



CENTRO FEDERAL DE EDUCAÇÃO TECNOLÓGICA DE MINAS GERAIS
DEPARTAMENTO DE COMPUTAÇÃO



IMPLEMENTAÇÃO COMPILADOR

ANALISADOR SINTÁTICO

AUTOR: ALAN FERREIRA LEITE SANTOS

MARCOS JUNIO DA SILVA XAVIER

SAMUEL FILIPE DOS SANTOS

ORIENTADOR: KECIA ALINE MARQUES

FERREIRA

2 de abril de 2021, BELO HORIZONTE



CENTRO FEDERAL DE EDUCAÇÃO TECNOLÓGICA DE MINAS GERAIS
DEPARTAMENTO DE COMPUTAÇÃO



Alan Ferreira Leite Santos
Marcos Junio da Silva Xavier
Samuel Filipe dos Santos

IMPLEMENTAÇÃO COMPILADOR PARTE II - ANALISADOR SINTÁTICO

Trabalho realizado em consonância com os princípios teóricos aprendidos durante o semestre a fim de criar, por etapas, um compilador para uma determinada linguagem proposta pela orientadora.

Orientador: Kécia Aline Marques Ferreira

2 de abril de 2021, Belo Horizonte

RESUMO

Na disciplina de Compiladores, ofertada pelo CEFET-MG, iremos construir um compilador ao longo do semestre, este, dividido em etapas. Na primeira etapa, entregamos a implementação do analisador léxico e a Tabela de Símbolos em Java. Nesta segunda etapa, estamos entregando a análise do programa levando em consideração a derivação da linguagem. Tais conhecimentos obtidos com a disciplina orientados pela Dr Kecia Ferreira, para que seja possível, utilizamos como base os conceitos abordados pelo Aho, em seu livro : "Compiladores Princípios ,Técnicas e Ferramentas"

Palavras Chaves: java,compilador

ABSTRACT

In the course of Compilers, offered by CEFET-MG, we will build a compiler throughout the semester, this one, divided into stages. In the first step, we delivered the implementation of the lexical analyzer and the Symbol Table in Java. In this second stage, we are delivering the analysis of the program taking into account the language derivation. Such knowledge obtained with the discipline guided by Dr Kecia Ferreira, to make it possible, we use as a basis the concepts addressed by Aho, in his book: "Compilers Principles, Techniques and Tools "

Key-words: java,compiler

LISTA DE FIGURAS

Figura 1 – Planejamento Analise TP Compiladores	7
---	---

LISTA DE TABELAS

Tabela 1 – Relação das Palavras Reservadas da Linguagem	2
---	---

LISTA DE SIMBOLOS

$\backslash n$	Quebra de Linha
(Abertura de Parenteses
)	Fechamento de parenteses
*	Operador de Multiplicação
+	Operador de Adição
−	Operador de Subtração
/	Operador de Divisão
:=	Operador de Atribuição
;	Indicador de Fim de Linha
<	Menor que
<=	Menor ou igual a
<>	Diferente
>	Maior que
>=	Maior ou igual a
%	Representa o inicio de um comentário

Sumário

1. INTRODUÇÃO	1
2. PALAVRAS RESERVADAS	2
3. MELHORIAS GRADATIVAS	3
4. FORMA DE UTILIZAÇÃO.....	4
5. DOCUMENTAÇÃO.....	6
6. PLANEJAMENTO	7
7. TESTES REALIZADOS	8
7.1. Teste 1	8
7.2. Teste 2	12
7.3. Teste 3	14
7.4. Teste 4	19
7.5. Teste 5	25
7.6. Teste 6	33
7.7. Teste 7	40
7.8. Teste 8	44
8. METODOLOGIA	46
9. SUGESTÕES PARA TRABALHOS FUTUROS	57
10. REFERÊNCIAS.....	58
10.1. Livros	58
10.2. Programas	58
10.3. Site.....	58

1. INTRODUÇÃO

Um compilador é um programa de computador (ou um grupo de programas) que, a partir de um código fonte escrito em uma linguagem compilada, cria um programa semanticamente equivalente, porém escrito em outra linguagem, código objeto. Classicamente, um compilador traduz um programa de uma linguagem textual facilmente entendida por um ser humano para uma linguagem de máquina, específica para um processador e sistema operacional. Atualmente, porém, são comuns compiladores que geram código para uma máquina virtual que é, depois, interpretada por um interpretador. Ele é chamado compilador por razões históricas; nos primeiros anos da programação automática, existiam programas que percorriam bibliotecas de sub-rotinas e as reunia, ou compilava, as sub-rotinas necessárias para executar uma determinada tarefa.

O nome "compilador" é usado principalmente para os programas que traduzem o código fonte de uma linguagem de programação de alto nível para uma linguagem de programação de baixo nível (por exemplo, Assembly ou código de máquina). Contudo alguns autores citam exemplos de compiladores que traduzem para linguagens de alto nível como C. Para alguns autores um programa que faz uma tradução entre linguagens de alto nível é normalmente chamado um tradutor, filtro ou conversor de linguagem. Um programa que traduz uma linguagem de programação de baixo nível para uma linguagem de programação de alto nível é um descompilador. Um programa que faz uma tradução entre uma linguagem de montagem e o código de máquina é denominado montador (assembler). Um programa que faz uma tradução entre o código de máquina e uma linguagem de montagem é denominado desmontador (disassembler). Se o programa compilado pode ser executado em um computador cuja CPU ou sistema operacional é diferente daquele em que o compilador é executado, o compilador é conhecido como um compilador cruzado.

2. PALAVRAS RESERVADAS

Para que o compilador seja executado de forma correta, existem algumas regrinhas a serem seguidas, uma delas é a definição de palavras reservadas, essas, são palavras que não devem ser utilizadas em nenhum outro momento a não ser o que foi pre-determinado pelos programadores da linguagem. Como por exemplo: No caso deste compilador, existe a palavra stop, a qual significa que quando o compilador ler esse token, significa que o programa chegou ao final, logo, em nenhum outro local esse token deve ter outro significado a não ser o significado que já foi determinado. Não se pode atribuir uma outra função a uma palavra reservada.

Tabela 1 – Relação das Palavras Reservadas da Linguagem

Palavra Reservada	Significado
init	Início do Programa
stop	Indica a finalização do programa
is	Usado para determinador o tipo de uma variável
integer	Tipo Inteiro para variável
string	Determina a variável como tipo string
real	Determina a variável como tipo Real
if	Determina o início de um bloco de código sob uma condicional
begin	Determina o início de uma condição do if
end	Determina o fim da condição do if
else	Determina a contraposição da condição do if
end else	Indica o Fim da condição do Else if
do	Determina a entrada de um laço
while	determina a condição para o laço Do
read	Prepara para ler e reservar a próxima variável a ser lida
write	Determina que irá imprimir na tela a literal que virá a seguir
not	Determina a negação do valor booleano de uma expressão
or	Determina a soma binária entre dois valores
and	Determina a multiplicação binária entre dois valores

Fonte: Documentação Fornecida pela Orientadora o qual pode ser consultado no diretório do projeto

3. MELHORIAS GRADATIVAS

Para iniciar o desenvolvimento dessa segunda parte do trabalho que faz parte do desenvolvimento do compilador, tivemos que consertar erros que foram relatados na parte anterior (Analisador Léxico) pela nossa orientadora Kecia. Foram pontuados os seguintes erros:

- O resultado do Teste 1 mostrado não corresponde a ele. -0,25.
- Teste 4: não reconheceu o erro de comentário não fechado. -0,25
- Teste 2: ”_valor”tem que ser reportado como erro. -0,25
- Faltou o Teste 5. -0,25

Esses erros passaram despercebidos e foram corrigidos para essa entrega do trabalho.

4. FORMA DE UTILIZAÇÃO

Para que seja possível a utilização deste programa, basta entrar no terminal e navegar até o diretório: "TP_Compilador/src/", e então, digitar o seguinte comando :

```
1 javac Main.java
2 java Main
```

Listing 4.1: Entrada terminal

Acima podemos ver duas linhas de comando, a primeira serve para que possa compilar a classe Main.java e então, é criado o executável e assim, podemos acessar a classe a partir da linha seguinte, java Main .

Definimos qual arquivo de teste que o programa deve compilar dentro da classe Main, no seguinte trecho de código:

```
1 public class Main {
2     public static void main(String[] args) {
3         ArrayList<Token> tokens = new ArrayList<Token> ();
4         Lexer L = null;
5         int line = -5;
6         try {
7             L = new Lexer("codigos_teste/corretos/Teste1.txt");
8             L.adicionapalavras();//Inicia adicionando palavras reservadas
9             System.out.println("**** Tokens lidos ****");
10            // Apenas para entrar no laço
11            Token T = new Token(0, line);
12            while (T.tag != Tag.EOF) {
13                try {
14                    T = L.scan();
15                    if(T.tag == Tag.EOF)
16                        break;
17                    T.imprimeToken(T);
18                    tokens.add(T);
19                    line = T.line;
20                } catch (InvalidTokenException | IOException e) {
21                    System.out.println(e.getMessage());
22                    try {
23                        L.readch();
24                    } catch (IOException e1) {
25                        e1.printStackTrace();
26                    }
27                }
28            }
29            line++;
30            tokens.add(new Token(Tag.EOF, line));
31            //L.imprimirTabela();
```

```
32     Parser P = new Parser(tokens);
33     System.out.println("\n\n\n**** Inicio Parser ****");
34     P.init();
35 } catch (FileNotFoundException e) {
36     e.printStackTrace();
37 }
38 }
39 }
40 }
```

Listing 4.2: Indicando arquivo para teste

Na linha 7 (sete) do trecho do código destacado acima, mostra como determinamos qual programa nosso compilador irá testar. Veja que criamos uma pasta chamada testes para que seja incluídos somente os arquivos de teste do programa. Foram criados 10 arquivos de teste.

5. DOCUMENTAÇÃO

O Analisador Sintático tem a função de solicitar ao Analisador léxico o próximo Tokens desde o início do programa até o final, e após a cada solicitação, verificar se o Token lido está em consonância com o que foi determinado pela gramática do programa.

