos.adicionarEntrada(token);
   }
   return token; }
Text: integer
Index: 3
Name: Palavra reservada: integer
Line: 4
Begin: 49
End: 56
line: 5 char: 56 match: -- --
action [233] { // Ignorar espaços em branco e quebras de linha }
line: 5 char: 57 match: --result--
action [193] { Yytoken token = new Yytoken(20, yytext(), yyline, yychar, yychar + yylength(), "Identifier");
   if (!tabelaSimbolos.contemToken(token.getText())) {
      tabelaSimbolos.adicionarEntrada(token);
   }
   return token; }
Text: result
Index: 20
Name: Identifier
Line: 4
Begin: 57
End: 63
line: 5 char: 63 match: --;--
action [260] { Yytoken token = new Yytoken(29, yytext(), yyline, yychar, yychar + yylength(), "Caractere");
   if (!tabelaSimbolos.contemToken(token.getText())) {
      tabelaSimbolos.adicionarEntrada(token);
   }
   return token; }
Text: ;
Index: 29
Name: Caractere
Line: 4
Begin: 63
End: 64
line: 5 char: 64 match: --\u000A--
action [233] { // Ignorar espaços em branco e quebras de linha }
line: 6 char: 65 match: --begin--
action [49] { Yytoken token = new Yytoken(2, yytext(), yyline, yychar, yychar + 5, "Palavra reservada: begin");
   if (!tabelaSimbolos.contemToken(token.getText())) {
      tabelaSimbolos.adicionarEntrada(token);
   }
   return token; }
Text: begin
Index: 2
Name: Palavra reservada: begin
Line: 5
Begin: 65
End: 70
line: 6 char: 70 match: --\u000A--
action [233] { // Ignorar espaços em branco e quebras de linha }
line: 7 char: 71 match: -- --
action [233] { // Ignorar espaços em branco e quebras de linha }
```

```
line: 7 char: 72 match: -- --
action [233] { // Ignorar espaços em branco e quebras de linha }
line: 7 char: 73 match: -- --
action [233] { // Ignorar espaços em branco e quebras de linha }
line: 7 char: 74 match: -- --
action [233] { // Ignorar espaços em branco e quebras de linha }
line: 7 char: 75 match: --read--
action [137] { Yytoken token = new Yytoken(13, yytext(), yyline, yychar, yychar + 4, "Palavra reservada: read");
   if (!tabelaSimbolos.contemToken(token.getText())) {
      tabelaSimbolos.adicionarEntrada(token);
   }
   return token; }
Text: read
Index: 13
Name: Palavra reservada: read
Line: 6
Begin: 75
End: 79
line: 7 char: 79 match: -- --
action [233] { // Ignorar espaços em branco e quebras de linha }
line: 7 char: 80 match: --(--
action [260] { Yytoken token = new Yytoken(29, yytext(), yyline, yychar, yychar + yylength(), "Caractere");
   if (!tabelaSimbolos.contemToken(token.getText())) {
      tabelaSimbolos.adicionarEntrada(token);
   }
   return token; }
Text: (
Index: 29
Name: Caractere
Line: 6
Begin: 80
End: 81
line: 7 char: 81 match: --a--
action [193] { Yytoken token = new Yytoken(20, yytext(), yyline, yychar, yychar + yylength(), "Identifier");
   if (!tabelaSimbolos.contemToken(token.getText())) {
      tabelaSimbolos.adicionarEntrada(token);
   }
   return token; }
Text: a
Index: 20
Name: Identifier
Line: 6
Begin: 81
End: 82
line: 7 char: 82 match: --)--
action [260] { Yytoken token = new Yytoken(29, yytext(), yyline, yychar, yychar + yylength(), "Caractere");
   if (!tabelaSimbolos.contemToken(token.getText())) {
      tabelaSimbolos.adicionarEntrada(token);
   }
   return token; }
Text: )
Index: 29
Name: Caractere
Line: 6
Begin: 82
End: 83
line: 7 char: 83 match: --;--
