

一、实验目的与方法

1. 实验目的

加深对自顶向下语法制导翻译技术的理解与掌握，加深对自底下上语法制导翻译技术的理解与掌握。巩固对语义分析的基本功能和原理的认识，理解中间代码生产的作用。

2. 实验方法

对用户输入的字符串表达式进行语义分析，并在过程中生成中间代码，以及进行对符号表的增删改操作。

3. 实验语言

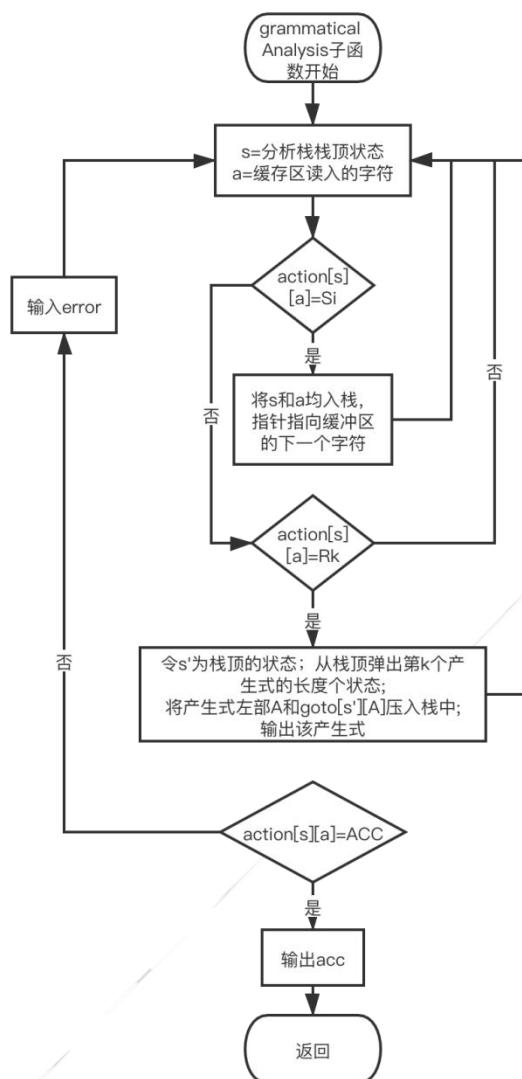
所使用的语言为 C++语言；操作系统环境为 macOS Mojave；软件环境为 CLion。

二、实验总体流程与函数功能描述

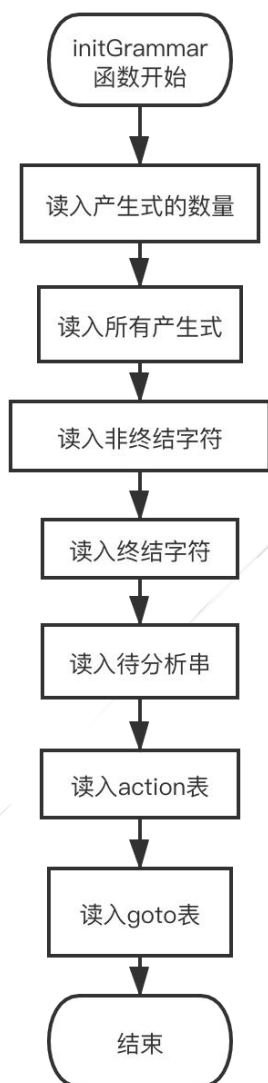
实验三是在实验二的基础上进行的，由于实验三要求支持基本的赋值操作，但是实验二中我只支持加法和乘法操作，所以更新了实验二的预测分析表以及输入的文法，将在第三部分的实验内容进行展示。

实验二中的读入函数和语法分析函数在算法框架上是不变的，如下：

1. 语法分析程序 void grammaticalAnalysis()



2. 根据读入文件初始化数据结构 void initGrammar()



3. 判断 `ch` 是否是终结符 `int isInT(char ch)`
若是，返回下标加一；若不是，返回 0。
4. 判断 `ch` 是否是非终结符 `int isInN(char ch)`
若是，返回下标加一；若不是，返回 0。

增加的主要有两个部分：

1. 规约时根据不同的产生式生成不同的中间代码：

```
// 中间代码生成
// 三地址码的生成
int p_num = action[s][j].second;

if (p_num == 1){
    string fir = tri.back();
    tri.pop_back();
    string sec = tri.back();
    tri.pop_back();
    cout << sec + " = " + fir << endl;
}
else if (p_num == 2) {
    string fir = tri.back();
    tri.pop_back();
    string sec = tri.back();
    tri.pop_back();
    tnum++;
    string tmpn;
    tmpn = to_string(tnum);
    tmpn = "t" + tmpn;
    tri.push_back(tmpn);
    cout << tmpn << " = " + sec + " + " + fir << endl;
} else if (p_num == 3){
    string fir = tri.back();
    tri.pop_back();
    string sec = tri.back();
    tri.pop_back();
    tnum++;
    string tmpn;
    tmpn = to_string(tnum);
    tmpn = "t" + tmpn;
    tri.push_back(tmpn);
    cout << tmpn << " = " + sec + " - " + fir << endl;
} else if (p_num == 5) {
    string fir = tri.back();
    tri.pop_back();
    string sec = tri.back();
    tri.pop_back();
    tnum++;
    string tmpn;
    tmpn = to_string(tnum);
    tmpn = "t" + tmpn;
    tri.push_back(tmpn);
    cout << tmpn << " = " + sec + " * " + fir << endl;
} else if (p_num == 6){
    string fir = tri.back();
    tri.pop_back();
    string sec = tri.back();
    tri.pop_back();
    tnum++;
    string tmpn;
    tmpn = to_string(tnum);
    tmpn = "t" + tmpn;
    tri.push_back(tmpn);
    cout << tmpn << " = " + sec + " / " + fir << endl;
}
```