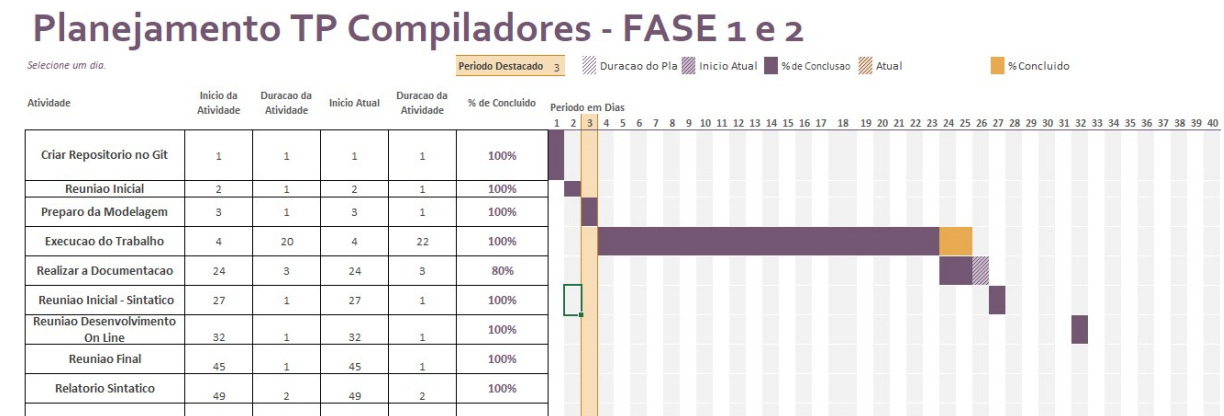
O Projeto foi desenvolvido em conjunto com as plataformas Visual Studio Code e Apache Netbeans, a princípio todo o código seria criado somente "à mão" criando as pastas e arquivos manualmente, porém, devido a algumas facilidades que o Netbeans oferece, o projeto foi alterado em alguns parâmetros para que seja possível abrir no Netbeans. Uma dessas facilidades é a opção de geração da Documentação do projeto, assim o fizemos e esta documentação gerada automaticamente pelo Netbeans se encontra no diretório: `"/TP_Compilador/Projeto/dist/javadoc/"`. Para esta implementação foi adotado o modelo abordado pelo autor do livro base da disciplina: Aho

6. PLANEJAMENTO

Para que todo projeto possa ser bem sucedido, devem se analisar, projetar e estudar minuciosamente e fazer o levantamento de todos os requisitos necessários para o projeto. Apartir das matérias de Modelagem e Desenvolvimento de Sistemas e Engenharia de Software, foi possível adquirir conhecimentos para um bom planejamento do trabalho. Como esse trabalho demonstrou ser extenso e complexo, decidimos por nos planejarmos a fim de evitar que possíveis erros de implementação fosse acarretado durante a fase de Desenvolvimento. Assim, fizemos uma reunião inicial para decidir os pontos importantes e decidir pontos sobre implementação do código. Utilizamos um organizador de tarefas no Excel para acompanhar as tarefas. Tal diagrama será utilizado para as próximas etapas também. Veja abaixo, como ficou a organização do planejamento deste trabalho na figura abaixo.

Figura 1 – Planejamento Analise TP Compiladores

(a) Diagrama Planejamento



Fonte: Desenvolvedores do Código

Tal arquivo se encontra dentro do diretório deste projeto criado pelo Programa Microsoft Excel, podendo ser acesso pelo caminho: "TP_Compilador/Planejamento/

7. TESTES REALIZADOS

Para testar todo o compilador construído, foram propostos sete exemplos de teste, abordando algumas possibilidades de programas a serem compilados. A partir desses arquivos podemos visualizar os seguintes retornos do compilador :

7.1. TESTE 1

```

1
2 init
3 a, b, $c, is integer;
4 result is real;
5 write("Digite o valor de a:")
6 read (a);
7 write("Digite o valor de c:")
8 read (c);
9 b := 10
10 result := (a * c)/(b + 5 - 345);
11 write("O resultado e:");
12 write(result);
13 stop

```

Listing 7.1: Teste1.txt

Para o código acima o compilador obteve a seguinte saída :

```

1 **** Inicio Parser ****
2 Token Consumido(1): < init_program >
3 Token Consumido(2): < identifier, a >
4 Token Consumido(2): < virgula >
5 Token Consumido(2): < identifier, b >
6 Token Consumido(2): < virgula >
7 Token Consumido(2): < identifier, c >
8 Token Consumido(2): < virgula >
9 Error in: identList; Error(2): Token não esperado:< is_decl >
10 Fim de arquivo inesperado.

```

Listing 7.2: Saída para o Código de teste : Teste1.txt

NO exemplo acima, a declaração da última variável foi declarada errada, dessa forma, o programa é abortado.

7.1.1. Correção Do Teste 1

Foram feitas as devidas correções no código acima para adequar as regras da gramática da linguagem. Assim, obtemos o seguinte resultado.

```

1  **** Tokens lidos ****
2  < init_program >
3  < identifier, a >
4  < virgula >
5  < identifier, b >
6  < virgula >
7  Error na Linha(2): Token '$' Nao esperado
8  < identifier, c >
9  < is_decl >
10 < integer_type >
11 < ponto_virgula >
12 < identifier, result >
13 < is_decl >
14 < real_type >
15 < ponto_virgula >
16 < begin >
17 < write >
18 < abre_parent >
19 < literal, "Digite o valor de a:" >
20 < fecha_parent >
21 < ponto_virgula >
22 < read >
23 < abre_parent >
24 < identifier, a >
25 < fecha_parent >
26 < ponto_virgula >
27 < write >
28 < abre_parent >
29 < literal, "Digite o valor de c:" >
30 < fecha_parent >
31 < ponto_virgula >
32 < read >
33 < abre_parent >
34 < identifier, c >
35 < fecha_parent >
36 < ponto_virgula >
37 < identifier, b >
38 < assign >
39 < num, 10 >
40 < ponto_virgula >
41 < identifier, result >
42 < assign >
43 < abre_parent >

```

```

44 < identifier, a >
45 < mult_mulop >
46 < identifier, c >
47 < fecha_parent >
48 < div_mulop >
49 < abre_parent >
50 < identifier, b >
51 < soma_addop >
52 < num, 5 >
53 < menos_addop >
54 < num, 345 >
55 < fecha_parent >
56 < ponto_virgula >
57 < write >
58 < abre_parent >
59 < literal, "0 resultado e:" >
60 < fecha_parent >
61 < ponto_virgula >
62 < write >
63 < abre_parent >
64 < identifier, result >
65 < fecha_parent >
66 < ponto_virgula >
67 < stop_program >
68
69
70
71 **** Inicio Parser ****
72 Token Consumido(1): < init_program >
73 Token Consumido(2): < identifier, a >
74 Token Consumido(2): < virgula >
75 Token Consumido(2): < identifier, b >
76 Token Consumido(2): < virgula >
77 Token Consumido(2): < identifier, c >
78 Token Consumido(2): < is_decl >
79 Token Consumido(2): < integer_type >
80 Token Consumido(2): < ponto_virgula >
81 Token Consumido(3): < identifier, result >
82 Token Consumido(3): < is_decl >
83 Token Consumido(3): < real_type >
84 Token Consumido(3): < ponto_virgula >
85 Token Consumido(4): < begin >
86 Token Consumido(5): < write >
87 Token Consumido(5): < abre_parent >
88 Token Consumido(5): < literal, "Digite o valor de a:" >
89 Token Consumido(5): < fecha_parent >
90 Token Consumido(5): < ponto_virgula >

```

```

91 Token Consumido(6): < read >
92 Token Consumido(6): < abre_parent >
93 Token Consumido(6): < identifier, a >
94 Token Consumido(6): < fecha_parent >
95 Token Consumido(6): < ponto_virgula >
96 Token Consumido(7): < write >
97 Token Consumido(7): < abre_parent >
98 Token Consumido(7): < literal, "Digite o valor de c:" >
99 Token Consumido(7): < fecha_parent >
100 Token Consumido(7): < ponto_virgula >
101 Token Consumido(8): < read >
102 Token Consumido(8): < abre_parent >
103 Token Consumido(8): < identifier, c >
104 Token Consumido(8): < fecha_parent >
105 Token Consumido(8): < ponto_virgula >
106 Token Consumido(9): < identifier, b >
107 Token Consumido(9): < assign >
108 Token Consumido(9): < num, 10 >
109 Token Consumido(9): < ponto_virgula >
110 Token Consumido(10): < identifier, result >
111 Token Consumido(10): < assign >
112 Token Consumido(10): < abre_parent >
113 Token Consumido(10): < identifier, a >
114 Token Consumido(10): < mult_mulop >
115 Token Consumido(10): < identifier, c >
116 Token Consumido(10): < fecha_parent >
117 Token Consumido(10): < div_mulop >
118 Token Consumido(10): < abre_parent >
119 Token Consumido(10): < identifier, b >
120 Token Consumido(10): < soma_addop >
121 Token Consumido(10): < num, 5 >
122 Token Consumido(10): < menos_addop >
123 Token Consumido(10): < num, 345 >
124 Token Consumido(10): < fecha_parent >
125 Token Consumido(10): < ponto_virgula >
126 Token Consumido(11): < write >
127 Token Consumido(11): < abre_parent >
128 Token Consumido(11): < literal, "O resultado e:" >
129 Token Consumido(11): < fecha_parent >
130 Token Consumido(11): < ponto_virgula >
131 Token Consumido(12): < write >
132 Token Consumido(12): < abre_parent >
133 Token Consumido(12): < identifier, result >
134 Token Consumido(12): < fecha_parent >
135 Token Consumido(12): < ponto_virgula >

```

Listing 7.3: Saida Correta para o Codigo de teste : Teste1.txt

7.2. TESTE 2

```

1  init
2  write("Entre com o valor de a:");
3  read (a);
4  b_1 := a * a;
5  write("O valor de b1 e:");
6  write (b_1);
7  b_2 = b + a/2 * (a + 5);
8  write("O valor de b1 e:");
9  Write (b1);
10 stop

```

Listing 7.4: Teste2.txt

Para o código acima o compilador obteve a seguinte saída :

```

1
2 **** Tokens lidos ****
3 < identifier, a >
4 < virgula >
5 Error na Linha(1): Token '_' Nao esperado
6 < identifier, b_1 >
7 < virgula >
8 < identifier, b_2 >
9 Error na Linha(1): Token ':' Nao esperado
10 < integer_type >
11 < ponto_virgula >
12 < init_program >
13 < write >
14 < abre_parent >
15 < literal, "Entre com o valor de a:" >
16 < fecha_parent >
17 < ponto_virgula >
18 < read >
19 < abre_parent >
20 < identifier, a >
21 < fecha_parent >
22 < ponto_virgula >
23 < identifier, b_1 >
24 < assign >
25 < identifier, a >
26 < mult_mulop >
27 < identifier, a >
28 < ponto_virgula >
29 < write >
30 < abre_parent >
31 < literal, "O valor de b1 e:" >

```

```

32 < fecha_parent >
33 < ponto_virgula >
34 < write >
35 < abre_parent >
36 < identifier, b_1 >
37 < fecha_parent >
38 < ponto_virgula >
39 < identifier, b_2 >
40 < equal_relop >
41 < identifier, b >
42 < soma_addop >
43 < identifier, a >
44 < div_mulop >
45 < num, 2 >
46 < mult_mulop >
47 < abre_parent >
48 < identifier, a >
49 < soma_addop >
50 < num, 5 >
51 < fecha_parent >
52 < ponto_virgula >
53 < write >
54 < abre_parent >
55 < literal, "0 valor de b1 e:" >
56 < fecha_parent >
57 < ponto_virgula >
58 < write >
59 < abre_parent >
60 < identifier, b1 >
61 < fecha_parent >
62 < ponto_virgula >
63 < stop_program >
64
65
66
67 **** Inicio Parser ****
68 Error in: program; Error(1): Token não esperado:< identifier, a >

```

Listing 7.5: Saida para o Codigo de teste : Teste2.txt

No Exemplo acima, foi reportado um Token não esperado, pois se esperava o identificador init para inicio de qualquer programa.

