



**Universidade Estadual de Santa Cruz – UESC**

**Relatórios de Implementações de Analisador Descendente Recursivo  
para GLC**

**Docente César Alberto Bravo Pariente**

**Discente Matheus Miranda Brandão**

**Matrícula 201820065**

**Disciplina Compiladores.**

**Curso Ciência da Computação**

**Semestre 2022.2**

**Ilhéus – BA  
2022**

# Índice

---

<b>GLC LL</b>	<b>3</b>
Produções:	3
<b>Compilando e Executando</b>	<b>4</b>
<b>Testes</b>	<b>5</b>
$m() \{ r(1); \}$	5
$m() \{ h=(x+y); r(0); \}$	5
$m() \{ (1-1); r(1); \}$	5
$m() \{ w(1) \{ (1/x); \}; r(1); \}$	6
$n() \{ (0/y); r(y); \} \quad g() \{ i=y; r(x); \} \quad m() \{ (1-x); r(0); \}$	6
<b>Link para download</b>	<b>7</b>
<b>Referências</b>	<b>8</b>

# GLC LL

---

O projeto consiste na implementação em C de um algoritmo que busca simular um Analisador Descendente Recursivo que reconhece a linguagem gerada por uma gramática livre de contexto. A execução do código recebe como entrada a palavra e retorna se a palavra foi aceita ou ocorreu um erro, suas produções e o vetor referente a Árvore de Análise.

O analisador implementado ignora o token ' '.

## **Produções:**

$p_1: S \rightarrow M \mid GM \mid NGM$

$p_4: N \rightarrow n() \{ C; r(E); \}$

$p_5: G \rightarrow g() \{ C; r(E); \}$

$p_6: M \rightarrow m() \{ C; r(E); \}$

$p_7: E \rightarrow 0 \mid 1 \mid x \mid y \mid (EXE)$

$p_{12}: X \rightarrow + \mid - \mid * \mid /$

$p_{16}: C \rightarrow h=E \mid i=E \mid j=E \mid k=E \mid z=E \mid (EXE) \mid w(E) \{ C; \} \mid f(E) \{ C; \} \mid o(E;E;E) \{ C; \}$

# Compilando e Executando

---

Para a execução não é necessário o uso de nenhuma dependência, basta compila-lo normalmente.

```
$ gcc proj2_b.c -o proj2_b
```

Ao executar é necessário digitar o nome do arquivo destino contendo as palavras, caso contrário resultará em erro.

Exemplo:

```
$ ./proj2_b examples/inputs.txt
```

Neste projeto pode-se adicionar num .txt todas as palavras separadas por uma quebra de linha. Em caso de erro, pularemos para a próxima palavra.

Exemplo:

```
m(){ r(1); }
m(){ h=(x+y); r(0); }
m(){ (1-1); r(1); }
m(){ w(1) { (1/x); }; r(1); }
n() { (0/y); r(y); } g() { i=y; r(x); } m() { (1-x); r(0); }
```

# Testes

---

Caso a entrada dada seja incorreta o programa irá imprimir os tokens até o momento, então avisará sobre o erro, informará qual o código e o token inesperado, então pulará para a próxima palavra. Como outputs temos a palavra analisada, as produções e a árvore sintática (no formato 'palavra, posicao\_no\_vetor'), foi considerado o pior caso da produção 24, onde temos uma árvore n-ária de 4.

Para criação de palavras compatíveis com a linguagem gerada pela GLC foi utilizado o website "CFG Developer".

**m(){ r(1); }**

Palavra 1: m(

Erro 37. Token '1' inesperado. Prosseguindo para proxima palavra.

Producoes: P1

Arvore: S[0] M[1]

**m(){ h=(x+y); r(0); }**

Palavra 2: m(){ h=(x+y); r(0); }

Palavra aceita.

Producoes: P1, P6, P16, P11, P9, P12, P10, P7

Arvore: S[0] M[1] C[5] E[6] E[21] 0[25] E[85] X[86] E[87] x[341]  
+[345] y[349]

**m(){ (1-1); r(1); }**

Palavra 3: m(){ (1-1); r(1); }

Palavra aceita.

Producoes: P1, P6, P21, P8, P13, P8, P8

Arvore: S[0] M[1] C[5] E[6] E[21] X[22] E[23] 1[25] 1[85] -[89] 1[93]

**$m() \{ w(1) \{ (1/x); \}; r(1); \}$**

Palavra 4:  $m() \{ w(1) \{ (1/x); \}; r(1); \}$

Palavra aceita.

Producoes: P1, P6, P22, P8, P21, P8, P15, P9, P8

Arvore: S[0] M[1] C[5] E[6] E[21] C[22] 1[25] 1[85] E[89] X[90] E[91]  
1[357] /[361] x[365]

**$n() \{ (0/y); r(y); \} g() \{ i=y; r(x); \} m() \{ (1-x); r(0); \}$**

Palavra 5:  $n() \{ (0/y); r(y); \} g() \{ i=y; r(x); \} m() \{ (1-x); r(0); \}$

Palavra aceita.

Producoes: P3, P4, P21, P7, P15, P10, P10, P5, P17, P10, P9, P6, P21, P8, P13, P9, P7

Arvore: S[0] N[1] G[2] M[3] C[5] E[6] C[9] E[10] C[13] E[14] E[21]  
X[22] E[23] y[25] E[37] x[41] E[53] X[54] E[55] 0[57] 0[85] /[89] y[93]  
y[149] 1[213] -[217] x[221]

# Link para download

---

Código fonte e exemplos encontram-se para download no seguinte link:

<https://github.com/MatBrands/Compiladores/tree/master/Proj2/Proj2b>

# Referências

---

<https://web.stanford.edu/class/archive/cs/cs103/cs103.1156/tools/cfg/>