action [260] { Yytoken token = new Yytoken(29, yytext(), yyline, yychar, yychar + yylength(), "Caractere");
   if (!tabelaSimbolos.contemToken(token.getText())) {
      tabelaSimbolos.adicionarEntrada(token);
   }
   return token; }
Text: ;
Index: 29
Name: Caractere
Line: 6
Begin: 83
End: 84
line: 7 char: 84 match: --\u000A--
```

```
action [233] { // Ignorar espaços em branco e quebras de linha }
line: 8 char: 85 match: -- --
action [233] { // Ignorar espaços em branco e quebras de linha }
line: 8 char: 86 match: -- --
action [233] { // Ignorar espaços em branco e quebras de linha }
line: 8 char: 87 match: -- --
action [233] { // Ignorar espaços em branco e quebras de linha }
line: 8 char: 88 match: -- --
action [233] { // Ignorar espaços em branco e quebras de linha }
line: 8 char: 89 match: --read--
action [137] { Yytoken token = new Yytoken(13, yytext(), yyline, yychar, yychar + 4, "Palavra reservada: read");
   if (!tabelaSimbolos.contemToken(token.getText())) {
      tabelaSimbolos.adicionarEntrada(token);
   }
   return token; }
Text: read
Index: 13
Name: Palavra reservada: read
Line: 7
Begin: 89
End: 93
line: 8 char: 93 match: -- --
action [233] { // Ignorar espaços em branco e quebras de linha }
line: 8 char: 94 match: --(--
action [260] { Yytoken token = new Yytoken(29, yytext(), yyline, yychar, yychar + yylength(), "Caractere");
   if (!tabelaSimbolos.contemToken(token.getText())) {
      tabelaSimbolos.adicionarEntrada(token);
   }
   return token; }
Text: (
Index: 29
Name: Caractere
Line: 7
Begin: 94
End: 95
line: 8 char: 95 match: --c--
action [193] { Yytoken token = new Yytoken(20, yytext(), yyline, yychar, yychar + yylength(), "Identifier");
   if (!tabelaSimbolos.contemToken(token.getText())) {
      tabelaSimbolos.adicionarEntrada(token);
   }
   return token; }
Text: c
Index: 20
Name: Identifier
Line: 7
Begin: 95
End: 96
line: 8 char: 96 match: --)--
action [260] { Yytoken token = new Yytoken(29, yytext(), yyline, yychar, yychar + yylength(), "Caractere");
   if (!tabelaSimbolos.contemToken(token.getText())) {
      tabelaSimbolos.adicionarEntrada(token);
   }
   return token; }
Text: )
Index: 29
Name: Caractere
Line: 7
Begin: 96
End: 97
line: 8 char: 97 match: --;--
action [260] { Yytoken token = new Yytoken(29, yytext(), yyline, yychar, yychar + yylength(), "Caractere");
   if (!tabelaSimbolos.contemToken(token.getText())) {
      tabelaSimbolos.adicionarEntrada(token);
   }
   return token; }
Text: ;
Index: 29
Name: Caractere
Line: 7
```

```
Begin: 97
End: 98
line: 8 char: 98 match: --\u000A--
action [233] { // Ignorar espaços em branco e quebras de linha }
line: 9 char: 99 match: -- --
action [233] { // Ignorar espaços em branco e quebras de linha }
line: 9 char: 100 match: -- --
action [233] { // Ignorar espaços em branco e quebras de linha }
line: 9 char: 101 match: -- --
action [233] { // Ignorar espaços em branco e quebras de linha }
line: 9 char: 102 match: -- --
action [233] { // Ignorar espaços em branco e quebras de linha }
line: 9 char: 103 match: --b--
action [193] { Yytoken token = new Yytoken(20, yytext(), yyline, yychar, yychar + yylength(), "Identifier");
   if (!tabelaSimbolos.contemToken(token.getText())) {
      tabelaSimbolos.adicionarEntrada(token);
   }
   return token; }
Text: b
Index: 20
Name: Identifier
Line: 8
Begin: 103
End: 104