7.2.1. Correção Do Teste 2

Foram feitas as devidas correções no código acima para adequar as regras da gramática da linguagem. Assim, obtemos o seguinte resultado.

Listing 7.6: Saída Correta para o Código de teste : Teste2.txt

7.3. TESTE 3

```

1 % Programa de Teste Calculo de idade %
2 INIT
3 cont is int;
4 media, idade, soma is integer;
5 begin
6 cont := 5;
7 soma := 0;
8 do
9 write("Altura:" );
10 read (altura);
11 soma := soma+altura;
12 cont := cont - 1;
13 while(cont > 0)
14 media := soma / qts
15 write("Media: ");
16 write (media);
17 STOP

```

Listing 7.7: Teste3.txt

Para o código acima o compilador obteve a seguinte saída :

```

1 **** Tokens lidos ****
2 < init_program >
3 < identifier, cont >
4 < is_decl >
5 < identifier, int >
6 < ponto_virgula >
7 < identifier, media >
8 < virgula >
9 < identifier, idade >
10 < virgula >
11 < identifier, soma >
12 < is_decl >
13 < integer_type >
14 < ponto_virgula >
15 < begin >
16 < identifier, cont >
17 < assign >
18 < num, 5 >
19 < ponto_virgula >

```

```
20 < identifier, soma >
21 < assign >
22 < num, 0 >
23 < ponto_virgula >
24 < do >
25 < write >
26 < abre_parent >
27 < literal, "Altura:" >
28 < fecha_parent >
29 < ponto_virgula >
30 < read >
31 < abre_parent >
32 < identifier, altura >
33 < fecha_parent >
34 < ponto_virgula >
35 < identifier, soma >
36 < assign >
37 < identifier, soma >
38 < soma_addop >
39 < identifier, altura >
40 < ponto_virgula >
41 < identifier, cont >
42 < assign >
43 < identifier, cont >
44 < menos_addop >
45 < num, 1 >
46 < ponto_virgula >
47 < while >
48 < abre_parent >
49 < identifier, cont >
50 < greater_than_relop >
51 < num, 0 >
52 < fecha_parent >
53 < identifier, media >
54 < assign >
55 < identifier, soma >
56 < div_mulop >
57 < identifier, qts >
58 < write >
59 < abre_parent >
60 < literal, "Media: " >
61 < fecha_parent >
62 < ponto_virgula >
63 < write >
64 < abre_parent >
65 < identifier, media >
66 < fecha_parent >
```

```

67 < ponto_virgula >
68 < stop_program >
69
70
71
72 **** Inicio Parser ****
73 Token Consumido(3): < init_program >
74 Token Consumido(4): < identifier, cont >
75 Token Consumido(4): < is_decl >
76 Error in: type; Error(4): Token n?ço esperado:< identifier, int >
77 Fim de arquivo inesperado.

```

Listing 7.8: Saída para o Código de teste : Teste3.txt

No exemplo acima, o programa reportou um erro pois foi definido um tipo de variável inexistente na gramática para essa linguagem. O tipo de variável `int` não foi modelado para essa gramática.

7.3.1. Correção Do Teste 3

Foram feitas as devidas correções no código acima para adequar as regras da gramática da linguagem. Assim, obtemos o seguinte resultado.

```

1 **** Tokens lidos ****
2 < init_program >
3 < identifier, cont >
4 < virgula >
5 < identifier, qts >
6 < is_decl >
7 < integer_type >
8 < ponto_virgula >
9 < identifier, media >
10 < virgula >
11 < identifier, idade >
12 < virgula >
13 < identifier, soma >
14 < is_decl >
15 < integer_type >
16 < ponto_virgula >
17 < begin >
18 < identifier, cont >
19 < assign >
20 < num, 5 >
21 < ponto_virgula >
22 < identifier, soma >
23 < assign >
24 < num, 0 >
25 < ponto_virgula >

```



```
26 < identifier, qts >
27 < assign >
28 < num, 1 >
29 < ponto_virgula >
30 < do >
31 < write >
32 < abre_parent >
33 < literal, "Altura:" >
34 < fecha_parent >
35 < ponto_virgula >
36 < read >
37 < abre_parent >
38 < identifier, altura >
39 < fecha_parent >
40 < ponto_virgula >
41 < identifier, soma >
42 < assign >
43 < identifier, soma >
44 < soma_addop >
45 < identifier, altura >
46 < ponto_virgula >
47 < identifier, cont >
48 < assign >
49 < identifier, cont >
50 < menos_addop >
51 < num, 1 >
52 < ponto_virgula >
53 < while >
54 < abre_parent >
55 < identifier, cont >
56 < greater_than_relop >
57 < num, 0 >
58 < fecha_parent >
59 < ponto_virgula >
60 < identifier, media >
61 < assign >
62 < identifier, soma >
63 < div_mulop >
64 < identifier, qts >
65 < ponto_virgula >
66 < write >
67 < abre_parent >
68 < literal, "Media: " >
69 < fecha_parent >
70 < ponto_virgula >
71 < write >
72 < abre_parent >
```

```

73 < identifier, media >
74 < fecha_parent >
75 < ponto_virgula >
76 < stop_program >
77
78
79
80 **** Inicio Parser ****
81 Token Consumido(3): < init_program >
82 Token Consumido(4): < identifier, cont >
83 Token Consumido(4): < virgula >
84 Token Consumido(4): < identifier, qts >
85 Token Consumido(4): < is_decl >
86 Token Consumido(4): < integer_type >
87 Token Consumido(4): < ponto_virgula >
88 Token Consumido(5): < identifier, media >
89 Token Consumido(5): < virgula >
90 Token Consumido(5): < identifier, idade >
91 Token Consumido(5): < virgula >
92 Token Consumido(5): < identifier, soma >
93 Token Consumido(5): < is_decl >
94 Token Consumido(5): < integer_type >
95 Token Consumido(5): < ponto_virgula >
96 Token Consumido(6): < begin >
97 Token Consumido(7): < identifier, cont >
98 Token Consumido(7): < assign >
99 Token Consumido(7): < num, 5 >
100 Token Consumido(7): < ponto_virgula >
101 Token Consumido(8): < identifier, soma >
102 Token Consumido(8): < assign >
103 Token Consumido(8): < num, 0 >
104 Token Consumido(8): < ponto_virgula >
105 Token Consumido(9): < identifier, qts >
106 Token Consumido(9): < assign >
107 Token Consumido(9): < num, 1 >
108 Token Consumido(9): < ponto_virgula >
109 Token Consumido(10): < do >
110 Token Consumido(11): < write >
111 Token Consumido(11): < abre_parent >
112 Token Consumido(11): < literal, "Altura:" >
113 Token Consumido(11): < fecha_parent >
114 Token Consumido(11): < ponto_virgula >
115 Token Consumido(12): < read >
116 Token Consumido(12): < abre_parent >
117 Token Consumido(12): < identifier, altura >
118 Token Consumido(12): < fecha_parent >
119 Token Consumido(12): < ponto_virgula >

```

```

120 Token Consumido(13): < identifier, soma >
121 Token Consumido(13): < assign >
122 Token Consumido(13): < identifier, soma >
123 Token Consumido(13): < soma_addop >
124 Token Consumido(13): < identifier, altura >
125 Token Consumido(13): < ponto_virgula >
126 Token Consumido(14): < identifier, cont >
127 Token Consumido(14): < assign >
128 Token Consumido(14): < identifier, cont >
129 Token Consumido(14): < menos_addop >
130 Token Consumido(14): < num, 1 >
131 Token Consumido(14): < ponto_virgula >
132 Token Consumido(15): < while >
133 Token Consumido(15): < abre_parent >
134 Token Consumido(15): < identifier, cont >
135 Token Consumido(15): < greater_than_relop >
136 Token Consumido(15): < num, 0 >
137 Token Consumido(15): < fecha_parent >
138 Token Consumido(15): < ponto_virgula >
139 Token Consumido(16): < identifier, media >
140 Token Consumido(16): < assign >
141 Token Consumido(16): < identifier, soma >
142 Token Consumido(16): < div_mulop >
143 Token Consumido(16): < identifier, qts >
144 Token Consumido(16): < ponto_virgula >
145 Token Consumido(17): < write >
146 Token Consumido(17): < abre_parent >
147 Token Consumido(17): < literal, "Media: " >
148 Token Consumido(17): < fecha_parent >
149 Token Consumido(17): < ponto_virgula >
150 Token Consumido(18): < write >
151 Token Consumido(18): < abre_parent >
152 Token Consumido(18): < identifier, media >
153 Token Consumido(18): < fecha_parent >
154 Token Consumido(18): < ponto_virgula >

```

Listing 7.9: Saida Correta para o Codigo de teste : Teste3.txt

7.4. TESTE 4

```

1 init
2 % Outro programa de teste
3 i, j, k, @total is integer;
4 nome is string
5 write("Digite o seu nome: ");
6 read(nome);

```

```

7 write("Digite um valor inteiro: ");
8 read (I);
9 k := i * (5-i * 50 / 10);
10 j := i * 10;
11 k := i* j / k;
12 k := 4 + a $;
13 write(nome);
14 write(", os números gerados sao: ");
15 write(i);
16 write(j);
17 write(k);

```

Listing 7.10: Teste4.txt

Para o código acima o compilador obteve a seguinte saída :

```

1 **** Tokens lidos ****
2 < init_program >
3 Error(2): coment?ario n?ço fechado
4
5
6
7 **** Inicio Parser ****
8 Token Consumido(1): < init_program >
9 Fim de arquivo inesperado.

```

Listing 7.11: Saída para o Código de teste : Teste4.txt

No código acima, não foi fechado o comentário, assim, erro para iniciar o programa, pois era esperado o % para fim de comentário.

7.4.1. Correção Do Teste 4

Foram feitas as devidas correções no código acima para adequar as regras da gramática da linguagem. Assim, obtemos o seguinte resultado.

```

1
2
3 **** Tokens lidos ****
4 < init_program >
5 < identifier, i >
6 < virgula >
7 < identifier, j >
8 < virgula >
9 < identifier, k >
10 < virgula >
11 Error na Linha(3): Token '@' Nao esperado
12 < identifier, total >
13 < is_decl >

```

```
14 < integer_type >
15 < ponto_virgula >
16 < identifier, nome >
17 < is_decl >
18 < string_type >
19 < ponto_virgula >
20 < begin >
21 < write >
22 < abre_parent >
23 < literal, "Digite o seu nome: " >
24 < fecha_parent >
25 < ponto_virgula >
26 < read >
27 < abre_parent >
28 < identifier, nome >
29 < fecha_parent >
30 < ponto_virgula >
31 < write >
32 < abre_parent >
33 < literal, "Digite um valor inteiro: " >
34 < fecha_parent >
35 < ponto_virgula >
36 < read >
37 < abre_parent >
38 < identifier, i >
39 < fecha_parent >
40 < ponto_virgula >
41 < identifier, k >
42 < assign >
43 < abre_parent >
44 < abre_parent >
45 < identifier, i >
46 < mult_mulop >
47 < num, 5 >
48 < fecha_parent >
49 < menos_addop >
50 < abre_parent >
51 < identifier, i >
52 < mult_mulop >
53 < num, 50 >
54 < fecha_parent >
55 < fecha_parent >
56 < div_mulop >
57 < num, 10 >
58 < ponto_virgula >
59 < identifier, j >
60 < assign >
```

```

61 < identifier, i >
62 < mult_mulop >
63 < num, 10 >
64 < ponto_virgula >
65 < identifier, k >
66 < assign >
67 < abre_parent >
68 < identifier, i >
69 < mult_mulop >
70 < identifier, j >
71 < fecha_parent >
72 < div_mulop >
73 < identifier, k >
74 < ponto_virgula >
75 < identifier, k >
76 < assign >
77 < num, 4 >
78 < soma_addop >
79 < identifier, a >
80 Error na Linha(13): Token '$' Nao esperado
81 < ponto_virgula >
82 < write >
83 < abre_parent >
84 < identifier, nome >
85 < fecha_parent >
86 < ponto_virgula >
87 < write >
88 < abre_parent >
89 < literal, ", os n?žmeros gerados sao: " >
90 < fecha_parent >
91 < ponto_virgula >
92 < write >
93 < abre_parent >
94 < identifier, i >
95 < fecha_parent >
96 < ponto_virgula >
97 < write >
98 < abre_parent >
99 < identifier, j >
100 < fecha_parent >
101 < ponto_virgula >
102 < write >
103 < abre_parent >
104 < identifier, k >
105 < fecha_parent >
106 < ponto_virgula >
107 < stop_program >

```

```

108
109
110
111 **** Inicio Parser ****
112 Token Consumido(1): < init_program >
113 Token Consumido(3): < identifier, i >
114 Token Consumido(3): < virgula >
115 Token Consumido(3): < identifier, j >
116 Token Consumido(3): < virgula >
117 Token Consumido(3): < identifier, k >
118 Token Consumido(3): < virgula >
119 Token Consumido(3): < identifier, total >
120 Token Consumido(3): < is_decl >
121 Token Consumido(3): < integer_type >
122 Token Consumido(3): < ponto_virgula >
123 Token Consumido(4): < identifier, nome >
124 Token Consumido(4): < is_decl >
125 Token Consumido(4): < string_type >
126 Token Consumido(4): < ponto_virgula >
127 Token Consumido(5): < begin >
128 Token Consumido(6): < write >
129 Token Consumido(6): < abre_parent >
130 Token Consumido(6): < literal, "Digite o seu nome: " >
131 Token Consumido(6): < fecha_parent >
132 Token Consumido(6): < ponto_virgula >
133 Token Consumido(7): < read >
134 Token Consumido(7): < abre_parent >
135 Token Consumido(7): < identifier, nome >
136 Token Consumido(7): < fecha_parent >
137 Token Consumido(7): < ponto_virgula >
138 Token Consumido(8): < write >
139 Token Consumido(8): < abre_parent >
140 Token Consumido(8): < literal, "Digite um valor inteiro: " >
141 Token Consumido(8): < fecha_parent >
142 Token Consumido(8): < ponto_virgula >
143 Token Consumido(9): < read >
144 Token Consumido(9): < abre_parent >
145 Token Consumido(9): < identifier, i >
146 Token Consumido(9): < fecha_parent >
147 Token Consumido(9): < ponto_virgula >
148 Token Consumido(10): < identifier, k >
149 Token Consumido(10): < assign >
150 Token Consumido(10): < abre_parent >
151 Token Consumido(10): < abre_parent >
152 Token Consumido(10): < identifier, i >
153 Token Consumido(10): < mult_mulop >
154 Token Consumido(10): < num, 5 >

```

```

155 Token Consumido(10): < fecha_parent >
156 Token Consumido(10): < menos_addop >
157 Token Consumido(10): < abre_parent >
158 Token Consumido(10): < identifier, i >
159 Token Consumido(10): < mult_mulop >
160 Token Consumido(10): < num, 50 >
161 Token Consumido(10): < fecha_parent >
162 Token Consumido(10): < fecha_parent >
163 Token Consumido(10): < div_mulop >
164 Token Consumido(10): < num, 10 >
165 Token Consumido(10): < ponto_virgula >
166 Token Consumido(11): < identifier, j >
167 Token Consumido(11): < assign >
168 Token Consumido(11): < identifier, i >
169 Token Consumido(11): < mult_mulop >
170 Token Consumido(11): < num, 10 >
171 Token Consumido(11): < ponto_virgula >
172 Token Consumido(12): < identifier, k >
173 Token Consumido(12): < assign >
174 Token Consumido(12): < abre_parent >
175 Token Consumido(12): < identifier, i >
176 Token Consumido(12): < mult_mulop >
177 Token Consumido(12): < identifier, j >
178 Token Consumido(12): < fecha_parent >
179 Token Consumido(12): < div_mulop >
180 Token Consumido(12): < identifier, k >
181 Token Consumido(12): < ponto_virgula >
182 Token Consumido(13): < identifier, k >
183 Token Consumido(13): < assign >
184 Token Consumido(13): < num, 4 >
185 Token Consumido(13): < soma_addop >
186 Token Consumido(13): < identifier, a >
187 Token Consumido(13): < ponto_virgula >
188 Token Consumido(14): < write >
189 Token Consumido(14): < abre_parent >
190 Token Consumido(14): < identifier, nome >
191 Token Consumido(14): < fecha_parent >
192 Token Consumido(14): < ponto_virgula >
193 Token Consumido(15): < write >
194 Token Consumido(15): < abre_parent >
195 Token Consumido(15): < literal, ", os n?žmeros gerados sao: " >
196 Token Consumido(15): < fecha_parent >
197 Token Consumido(15): < ponto_virgula >
198 Token Consumido(16): < write >
199 Token Consumido(16): < abre_parent >
200 Token Consumido(16): < identifier, i >
201 Token Consumido(16): < fecha_parent >

```



```

202 Token Consumido(16): < ponto_virgula >
203 Token Consumido(17): < write >
204 Token Consumido(17): < abre_parent >
205 Token Consumido(17): < identifier, j >
206 Token Consumido(17): < fecha_parent >
207 Token Consumido(17): < ponto_virgula >
208 Token Consumido(18): < write >
209 Token Consumido(18): < abre_parent >
210 Token Consumido(18): < identifier, k >
211 Token Consumido(18): < fecha_parent >
212 Token Consumido(18): < ponto_virgula >

```

Listing 7.12: Saida Correta para oCodigo de teste : Teste4.txt

7.5. TESTE 5

```

1  init
2  i, j, k, @total is integer;
3  nome is string
4  write("Digite o seu nome: ");
5  read(nome);
6  write("Digite um valor inteiro: ");
7  read (I);
8  k := i * (5-i * 50 / 10;
9  j := i * 10;
10 k := i* j / k;
11 k := 4 + a $;
12 write(nome);
13 write(", os números gerados sao: ");
14 write(i);
15 write(j);
16 write(k);

```

Listing 7.13: Teste5.txt

Para o codigo acima o compilador obteve a seguinte saida :

```

1
2 **** Tokens lidos ****
3 < init_program >
4 < identifier, i >
5 < virgula >
6 < identifier, j >
7 < virgula >
8 < identifier, k >
9 < virgula >
10 Error na Linha(2): Token '@' Nao esperado

```

```
11 < identifier, total >
12 < is_decl >
13 < integer_type >
14 < ponto_virgula >
15 < identifier, nome >
16 < is_decl >
17 < string_type >
18 < write >
19 < abre_parent >
20 < literal, "Digite o seu nome: " >
21 < fecha_parent >
22 < ponto_virgula >
23 < read >
24 < abre_parent >
25 < identifier, nome >
26 < fecha_parent >
27 < ponto_virgula >
28 < write >
29 < abre_parent >
30 Error na Linha(6): Token '"' Nao esperado
31 < read >
32 < abre_parent >
33 < identifier, i >
34 < fecha_parent >
35 < ponto_virgula >
36 < identifier, k >
37 < assign >
38 < identifier, i >
39 < mult_mulop >
40 < abre_parent >
41 < num, 5 >
42 < menos_addop >
43 < identifier, i >
44 < mult_mulop >
45 < num, 50 >
46 < div_mulop >
47 < num, 10 >
48 < ponto_virgula >
49 < identifier, j >
50 < assign >
51 < identifier, i >
52 < mult_mulop >
53 < num, 10 >
54 < ponto_virgula >
55 < identifier, k >
56 < assign >
57 < identifier, i >
```

```

58 < mult_mulop >
59 < identifier, j >
60 < div_mulop >
61 < identifier, k >
62 < ponto_virgula >
63 < identifier, k >
64 < assign >
65 < num, 4 >
66 < soma_addop >
67 < identifier, a >
68 Error na Linha(10): Token '$' Nao esperado
69 < ponto_virgula >
70 < write >
71 < abre_parent >
72 < identifier, nome >
73 < fecha_parent >
74 < ponto_virgula >
75 < write >
76 < abre_parent >
77 < literal, ", os n?žmeros gerados sao: " >
78 < fecha_parent >
79 < ponto_virgula >
80 < write >
81 < abre_parent >
82 < identifier, i >
83 < fecha_parent >
84 < ponto_virgula >
85 < write >
86 < abre_parent >
87 < identifier, j >
88 < fecha_parent >
89 < ponto_virgula >
90 < write >
91 < abre_parent >
92 < identifier, k >
93 < fecha_parent >
94 < ponto_virgula >
95
96
97
98 **** Inicio Parser ****
99 Token Consumido(1): < init_program >
100 Token Consumido(2): < identifier, i >
101 Token Consumido(2): < virgula >
102 Token Consumido(2): < identifier, j >
103 Token Consumido(2): < virgula >
104 Token Consumido(2): < identifier, k >

```

```

105 Token Consumido(2): < virgula >
106 Token Consumido(2): < identifier, total >
107 Token Consumido(2): < is_decl >
108 Token Consumido(2): < integer_type >
109 Token Consumido(2): < ponto_virgula >
110 Token Consumido(3): < identifier, nome >
111 Token Consumido(3): < is_decl >
112 Token Consumido(3): < string_type >
113 Error in: declList; Error(4): Token n?ço esperado:< write >
114 Fim de arquivo inesperado.