line: 9 char: 104 match: -- --
action [233] { // Ignorar espaços em branco e quebras de linha }
line: 9 char: 105 match: --:=--
action [253] { Yytoken token = new Yytoken(27, yytext(), yyline, yychar, yychar + 2, "Assing_op");
   if (!tabelaSimbolos.contemToken(token.getText())) {
      tabelaSimbolos.adicionarEntrada(token);
   }
   return token; }
Text: :=
Index: 27
Name: Assing_op
Line: 8
Begin: 105
End: 107
line: 9 char: 107 match: -- --
action [233] { // Ignorar espaços em branco e quebras de linha }
line: 9 char: 108 match: --10--
action [237] { Yytoken token = new Yytoken(25, yytext(), yyline, yychar, yychar + yylength(), "Interger");
   if (!tabelaSimbolos.contemToken(token.getText())) {
      tabelaSimbolos.adicionarEntrada(token);
   }
   return token; }
Text: 10
Index: 25
Name: Interger
Line: 8
Begin: 108
End: 110
line: 9 char: 110 match: --;--
action [260] { Yytoken token = new Yytoken(29, yytext(), yyline, yychar, yychar + yylength(), "Caractere");
   if (!tabelaSimbolos.contemToken(token.getText())) {
      tabelaSimbolos.adicionarEntrada(token);
   }
   return token; }
Text: ;
Index: 29
Name: Caractere
Line: 8
Begin: 110
End: 111
line: 9 char: 111 match: --\u000A--
action [233] { // Ignorar espaços em branco e quebras de linha }
line: 10 char: 112 match: -- --
action [233] { // Ignorar espaços em branco e quebras de linha }
line: 10 char: 113 match: -- --
```

```
action [233] { // Ignorar espaços em branco e quebras de linha }
line: 10 char: 114 match: -- --
action [233] { // Ignorar espaços em branco e quebras de linha }
line: 10 char: 115 match: -- --
action [233] { // Ignorar espaços em branco e quebras de linha }
line: 10 char: 116 match: --result--
action [193] { Yytoken token = new Yytoken(20, yytext(), yyline, yychar, yychar + yylength(), "Identifier");
   if (!tabelaSimbolos.contemToken(token.getText())) {
      tabelaSimbolos.adicionarEntrada(token);
   }
   return token; }
Text: result
Index: 20
Name: Identifier
Line: 9
Begin: 116
End: 122
line: 10 char: 122 match: -- --
action [233] { // Ignorar espaços em branco e quebras de linha }
line: 10 char: 123 match: --:=--
action [253] { Yytoken token = new Yytoken(27, yytext(), yyline, yychar, yychar + 2, "Assing_op");
   if (!tabelaSimbolos.contemToken(token.getText())) {
      tabelaSimbolos.adicionarEntrada(token);
   }
   return token; }
Text: :=
Index: 27
Name: Assing_op
Line: 9
Begin: 123
End: 125
line: 10 char: 125 match: -- --
action [233] { // Ignorar espaços em branco e quebras de linha }
line: 10 char: 126 match: --(--
action [260] { Yytoken token = new Yytoken(29, yytext(), yyline, yychar, yychar + yylength(), "Caractere");
   if (!tabelaSimbolos.contemToken(token.getText())) {
      tabelaSimbolos.adicionarEntrada(token);
   }
   return token; }
Text: (
Index: 29
Name: Caractere
Line: 9
Begin: 126
End: 127
line: 10 char: 127 match: --a--
action [193] { Yytoken token = new Yytoken(20, yytext(), yyline, yychar, yychar + yylength(), "Identifier");
   if (!tabelaSimbolos.contemToken(token.getText())) {
      tabelaSimbolos.adicionarEntrada(token);
   }
   return token; }
Text: a
Index: 20
Name: Identifier
Line: 9
Begin: 127
End: 128
line: 10 char: 128 match: -- --
action [233] { // Ignorar espaços em branco e quebras de linha }
line: 10 char: 129 match: --*--