```

Listing 7.14: Saida para oCodigo de teste : Teste5.txt

No Código acima, não foi declarado o begin. Assim o comando write foi dado como inesperado, pois era esperado o identificador begin.

7.5.1. Correção Do Teste 5

Foram feitas as devidas correções no código acima para adequar as regras da gramática da linguagem. Assim, obtemos o seguinte resultado.

```

1
2
3 **** Tokens lidos ****
4 < init_program >
5 < identifier, i >
6 < virgula >
7 < identifier, j >
8 < virgula >
9 < identifier, k >
10 < virgula >
11 Error na Linha(3): Token '@' Nao esperado
12 < identifier, total >
13 < is_decl >
14 < integer_type >
15 < ponto_virgula >
16 < identifier, nome >
17 < is_decl >
18 < string_type >
19 < ponto_virgula >
20 < begin >
21 < write >
22 < abre_parent >
23 < literal, "Digite o seu nome: " >
24 < fecha_parent >
25 < ponto_virgula >
26 < read >
27 < abre_parent >

```

```
28 < identifier, nome >
29 < fecha_parent >
30 < ponto_virgula >
31 < write >
32 < abre_parent >
33 < literal, "Digite um valor inteiro: " >
34 < fecha_parent >
35 < ponto_virgula >
36 < read >
37 < abre_parent >
38 < identifier, i >
39 < fecha_parent >
40 < ponto_virgula >
41 < identifier, k >
42 < assign >
43 < abre_parent >
44 < abre_parent >
45 < identifier, i >
46 < mult_mulop >
47 < num, 5 >
48 < fecha_parent >
49 < menos_addop >
50 < abre_parent >
51 < identifier, i >
52 < mult_mulop >
53 < num, 50 >
54 < fecha_parent >
55 < fecha_parent >
56 < div_mulop >
57 < num, 10 >
58 < ponto_virgula >
59 < identifier, j >
60 < assign >
61 < identifier, i >
62 < mult_mulop >
63 < num, 10 >
64 < ponto_virgula >
65 < identifier, k >
66 < assign >
67 < abre_parent >
68 < identifier, i >
69 < mult_mulop >
70 < identifier, j >
71 < fecha_parent >
72 < div_mulop >
73 < identifier, k >
74 < ponto_virgula >
```

```

75 < identifier, k >
76 < assign >
77 < num, 4 >
78 < soma_addop >
79 < identifier, a >
80 Error na Linha(13): Token '$' Nao esperado
81 < ponto_virgula >
82 < write >
83 < abre_parent >
84 < identifier, nome >
85 < fecha_parent >
86 < ponto_virgula >
87 < write >
88 < abre_parent >
89 < literal, ", os n?žmeros gerados sao: " >
90 < fecha_parent >
91 < ponto_virgula >
92 < write >
93 < abre_parent >
94 < identifier, i >
95 < fecha_parent >
96 < ponto_virgula >
97 < write >
98 < abre_parent >
99 < identifier, j >
100 < fecha_parent >
101 < ponto_virgula >
102 < write >
103 < abre_parent >
104 < identifier, k >
105 < fecha_parent >
106 < ponto_virgula >
107 < stop_program >
108
109
110
111 **** Inicio Parser ****
112 Token Consumido(1): < init_program >
113 Token Consumido(3): < identifier, i >
114 Token Consumido(3): < virgula >
115 Token Consumido(3): < identifier, j >
116 Token Consumido(3): < virgula >
117 Token Consumido(3): < identifier, k >
118 Token Consumido(3): < virgula >
119 Token Consumido(3): < identifier, total >
120 Token Consumido(3): < is_decl >
121 Token Consumido(3): < integer_type >

```

```

122 Token Consumido(3): < ponto_virgula >
123 Token Consumido(4): < identifier, nome >
124 Token Consumido(4): < is_decl >
125 Token Consumido(4): < string_type >
126 Token Consumido(4): < ponto_virgula >
127 Token Consumido(5): < begin >
128 Token Consumido(6): < write >
129 Token Consumido(6): < abre_parent >
130 Token Consumido(6): < literal, "Digite o seu nome: " >
131 Token Consumido(6): < fecha_parent >
132 Token Consumido(6): < ponto_virgula >
133 Token Consumido(7): < read >
134 Token Consumido(7): < abre_parent >
135 Token Consumido(7): < identifier, nome >
136 Token Consumido(7): < fecha_parent >
137 Token Consumido(7): < ponto_virgula >
138 Token Consumido(8): < write >
139 Token Consumido(8): < abre_parent >
140 Token Consumido(8): < literal, "Digite um valor inteiro: " >
141 Token Consumido(8): < fecha_parent >
142 Token Consumido(8): < ponto_virgula >
143 Token Consumido(9): < read >
144 Token Consumido(9): < abre_parent >
145 Token Consumido(9): < identifier, i >
146 Token Consumido(9): < fecha_parent >
147 Token Consumido(9): < ponto_virgula >
148 Token Consumido(10): < identifier, k >
149 Token Consumido(10): < assign >
150 Token Consumido(10): < abre_parent >
151 Token Consumido(10): < abre_parent >
152 Token Consumido(10): < identifier, i >
153 Token Consumido(10): < mult_mulop >
154 Token Consumido(10): < num, 5 >
155 Token Consumido(10): < fecha_parent >
156 Token Consumido(10): < menos_addop >
157 Token Consumido(10): < abre_parent >
158 Token Consumido(10): < identifier, i >
159 Token Consumido(10): < mult_mulop >
160 Token Consumido(10): < num, 50 >
161 Token Consumido(10): < fecha_parent >
162 Token Consumido(10): < fecha_parent >
163 Token Consumido(10): < div_mulop >
164 Token Consumido(10): < num, 10 >
165 Token Consumido(10): < ponto_virgula >
166 Token Consumido(11): < identifier, j >
167 Token Consumido(11): < assign >
168 Token Consumido(11): < identifier, i >

```

```

169 Token Consumido(11): < mult_mulop >
170 Token Consumido(11): < num, 10 >
171 Token Consumido(11): < ponto_virgula >
172 Token Consumido(12): < identifier, k >
173 Token Consumido(12): < assign >
174 Token Consumido(12): < abre_parent >
175 Token Consumido(12): < identifier, i >
176 Token Consumido(12): < mult_mulop >
177 Token Consumido(12): < identifier, j >
178 Token Consumido(12): < fecha_parent >
179 Token Consumido(12): < div_mulop >
180 Token Consumido(12): < identifier, k >
181 Token Consumido(12): < ponto_virgula >
182 Token Consumido(13): < identifier, k >
183 Token Consumido(13): < assign >
184 Token Consumido(13): < num, 4 >
185 Token Consumido(13): < soma_addop >
186 Token Consumido(13): < identifier, a >
187 Token Consumido(13): < ponto_virgula >
188 Token Consumido(14): < write >
189 Token Consumido(14): < abre_parent >
190 Token Consumido(14): < identifier, nome >
191 Token Consumido(14): < fecha_parent >
192 Token Consumido(14): < ponto_virgula >
193 Token Consumido(15): < write >
194 Token Consumido(15): < abre_parent >
195 Token Consumido(15): < literal, ", os n?žmeros gerados sao: " >
196 Token Consumido(15): < fecha_parent >
197 Token Consumido(15): < ponto_virgula >
198 Token Consumido(16): < write >
199 Token Consumido(16): < abre_parent >
200 Token Consumido(16): < identifier, i >
201 Token Consumido(16): < fecha_parent >
202 Token Consumido(16): < ponto_virgula >
203 Token Consumido(17): < write >
204 Token Consumido(17): < abre_parent >
205 Token Consumido(17): < identifier, j >
206 Token Consumido(17): < fecha_parent >
207 Token Consumido(17): < ponto_virgula >
208 Token Consumido(18): < write >
209 Token Consumido(18): < abre_parent >
210 Token Consumido(18): < identifier, k >
211 Token Consumido(18): < fecha_parent >
212 Token Consumido(18): < ponto_virgula >

```

Listing 7.15: Saida Correta para o Codigo de teste : Teste5.txt

7.6. TESTE 6

```

1  init
2  a, b, c, maior is integer;
3  write("Digite uma idade: ");
4  read(a);
5  write("Digite outra idade: ");
6  read(b);
7  write("Digite mais uma idade: ");
8  read(c);
9  maior := 0;
10 if ( a>b and a>c )
11 maior := a;
12 else
13 if (b>c)
14 maior := b;
15 else
16 maior := c;
17 write("Maior idade: ");
18 write(maior);
19 end

```

Listing 7.16: Teste6.txt

Para o código acima o compilador obteve a seguinte saída :

```

1  **** Tokens lidos ****
2  < init_program >
3  < identifier, a >
4  < virgula >
5  < identifier, b >
6  < virgula >
7  < identifier, c >
8  < virgula >
9  < identifier, maior >
10 < is_decl >
11 < integer_type >
12 < ponto_virgula >
13 < write >
14 < abre_parent >
15 < literal, "Digite uma idade: " >
16 < fecha_parent >
17 < ponto_virgula >
18 < read >
19 < abre_parent >
20 < identifier, a >
21 < fecha_parent >
22 < ponto_virgula >