action [217] { Yytoken token = new Yytoken(23, yytext(), yyline, yychar, yychar + yylength(), "Mulop");
   if (!tabelaSimbolos.contemToken(token.getText())) {
      tabelaSimbolos.adicionarEntrada(token);
   }
   return token; }
Text: *
Index: 23
Name: Mulop
Line: 9
```

```
Begin: 129
End: 130
line: 10 char: 130 match: -- --
action [233] { // Ignorar espaços em branco e quebras de linha }
line: 10 char: 131 match: --c--
action [193] { Yytoken token = new Yytoken(20, yytext(), yyline, yychar, yychar + yylength(), "Identifier");
   if (!tabelaSimbolos.contemToken(token.getText())) {
      tabelaSimbolos.adicionarEntrada(token);
   }
   return token; }
Text: c
Index: 20
Name: Identifier
Line: 9
Begin: 131
End: 132
line: 10 char: 132 match: --)--
action [260] { Yytoken token = new Yytoken(29, yytext(), yyline, yychar, yychar + yylength(), "Caractere");
   if (!tabelaSimbolos.contemToken(token.getText())) {
      tabelaSimbolos.adicionarEntrada(token);
   }
   return token; }
Text: )
Index: 29
Name: Caractere
Line: 9
Begin: 132
End: 133
line: 10 char: 133 match: --/--
action [217] { Yytoken token = new Yytoken(23, yytext(), yyline, yychar, yychar + yylength(), "Mulop");
   if (!tabelaSimbolos.contemToken(token.getText())) {
      tabelaSimbolos.adicionarEntrada(token);
   }
   return token; }
Text: /
Index: 23
Name: Mulop
Line: 9
Begin: 133
End: 134
line: 10 char: 134 match: --(--
action [260] { Yytoken token = new Yytoken(29, yytext(), yyline, yychar, yychar + yylength(), "Caractere");
   if (!tabelaSimbolos.contemToken(token.getText())) {
      tabelaSimbolos.adicionarEntrada(token);
   }
   return token; }
Text: (
Index: 29
Name: Caractere
Line: 9
Begin: 134
End: 135
line: 10 char: 135 match: --b--
action [193] { Yytoken token = new Yytoken(20, yytext(), yyline, yychar, yychar + yylength(), "Identifier");
   if (!tabelaSimbolos.contemToken(token.getText())) {
      tabelaSimbolos.adicionarEntrada(token);
   }
   return token; }
Text: b
Index: 20
Name: Identifier
Line: 9
Begin: 135
End: 136
line: 10 char: 136 match: -- --
action [233] { // Ignorar espaços em branco e quebras de linha }
line: 10 char: 137 match: --+--
action [209] { Yytoken token = new Yytoken(22, yytext(), yyline, yychar, yychar + yylength(), "Addop");
   if (!tabelaSimbolos.contemToken(token.getText())) {
```

```
        tabelaSimbolos.adicionarEntrada(token);
    }
    return token; }
Text: +
Index: 22
Name: Addop
Line: 9
Begin: 137
End: 138
line: 10 char: 138 match: -- --
action [233] { // Ignorar espaços em branco e quebras de linha }
line: 10 char: 139 match: --5--
action [237] { Yytoken token = new Yytoken(25, yytext(), yyline, yychar, yychar + yylength(), "Interger");
    if (!tabelaSimbolos.contemToken(token.getText())) {
        tabelaSimbolos.adicionarEntrada(token);
    }
    return token; }
Text: 5
Index: 25
Name: Interger
Line: 9
Begin: 139
End: 140
line: 10 char: 140 match: -- --
action [233] { // Ignorar espaços em branco e quebras de linha }
line: 10 char: 141 match: --%--
action [185] { Yytoken token = new Yytoken(19, yytext(), yyline, yychar, yychar + 1, "Comentario / Mod");
    if (!tabelaSimbolos.contemToken(token.getText())) {
        tabelaSimbolos.adicionarEntrada(token);
    }
    return token; }
Text: %
Index: 19
Name: Comentario / Mod
Line: 9
Begin: 141
End: 142
line: 10 char: 142 match: -- --
action [233] { // Ignorar espaços em branco e quebras de linha }
line: 10 char: 143 match: --345--
action [237] { Yytoken token = new Yytoken(25, yytext(), yyline, yychar, yychar + yylength(), "Interger");
    if (!tabelaSimbolos.contemToken(token.getText())) {
        tabelaSimbolos.adicionarEntrada(token);
    }
    return token; }
Text: 345
Index: 25
Name: Interger
Line: 9
Begin: 143
End: 146
line: 10 char: 146 match: -- --
action [233] { // Ignorar espaços em branco e quebras de linha }
line: 10 char: 147 match: -----
action [209] { Yytoken token = new Yytoken(22, yytext(), yyline, yychar, yychar + yylength(), "Addop");
    if (!tabelaSimbolos.contemToken(token.getText())) {
        tabelaSimbolos.adicionarEntrada(token);
    }
    return token; }
Text: -
Index: 22
Name: Addop
Line: 9
Begin: 147
End: 148
line: 10 char: 148 match: --3--
action [237] { Yytoken token = new Yytoken(25, yytext(), yyline, yychar, yychar + yylength(), "Interger");
    if (!tabelaSimbolos.contemToken(token.getText())) {
        tabelaSimbolos.adicionarEntrada(token);
```

```
    }
    return token; }
Text: 3
Index: 25
Name: Interger
Line: 9
Begin: 148
End: 149
line: 10 char: 149 match: --)--
action [260] { Yytoken token = new Yytoken(29, yytext(), yyline, yychar, yychar + yylength(), "Caractere");
    if (!tabelaSimbolos.contemToken(token.getText())) {
        tabelaSimbolos.adicionarEntrada(token);
    }
    return token; }
Text: )
Index: 29
Name: Caractere
Line: 9
Begin: 149
End: 150
line: 10 char: 150 match: -- --
action [233] { // Ignorar espaços em branco e quebras de linha }
line: 10 char: 151 match: --;--
action [260] { Yytoken token = new Yytoken(29, yytext(), yyline, yychar, yychar + yylength(), "Caractere");
    if (!tabelaSimbolos.contemToken(token.getText())) {
        tabelaSimbolos.adicionarEntrada(token);
    }
    return token; }
Text: ;
Index: 29
Name: Caractere
Line: 9
Begin: 151
End: 152
line: 10 char: 152 match: --\u000A--
action [233] { // Ignorar espaços em branco e quebras de linha }
line: 11 char: 153 match: -- --
action [233] { // Ignorar espaços em branco e quebras de linha }
line: 11 char: 154 match: -- --
action [233] { // Ignorar espaços em branco e quebras de linha }
line: 11 char: 155 match: -- --
action [233] { // Ignorar espaços em branco e quebras de linha }
line: 11 char: 156 match: -- --
action [233] { // Ignorar espaços em branco e quebras de linha }
line: 11 char: 157 match: --write--
action [145] { Yytoken token = new Yytoken(14, yytext(), yyline, yychar, yychar + 5, "Palavra reservada: write");
    if (!tabelaSimbolos.contemToken(token.getText())) {
        tabelaSimbolos.adicionarEntrada(token);
    }
    return token; }
Text: write
Index: 14
Name: Palavra reservada: write
Line: 10
Begin: 157
End: 162
line: 11 char: 162 match: --(--
action [260] { Yytoken token = new Yytoken(29, yytext(), yyline, yychar, yychar + yylength(), "Caractere");
    if (!tabelaSimbolos.contemToken(token.getText())) {
        tabelaSimbolos.adicionarEntrada(token);
    }
    return token; }
Text: (
Index: 29
Name: Caractere
Line: 10
Begin: 162
End: 163
line: 11 char: 163 match: --result--
```

```
action [193] { Yytoken token = new Yytoken(20, yytext(), yyline, yychar, yychar + yylength(), "Identifier");
   if (!tabelaSimbolos.contemToken(token.getText())) {