```

```
23 < write >
24 < abre_parent >
25 < literal, "Digite outra idade: " >
26 < fecha_parent >
27 < ponto_virgula >
28 < read >
29 < abre_parent >
30 < identifiier, b >
31 < fecha_parent >
32 < ponto_virgula >
33 < write >
34 < abre_parent >
35 < literal, "Digite mais uma idade: " >
36 < fecha_parent >
37 < ponto_virgula >
38 < read >
39 < abre_parent >
40 < identifiier, c >
41 < ponto_virgula >
42 < identifiier, maior >
43 < assign >
44 < num, 0 >
45 < ponto_virgula >
46 < if >
47 < abre_parent >
48 < identifiier, a >
49 < greater_than_relop >
50 < identifiier, b >
51 < and_mulop >
52 < identifiier, a >
53 < greater_than_relop >
54 < identifiier, c >
55 < fecha_parent >
56 < identifiier, maior >
57 < assign >
58 < identifiier, a >
59 < ponto_virgula >
60 < else >
61 < if >
62 < abre_parent >
63 < identifiier, b >
64 < greater_than_relop >
65 < identifiier, c >
66 < fecha_parent >
67 < identifiier, maior >
68 < assign >
69 < identifiier, b >
```

```

70 < ponto_virgula >
71 < else >
72 < identifier, maior >
73 < assign >
74 < identifier, c >
75 < ponto_virgula >
76 < write >
77 < abre_parent >
78 < literal, "Maior idade: " >
79 < fecha_parent >
80 < ponto_virgula >
81 < write >
82 < abre_parent >
83 < identifier, maior >
84 < fecha_parent >
85 < ponto_virgula >
86 < end >
87
88
89
90 **** Inicio Parser ****
91 Token Consumido(1): < init_program >
92 Token Consumido(2): < identifier, a >
93 Token Consumido(2): < virgula >
94 Token Consumido(2): < identifier, b >
95 Token Consumido(2): < virgula >
96 Token Consumido(2): < identifier, c >
97 Token Consumido(2): < virgula >
98 Token Consumido(2): < identifier, maior >
99 Token Consumido(2): < is_decl >
100 Token Consumido(2): < integer_type >
101 Token Consumido(2): < ponto_virgula >
102 Error in: identList; Error(3): Token n?ço esperado:< write >
103 Fim de arquivo inesperado.

```

Listing 7.17: Saída para o Código de teste : Teste6.txt

No Código acima, não foi declarado o begin. Assim o comando write foi dado como inesperado, pois era esperado o identificador begin.

7.6.1. Correção Do Teste 6

Foram feitas as devidas correções no código acima para adequar as regras da gramática da linguagem. Assim, obtemos o seguinte resultado.

```

1
2
3 **** Tokens lidos ****

```

```
4 < init_program >
5 < identifier, a >
6 < virgula >
7 < identifier, b >
8 < virgula >
9 < identifier, c >
10 < virgula >
11 < identifier, maior >
12 < is_decl >
13 < integer_type >
14 < ponto_virgula >
15 < begin >
16 < write >
17 < abre_parent >
18 < literal, "Digite uma idade: " >
19 < fecha_parent >
20 < ponto_virgula >
21 < read >
22 < abre_parent >
23 < identifier, a >
24 < fecha_parent >
25 < ponto_virgula >
26 < write >
27 < abre_parent >
28 < literal, "Digite outra idade: " >
29 < fecha_parent >
30 < ponto_virgula >
31 < read >
32 < abre_parent >
33 < identifier, b >
34 < fecha_parent >
35 < ponto_virgula >
36 < write >
37 < abre_parent >
38 < literal, "Digite mais uma idade: " >
39 < fecha_parent >
40 < ponto_virgula >
41 < read >
42 < abre_parent >
43 < identifier, c >
44 < fecha_parent >
45 < ponto_virgula >
46 < identifier, maior >
47 < assign >
48 < num, 0 >
49 < ponto_virgula >
50 < if >
```

```
51 < abre_parent >
52 < abre_parent >
53 < identifier, a >
54 < greater_than_relop >
55 < identifier, b >
56 < fecha_parent >
57 < and_mulop >
58 < abre_parent >
59 < identifier, a >
60 < greater_than_relop >
61 < identifier, c >
62 < fecha_parent >
63 < fecha_parent >
64 < begin >
65 < identifier, maior >
66 < assign >
67 < identifier, a >
68 < ponto_virgula >
69 < end >
70 < else >
71 < begin >
72 < if >
73 < abre_parent >
74 < identifier, b >
75 < greater_than_relop >
76 < identifier, c >
77 < fecha_parent >
78 < begin >
79 < identifier, maior >
80 < assign >
81 < identifier, b >
82 < ponto_virgula >
83 < end >
84 < else >
85 < begin >
86 < identifier, maior >
87 < assign >
88 < identifier, c >
89 < ponto_virgula >
90 < end >
91 < end >
92 < write >
93 < abre_parent >
94 < literal, "Maior idade: " >
95 < fecha_parent >
96 < ponto_virgula >
97 < write >
```

```

98 < abre_parent >
99 < identifier, maior >
100 < fecha_parent >
101 < ponto_virgula >
102 < stop_program >
103
104
105
106 **** Inicio Parser ****
107 Token Consumido(1): < init_program >
108 Token Consumido(2): < identifier, a >
109 Token Consumido(2): < virgula >
110 Token Consumido(2): < identifier, b >
111 Token Consumido(2): < virgula >
112 Token Consumido(2): < identifier, c >
113 Token Consumido(2): < virgula >
114 Token Consumido(2): < identifier, maior >
115 Token Consumido(2): < is_decl >
116 Token Consumido(2): < integer_type >
117 Token Consumido(2): < ponto_virgula >
118 Token Consumido(3): < begin >
119 Token Consumido(4): < write >
120 Token Consumido(4): < abre_parent >
121 Token Consumido(4): < literal, "Digite uma idade: " >
122 Token Consumido(4): < fecha_parent >
123 Token Consumido(4): < ponto_virgula >
124 Token Consumido(5): < read >
125 Token Consumido(5): < abre_parent >
126 Token Consumido(5): < identifier, a >
127 Token Consumido(5): < fecha_parent >
128 Token Consumido(5): < ponto_virgula >
129 Token Consumido(6): < write >
130 Token Consumido(6): < abre_parent >
131 Token Consumido(6): < literal, "Digite outra idade: " >
132 Token Consumido(6): < fecha_parent >
133 Token Consumido(6): < ponto_virgula >
134 Token Consumido(7): < read >
135 Token Consumido(7): < abre_parent >
136 Token Consumido(7): < identifier, b >
137 Token Consumido(7): < fecha_parent >
138 Token Consumido(7): < ponto_virgula >
139 Token Consumido(8): < write >
140 Token Consumido(8): < abre_parent >
141 Token Consumido(8): < literal, "Digite mais uma idade: " >
142 Token Consumido(8): < fecha_parent >
143 Token Consumido(8): < ponto_virgula >
144 Token Consumido(9): < read >

```

```

145 Token Consumido(9): < abre_parent >
146 Token Consumido(9): < identifier, c >
147 Token Consumido(9): < fecha_parent >
148 Token Consumido(9): < ponto_virgula >
149 Token Consumido(10): < identifier, maior >
150 Token Consumido(10): < assign >
151 Token Consumido(10): < num, 0 >
152 Token Consumido(10): < ponto_virgula >
153 Token Consumido(11): < if >
154 Token Consumido(11): < abre_parent >
155 Token Consumido(11): < abre_parent >
156 Token Consumido(11): < identifier, a >
157 Token Consumido(11): < greater_than_relop >
158 Token Consumido(11): < identifier, b >
159 Token Consumido(11): < fecha_parent >
160 Token Consumido(11): < and_mulop >
161 Token Consumido(11): < abre_parent >
162 Token Consumido(11): < identifier, a >
163 Token Consumido(11): < greater_than_relop >
164 Token Consumido(11): < identifier, c >
165 Token Consumido(11): < fecha_parent >
166 Token Consumido(11): < fecha_parent >
167 Token Consumido(11): < begin >
168 Token Consumido(12): < identifier, maior >
169 Token Consumido(12): < assign >
170 Token Consumido(12): < identifier, a >
171 Token Consumido(12): < ponto_virgula >
172 Token Consumido(13): < end >
173 Token Consumido(13): < else >
174 Token Consumido(13): < begin >
175 Token Consumido(14): < if >
176 Token Consumido(14): < abre_parent >
177 Token Consumido(14): < identifier, b >
178 Token Consumido(14): < greater_than_relop >
179 Token Consumido(14): < identifier, c >
180 Token Consumido(14): < fecha_parent >
181 Token Consumido(14): < begin >
182 Token Consumido(15): < identifier, maior >
183 Token Consumido(15): < assign >
184 Token Consumido(15): < identifier, b >
185 Token Consumido(15): < ponto_virgula >
186 Token Consumido(16): < end >
187 Token Consumido(16): < else >
188 Token Consumido(16): < begin >
189 Token Consumido(17): < identifier, maior >
190 Token Consumido(17): < assign >
191 Token Consumido(17): < identifier, c >

```

```

192 Token Consumido(17): < ponto_virgula >
193 Token Consumido(18): < end >
194 Token Consumido(19): < end >
195 Token Consumido(20): < write >
196 Token Consumido(20): < abre_parent >
197 Token Consumido(20): < literal, "Maior idade: " >
198 Token Consumido(20): < fecha_parent >
199 Token Consumido(20): < ponto_virgula >
200 Token Consumido(21): < write >
201 Token Consumido(21): < abre_parent >
202 Token Consumido(21): < identifier, maior >
203 Token Consumido(21): < fecha_parent >
204 Token Consumido(21): < ponto_virgula >

```

Listing 7.18: Saida Correta para o Codigo de teste : Teste6.txt

7.7. TESTE 7

```

1 % Programa de Teste Calculo de idade %
2 init
3   cont_ is integer;
4   media, idade, soma_ is integer;
5 begin
6   cont_ := 5;
7   soma := 0;
8   do
9     write(Altura: );
10    read (altura);
11    soma := soma + altura;
12    cont_ := cont_ - 1;
13  while(cont_ > 0);
14  write(Media: );
15  write (soma / qtd);
16 stop

```

Listing 7.19: Teste7.txt

Para o codigo acima o compilador obteve a seguinte saida :

```

1 **** Tokens lidos ****
2 < init_program >
3 < identifier, cont_ >
4 < is_decl >
5 < integer_type >
6 < ponto_virgula >
7 < identifier, media >
8 < virgula >

```



```

9 < identifier, idade >
10 < virgula >
11 < identifier, soma_ >
12 < is_decl >
13 < integer_type >
14 < ponto_virgula >
15 < begin >
16 < identifier, cont_ >
17 < assign >
18 < num, 5 >
19 < ponto_virgula >
20 < identifier, soma >
21 < assign >
22 < num, 0 >
23 < ponto_virgula >
24 < do >
25 < write >
26 < abre_parent >
27 Error na Linha(9): Token 'â' Nao esperado
28 Error na Linha(9): Token '?' Nao esperado
29 Error na Linha(9): Token '?' Nao esperado
30 < identifier, Altura >
31 Error na Linha(9): Token ':' Nao esperado
32 Error na Linha(9): Token '?' Nao esperado
33 Error na Linha(9): Token '?' Nao esperado
34 < fecha_parent >
35 < ponto_virgula >
36 < read >
37 < abre_parent >
38 < identifier, altura >
39 < fecha_parent >
40 < ponto_virgula >
41 < identifier, soma >
42 < assign >
43 < identifier, soma >
44 < soma_addop >
45 < identifier, altura >
46 < ponto_virgula >
47 < identifier, cont_ >
48 < assign >
49 < identifier, cont_ >
50 < menos_addop >
51 < num, 1 >
52 < ponto_virgula >
53 < while >
54 < abre_parent >
55 < identifier, cont_ >

```

```

56 < greater_than_relop >
57 < num, 0 >
58 < fecha_parent >
59 < ponto_virgula >
60 < write >
61 < abre_parent >
62 Error na Linha(14): Token 'â' Nao esperado
63 Error na Linha(14): Token '?' Nao esperado
64 Error na Linha(14): Token '?' Nao esperado
65 < identifier, media >
66 Error na Linha(14): Token ':' Nao esperado
67 Error na Linha(14): Token 'â' Nao esperado
68 Error na Linha(14): Token '?' Nao esperado
69 Error na Linha(14): Token '?' Nao esperado
70 < fecha_parent >
71 < ponto_virgula >
72 < write >
73 < abre_parent >
74 < identifier, soma >
75 < div_mulop >
76 < identifier, qtd >
77 < fecha_parent >
78 < ponto_virgula >
79 < stop_program >
80
81
82
83 **** Inicio Parser ****
84 Token Consumido(2): < init_program >
85 Token Consumido(3): < identifier, cont_ >
86 Token Consumido(3): < is_decl >
87 Token Consumido(3): < integer_type >
88 Token Consumido(3): < ponto_virgula >
89 Token Consumido(4): < identifier, media >
90 Token Consumido(4): < virgula >
91 Token Consumido(4): < identifier, idade >
92 Token Consumido(4): < virgula >
93 Token Consumido(4): < identifier, soma_ >
94 Token Consumido(4): < is_decl >
95 Token Consumido(4): < integer_type >
96 Token Consumido(4): < ponto_virgula >
97 Token Consumido(5): < begin >
98 Token Consumido(6): < identifier, cont_ >
99 Token Consumido(6): < assign >
100 Token Consumido(6): < num, 5 >
101 Token Consumido(6): < ponto_virgula >
102 Token Consumido(7): < identifier, soma >

```

```

103 Token Consumido(7): < assign >
104 Token Consumido(7): < num, 0 >
105 Token Consumido(7): < ponto_virgula >
106 Token Consumido(8): < do >
107 Token Consumido(9): < write >
108 Token Consumido(9): < abre_parent >
109 Token Consumido(9): < identifier, Altura >
110 Token Consumido(9): < fecha_parent >
111 Token Consumido(9): < ponto_virgula >
112 Token Consumido(10): < read >
113 Token Consumido(10): < abre_parent >
114 Token Consumido(10): < identifier, altura >
115 Token Consumido(10): < fecha_parent >
116 Token Consumido(10): < ponto_virgula >
117 Token Consumido(11): < identifier, soma >
118 Token Consumido(11): < assign >
119 Token Consumido(11): < identifier, soma >
120 Token Consumido(11): < soma_addop >
121 Token Consumido(11): < identifier, altura >
122 Token Consumido(11): < ponto_virgula >
123 Token Consumido(12): < identifier, cont_ >
124 Token Consumido(12): < assign >
125 Token Consumido(12): < identifier, cont_ >
126 Token Consumido(12): < menos_addop >
127 Token Consumido(12): < num, 1 >
128 Token Consumido(12): < ponto_virgula >
129 Token Consumido(13): < while >
130 Token Consumido(13): < abre_parent >
131 Token Consumido(13): < identifier, cont_ >
132 Token Consumido(13): < greater_than_relop >
133 Token Consumido(13): < num, 0 >
134 Token Consumido(13): < fecha_parent >
135 Token Consumido(13): < ponto_virgula >
136 Token Consumido(14): < write >
137 Token Consumido(14): < abre_parent >
138 Token Consumido(14): < identifier, media >
139 Token Consumido(14): < fecha_parent >
140 Token Consumido(14): < ponto_virgula >
141 Token Consumido(15): < write >
142 Token Consumido(15): < abre_parent >
143 Token Consumido(15): < identifier, soma >
144 Token Consumido(15): < div_mulop >
145 Token Consumido(15): < identifier, qtd >
146 Token Consumido(15): < fecha_parent >
147 Token Consumido(15): < ponto_virgula >

```

Listing 7.20: Saida para o Codigo de teste : Teste.txt

O Código acima esta escrito conforme a gramatica da linguagem . Não foi reportado nenhum erro.

7.8. TESTE 8

```

1  init
2  a, b is integer;
3  begin
4  if ( a > b) begin
5  write ("O maior numero e: ");
6  write (a);
7  end else begin
8  write ("O menor numero e: ");
9  write (b);
10 end
11 stop

```

Listing 7.21: Teste8.txt

Para o codigo acima o compilador obteve a seguinte saida :

```

1  **** Tokens lidos ****
2  < init_program >
3  < identifier, a >
4  < virgula >
5  < identifier, b >
6  < is_decl >
7  < integer_type >
8  < ponto_virgula >
9  < begin >
10 < read >
11 < abre_parent >
12 < identifier, a >
13 < fecha_parent >
14 < ponto_virgula >
15 < read >
16 < abre_parent >
17 < identifier, b >
18 < fecha_parent >
19 < ponto_virgula >
20 < end >
21 < stop_program >
22
23
24
25 **** Inicio Parser ****
26 Token Consumido(1): < init_program >

```

```
27 Token Consumido(2): < identifier, a >  
28 Token Consumido(2): < virgula >  
29 Token Consumido(2): < identifier, b >  
30 Token Consumido(2): < is_decl >  
31 Token Consumido(2): < integer_type >  
32 Token Consumido(2): < ponto_virgula >  
33 Token Consumido(3): < begin >  
34 Token Consumido(4): < read >  
35 Token Consumido(4): < abre_parent >  
36 Token Consumido(4): < identifier, a >  
37 Token Consumido(4): < fecha_parent >  
38 Token Consumido(4): < ponto_virgula >  
39 Token Consumido(5): < read >  
40 Token Consumido(5): < abre_parent >  
41 Token Consumido(5): < identifier, b >  
42 Token Consumido(5): < fecha_parent >  
43 Token Consumido(5): < ponto_virgula >
```

Listing 7.22: Saída para o Código de teste : Teste8.txt

O Código acima foi criado seguindo as regras de derivação da gramática da linguagem. Não foram encontrados erros.

8. METODOLOGIA

Como dito anteriormente, este trabalho está sendo desenvolvido em consonância com os conceitos ensinados pela orientadora e pelo Autor Aho, pelo livro : Compiladores, Técnicas e ferramentas.

```

1 package analisador_sintatico;
2
3 import analisador_lexico.Tag;
4 import analisador_lexico.Token;
5 import java.util.ArrayList;
6 import java.util.Hashtable;
7
8 public class Parser {
9     private Token tok;
10    private int tag;
11    private int i;
12    private int line;
13    private int curType;
14    private int resultExprType;
15    private ArrayList<Token> tokens = new ArrayList<Token> ();
16
17    public Parser(ArrayList<Token> tokens){
18        this.tokens=tokens;
19        i=0;
20        tok=tokens.get(i);
21        tag=tok.tag;
22        line=tok.line;
23        curType=Tag.VOID;
24        resultExprType=Tag.VOID;
25    }
26
27    public void init(){
28        program();
29    };
30
31    private void advance(){
32        i++;
33        tok=tokens.get(i);
34        tag=tok.tag;
35        line=tok.line;
36    }
37
38    private void error(String funcaoDoErro){
39        if(tag==Tag.EOF) {
40            if(line==-4)

```

```

41         System.out.println("Arquivo de entrada vazio");
42         return;
43     }
44     System.out.print(funcaoDoErro+";"+" Error(" + line + "): Token
nã esperado:"); //debug
45     tok.imprimeToken(tok);
46     while(tag!=Tag.EOF)
47         advance();
48 }
49
50 /* Seta o tipo basico com que se esta trabalhando em uma expressao
de atribuicao */
51 public void setCurType(int tipo){
52     curType = tipo;
53 }
54
55 public void resetResultExprType(){
56     resultExprType = Tag.VOID;
57 }
58
59 private void eat(int t){
60     if(tag==t){
61         //Checa se a tag e um tipo basico
62         if(tag == Tag.INT || tag == Tag.STR){
63             setCurType(tag);
64         }
65         //Caso ; deve resetar o resultado esperado de uma expressao
66         if(tag == Tag.PV){
67             resetResultExprType();
68         }
69         System.out.print("Token Consumido("+line+"): ");
70         tok.imprimeToken(tok);
71         advance();
72     }
73     else { error("Error in: eat"); }
74 }
75
76 private void program(){
77     switch(tag) {
78         //G:: program ::= init [decl-list] begin stmt-list stop
79         case Tag.INIT:
80             eat(Tag.INIT); declList();
81             if (tag == Tag.BEGIN) {
82                 eat(Tag.BEGIN); stmtList();
83             }
84             else if (tag == Tag.EOF) {
85                 System.out.println("Fim de arquivo inesperado.");

```

```

86         } else {
87             eat(Tag.STOP);
88         }
89         break;
90     default:
91         error("Error in: program");
92     }
93 }
94 }
95
96 private void declList(){
97     //G:: decl-list ::= decl ";" { decl ";" }
98     decl();
99     switch(tag) {
100         case Tag.PV:
101             eat(Tag.PV);
102             if(tag == Tag.BEGIN){
103                 break;
104             }else{
105                 declList();
106             }
107             break;
108         default:
109             error("Error in: declList");
110     }
111 }
112
113 private void decl(){
114     identList();
115     switch(tag) {
116         //G:: decl ::= ident-list is type
117         case Tag.IS:
118             eat(Tag.IS); type();
119             break;
120         case Tag.VRG:
121             eat(Tag.VRG); decl();
122             break;
123         default:
124             error("Error in: decl");
125     }
126 }
127
128 private void identList(){
129     switch(tag) {
130         //G:: ident-list ::= identifier {"," identifier}
131         case Tag.ID:
132             eat(Tag.ID);