      tabelaSimbolos.adicionarEntrada(token);
   }
   return token; }
Text: result
Index: 20
Name: Identifier
Line: 10
Begin: 163
End: 169
line: 11 char: 169 match: --)--
action [260] { Yytoken token = new Yytoken(29, yytext(), yyline, yychar, yychar + yylength(), "Caractere");
   if (!tabelaSimbolos.contemToken(token.getText())) {
      tabelaSimbolos.adicionarEntrada(token);
   }
   return token; }
Text: )
Index: 29
Name: Caractere
Line: 10
Begin: 169
End: 170
line: 11 char: 170 match: --\u000A--
action [233] { // Ignorar espaços em branco e quebras de linha }
line: 12 char: 171 match: --end--
action [105] { Yytoken token = new Yytoken(9, yytext(), yyline, yychar, yychar + 3, "Palavra reservada: end");
   if (!tabelaSimbolos.contemToken(token.getText())) {
      tabelaSimbolos.adicionarEntrada(token);
   }
   return token; }
Text: end
Index: 9
Name: Palavra reservada: end
Line: 11
Begin: 171
End: 174
line: 12 char: 174 match: <<EOF>>
action [271] { Yytoken token = new Yytoken(29, yytext(), yyline, yychar, yychar + 5, "<EOF>");
   tabelaSimbolos.imprimirTabela();
   return token; }
Tabela de Símbolos:
ID: 0, Lexema: program, Nome: Palavra reservada: program, Index: 0
ID: 1, Lexema: declare, Nome: Palavra reservada: declare, Index: 1
ID: 2, Lexema: begin, Nome: Palavra reservada: begin, Index: 2
ID: 3, Lexema: integer, Nome: Palavra reservada: integer, Index: 3
ID: 4, Lexema: decimal, Nome: Palavra reservada: decimal, Index: 4
ID: 5, Lexema: if, Nome: Palavra reservada: if, Index: 5
ID: 6, Lexema: then, Nome: Palavra reservada: then, Index: 6
ID: 7, Lexema: else, Nome: Palavra reservada: else, Index: 7
ID: 8, Lexema: for, Nome: Palavra reservada: for, Index: 8
ID: 9, Lexema: end, Nome: Palavra reservada: end, Index: 9
ID: 10, Lexema: do, Nome: Palavra reservada: do, Index: 10
ID: 11, Lexema: to, Nome: Palavra reservada: to, Index: 11
ID: 12, Lexema: while, Nome: Palavra reservada: while, Index: 12
ID: 13, Lexema: read, Nome: Palavra reservada: read, Index: 13
ID: 14, Lexema: write, Nome: Palavra reservada: write, Index: 14
ID: 15, Lexema: or, Nome: Palavra reservada: or, Index: 15
ID: 16, Lexema: and, Nome: Palavra reservada: and, Index: 16
ID: 17, Lexema: mod, Nome: Palavra reservada: mod, Index: 17
ID: 18, Lexema: not, Nome: Palavra reservada: not, Index: 18
ID: 19, Lexema: teste1, Nome: Identifier, Index: 20
ID: 20, Lexema: a, Nome: Identifier, Index: 20
ID: 21, Lexema: ,, Nome: Caractere, Index: 29
ID: 22, Lexema: b, Nome: Identifier, Index: 20
ID: 23, Lexema: c, Nome: Identifier, Index: 20
ID: 24, Lexema: ;, Nome: Caractere, Index: 29
ID: 25, Lexema: result, Nome: Identifier, Index: 20
ID: 26, Lexema: (, Nome: Caractere, Index: 29
```

ID: 27, Lexema:), Nome: Caractere, Index: 29
ID: 28, Lexema: :=, Nome: Assing_op, Index: 27
ID: 29, Lexema: 10, Nome: Interger, Index: 25
ID: 30, Lexema: *, Nome: Mulop, Index: 23
ID: 31, Lexema: /, Nome: Mulop, Index: 23
ID: 32, Lexema: +, Nome: Addop, Index: 22
ID: 33, Lexema: 5, Nome: Interger, Index: 25
ID: 34, Lexema: %, Nome: Comentario / Mod, Index: 19
ID: 35, Lexema: 345, Nome: Interger, Index: 25
ID: 36, Lexema: -, Nome: Addop, Index: 22
ID: 37, Lexema: 3, Nome: Interger, Index: 25
Text:
Index: 29
Name: <EOF>
Line: 11
Begin: 174
End: 179

OBS*: Demais testes estão incluídos nos arquivos.