```



```

133         break;
134     case Tag.VRG:
135         eat(Tag.VRG); eat(Tag.ID);
136         break;
137     default:
138         error("Error in: identList");
139     }
140 }
141
142 private void type(){
143     switch(tag) {
144         //G:: type ::= int
145         case Tag.INT:
146             eat(Tag.INT);
147             break;
148         //G:: type ::= string
149         case Tag.STR:
150             eat(Tag.STR);
151             break;
152         //G:: type ::= real
153         case Tag.REAL:
154             eat(Tag.REAL);
155             break;
156         default:
157             error("Error in: type");
158     }
159 }
160
161 private void stmtList(){
162     switch(tag) {
163         //G:: stmt-list ::= stmt ";" { stmt ";" }
164         case Tag.IF:
165             stmt(); stmtList();
166             break;
167         case Tag.ID:
168         case Tag.DO:
169         case Tag.READ:
170         case Tag.WRITE:
171             stmt(); eat(Tag.PV); stmtList();
172             break;
173         case Tag.WHILE:
174             break;
175         case Tag.STOP:
176             break;
177         case Tag.END:
178             break;
179         default:

```

```

180         error("Error in: stmtList");
181     }
182 }
183
184 private void stmt(){
185     switch(tag) {
186         //G:: stmt ::= assign-stmt
187         case Tag.ID:
188             assignStmt();
189             break;
190         //G:: stmt ::= if-stmt
191         case Tag.IF:
192             ifStmt();
193             break;
194         //G:: stmt ::= while-stmt
195         case Tag.DO:
196             doStmt();
197             break;
198         //G:: stmt ::= read-stmt
199         case Tag.READ:
200             readStmt();
201             break;
202         //G:: stmt ::= write-stmt
203         case Tag.WRITE:
204             writeStmt();
205             break;
206         default:
207             error("Error in: stmt");
208     }
209 }
210
211 private void assignStmt(){
212     switch(tag) {
213         //G:: assign-stmt ::= identifier "=" simple_expr
214         case Tag.ID:
215             eat(Tag.ID); eat(Tag.PPV); simpleExpr();
216             break;
217         default:
218             error("Error in: assignStmt");
219     }
220 }
221
222 private void ifStmt(){
223     switch(tag) {
224         //G:: if-stmt ::= if "(" condition ")" begin stmt-list end
225         else begin stmt-list end
226         case Tag.IF:

```

```

226         eat(Tag.IF); eat(Tag.AP); condition(); eat(Tag.FP); eat(
Tag.BEGIN); stmtList(); eat(Tag.END);
227
228         case Tag.ELSE:
229             eat(Tag.ELSE); eat(Tag.BEGIN); stmtList(); eat(Tag.END);
230             break;
231         default:
232             error("Error in: ifStmt");
233     }
234 }
235
236 private void doStmt(){
237     switch(tag) {
238         //G:: do-stmt ::= do stmt-list do-suffix.
239         case Tag.DO:
240             eat(Tag.DO); stmtList(); doSufix();
241             break;
242         default:
243             error("Error in: doStmt");
244     }
245 }
246
247 private void doSufix(){
248     switch(tag) {
249         //G:: stmt-sufix ::= while "(" condition ")"
250         case Tag.WHILE:
251             eat(Tag.WHILE); eat(Tag.AP); condition(); eat(Tag.FP);
252             break;
253         default:
254             error("Error in: doSufix");
255     }
256 }
257
258 private void readStmt(){
259     switch(tag) {
260         //G:: read-stmt ::= read "(" identifier ")"
261         case Tag.READ:
262             eat(Tag.READ); eat(Tag.AP); eat(Tag.ID); eat(Tag.FP);
263             break;
264         default:
265             error("Error in: redStmt");
266     }
267 }
268
269 private void writeStmt(){
270     switch(tag) {
271         //G:: write-stmt ::= write "(" writable ")"

```

```

272         case Tag.WRITE:
273             eat(Tag.WRITE); eat(Tag.AP); writable(); eat(Tag.FP);
274             break;
275         default:
276             error("Error in: writeStmt");
277     }
278 }
279
280 private void writable() {
281     simpleExpr();
282 }
283
284 private void condition(){
285     switch(tag) {
286         //G:: expression ::= simple-expr | simple-expr relop simple-
287         expr
288         case Tag.ID:
289         case Tag.NUM:
290         case Tag.LIT:
291         case Tag.AP:
292         case Tag.NOT:
293         case Tag.MIN:
294             simpleExpr();
295             switch(tag){
296                 case Tag.GT:
297                 case Tag.LT:
298                 case Tag.GE:
299                 case Tag.LE:
300                 case Tag.NE:
301                 case Tag.EQ:
302                     relop(); simpleExpr();
303                     break;
304                 case Tag.FP:
305                     break;
306                 default:
307                     error("Error in: condition, 'simple-expr relop
308 simple-expr'");
309             }
310             break;
311         default:
312             error("Error in: condition, 'simple-expr'");
313     }
314 }
315
316 private void simpleExpr(){
317     switch(tag) {
318         //G:: simple-expr ::= term | simple-expr addop term

```

```

317         case Tag.ID:
318         case Tag.NUM:
319         case Tag.LIT:
320         case Tag.AP:
321         case Tag.NOT:
322         case Tag.MIN:
323             term();
324             switch(tag){
325                 case Tag.MIN:
326                 case Tag.SUM:
327                     addop(); term();
328                     break;
329                 case Tag.PV:
330                 case Tag.FP:
331                 case Tag.GT:
332                 case Tag.LT:
333                 case Tag.GE:
334                 case Tag.LE:
335                 case Tag.NE:
336                 case Tag.EQ:
337                     break;
338                 default:
339                     error("Error in: simpleExpr, 'simple-expr addop
term'");
340             }
341             break;
342         default:
343             error("Error in: simpleExpr, 'term'");
344     }
345 }
346
347 private void term(){
348     switch(tag) {
349         //G:: term ::= factor-a | term mulop factor-a
350         case Tag.ID:
351         case Tag.NUM:
352         case Tag.LIT:
353         case Tag.AP:
354         case Tag.NOT:
355         case Tag.MIN:
356             factorA(); term();
357             break;
358         case Tag.PV:
359         case Tag.FP:
360         case Tag.SUM:
361         case Tag.GT:
362             break;

```

```

363         case Tag.MUL:
364         case Tag.DIV:
365         case Tag.AND:
366             mulop(); factorA();
367             break;
368         default:
369             error("Error in: termA");
370     }
371 }
372
373
374 private void factorA(){
375     switch(tag) {
376         //G:: factor-a ::= factor
377         case Tag.ID:
378         case Tag.NUM:
379         case Tag.LIT:
380         case Tag.AP:
381             factor();
382             break;
383         //G:: factor-a ::= not factor
384         case Tag.NOT:
385             eat(Tag.NOT); factor();
386             break;
387         //G:: factor-a ::= "-" factor
388         case Tag.MIN:
389             eat(Tag.MIN); factor();
390             break;
391         default:
392             error("Error in: factorA");
393     }
394 }
395
396 private void factor(){
397     switch(tag) {
398         //G:: factor ::= identifier
399         case Tag.ID:
400             eat(Tag.ID);
401             break;
402         //G:: factor ::= constant
403         case Tag.NUM:
404         case Tag.LIT:
405             constant();
406             break;
407         //G:: factor ::= "(" expression ")"
408         case Tag.AP:
409             eat(Tag.AP); condition(); eat(Tag.FP);

```

```

410         break;
411     default:
412         error("Error in: factor");
413     }
414 }
415
416 private void relop(){
417     switch(tag) {
418         //G:: relop ::= "="
419         case Tag.EQ:
420             eat(Tag.EQ);
421             break;
422         //G:: relop ::= ">"
423         case Tag.GT:
424             eat(Tag.GT);
425             break;
426         //G:: relop ::= "<"
427         case Tag.LT:
428             eat(Tag.LT);
429             break;
430         //G:: relop ::= "<>"
431         case Tag.NE:
432             eat(Tag.NE);
433             break;
434         //G:: relop ::= ">="
435         case Tag.GE:
436             eat(Tag.GE);
437             break;
438         //G:: relop ::= "<="
439         case Tag.LE:
440             eat(Tag.LE);
441             break;
442         default:
443             error("Error in: relop");
444     }
445 }
446
447 private void addop(){
448     switch(tag) {
449         //G:: addop ::= "+"
450         case Tag.SUM:
451             eat(Tag.SUM);
452             break;
453         //G:: addop ::= "-"
454         case Tag.MIN:
455             eat(Tag.MIN);
456             break;

```

```

457         //G:: addop ::= "OR"
458         case Tag.OR:
459             eat(Tag.OR);
460             break;
461         default:
462             error("Error in: addop");
463     }
464 }
465
466 private void mulop(){
467     switch(tag) {
468         //G:: mulop ::= "*"
469         case Tag.MUL:
470             eat(Tag.MUL);
471             break;
472         //G:: mulop ::= "/"
473         case Tag.DIV:
474             eat(Tag.DIV);
475             break;
476         //G:: mulop ::= "AND"
477         case Tag.AND:
478             eat(Tag.AND);
479             break;
480         default:
481             error("Error in: mulop");
482     }
483 }
484
485 private void constant() {
486     switch (tag) {
487         //G:: constant ::= integer_const
488         case Tag.NUM:
489             eat(Tag.NUM);
490             break;
491         //G:: constant ::= literal
492         case Tag.LIT:
493             eat(Tag.LIT);
494             break;
495         default:
496             error("Error in: constant");
497     }
498 }
499 }

```


9. SUGESTÕES PARA TRABALHOS FUTUROS

Nesta etapa, finalizamos o processo de análise Sintática do Compilador (Etapa 2), assim, fica necessário implementar na próxima etapa o analisador Semântico e o Gerador de código. Para as próximas etapas, fica sugerido que devemos implementar a recuperação de Erros para que seja possível implementar um compilador mais eficiente e próximo da realidade.

10. REFERÊNCIAS

10.1. LIVROS

A. V. Aho, R. Sethi, J. D. Ullman Compiladores: Princípios, técnicas e ferramentas LTC - Livros Técnicos e Científicos Editora, 2013

10.2. PROGRAMAS

- Apache Netbeans
- Visual Studio
- Microsoft Office Excel

10.3. SITE

- Site : Overleaf Para criação do relatório