2. 判断是否符合变量命名规则 int isvar(char ch)
3. 判断是否符合数字形式 int isnum(char ch)

三、 实验内容

1. 拓广文法 G

$S \rightarrow A$

$A \rightarrow F=B$

$B \rightarrow B+T$

$B \rightarrow B-T$

$B \rightarrow T$

$T \rightarrow T * F$

$T \rightarrow T / F$

$T \rightarrow F$

$F \rightarrow (B)$

$F \rightarrow a$

2. LR(1)分析表

action_table.txt									
	+	-	*	/	=	()	a	\$
0						S1		S2	
1						S5		S6	
2					R9				
3									ACC
4					S10				
5						S5		S6	
6	R9	R9	R9	R9			R9		
7	S12	S13					S14		
8	R4	R4	S15	S16			R4		
9	R7	R7	R7	R7			R7		
10						S17		S18	
11	S12	S13					S22		
12						S5		S6	
13						S5		S6	
14					R8				
15						S5		S6	
16						S5		S6	
17						S5		S6	
18	R9	R9	R9	R9					R9
19	S28	S29							R1
20	R4	R4	S30	S31					R4
21	R7	R7	R7	R7					R7
22	R8	R8	R8	R8			R8		
23	R2	R2	S15	S16			R2		
24	R3	R3	S15	S16			R3		
25	R5	R5	R5	R5			R5		
26	R6	R6	R6	R6			R6		
27	S12	S13					S32		
28						S17		S18	
29						S17		S18	
30						S17		S18	
31						S17		S18	
32	R8	R8	R8	R8					R8
33	R2	R2	S30	S31					R2
34	R3	R3	S30	S31					R3
35	R5	R5	R5	R5					R5
36	R6	R6	R6	R6					R6

	A	B	T	F
0	3			4
1		7	8	9
2				
3				
4				
5		11	8	9
6				
7				
8				
9				
10		19	20	21
11				
12			23	9
13			24	9
14				
15				25
16				26
17		27	8	9
18				
19				
20				
21				
22				
23				
24				
25				
26				
27				
28			33	21
29			34	21
30				35
31				36
32				
33				
34				
35				
36				

3. 简单赋值语句的翻译

$S \rightarrow id := E$ { $p = \text{lookup}(id.name)$; if $p == \text{nil}$ then error;

$\text{gencode}(p \text{ '=' } E.addr)$;}

$E \rightarrow E1 + E2$ { $E.addr = \text{newtemp}()$;

$\text{gencode}(E.addr \text{ '=' } E1.addr \text{ '+' } E2.addr)$;}

$E \rightarrow E1 * E2$ { $E.addr = \text{newtemp}()$;

$\text{gencode}(E.addr \text{ '=' } E1.addr \text{ '*' } E2.addr)$;}

$E \rightarrow -E1$ { $E.addr = \text{newtemp}()$;

$\text{gencode}(E.addr \text{ '=' 'uminus' } E1.addr)$;}

$E \rightarrow (E1)$ { $E.addr = E1.addr$;}

$E \rightarrow id$ { $E.addr = \text{lookup}(id.name)$; if $E.addr == \text{nil}$ then error;}

4. 三地址码的数据结构

```
/* 三元组符号栈 */
vector<string> tri;
```

5. 符号表的数据结构

```
/* 变量扩展符号表 */
vector<string> vartab;
```

四、实验结果与分析

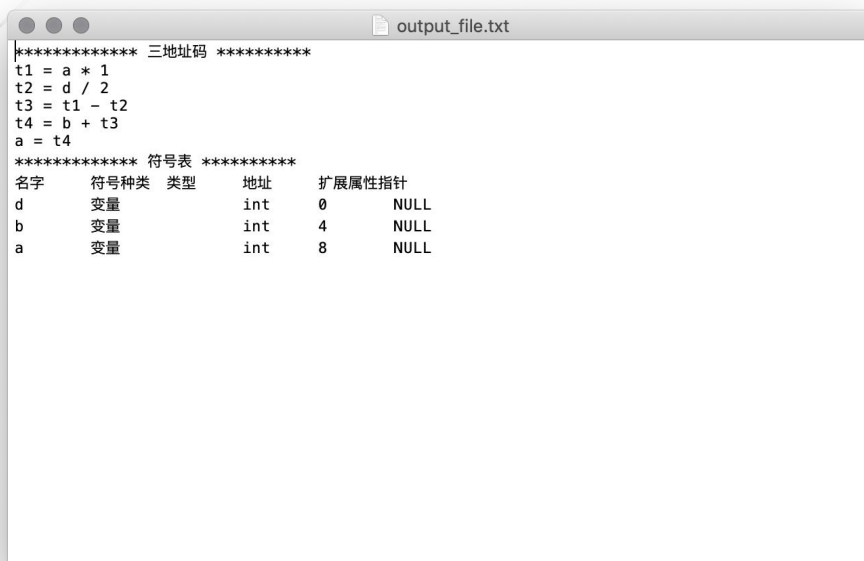
① 实验输入



```
10
S->A
A->F=B
B->B+T
B->B-T
B->T
T->T*F
T->T/F
T->F
F->(B)
F->a
+ - * / = ( ) a #
S A B T F #
a=b+(a*1-d/2)$
```

第一行表示有 10 个产生式。接下来的 10 行代表了 10 个产生式。第 11 行表示终结字符（其中 a 表示 ID）。第 12 行表示非终结字符。最后一行表示待分析的字符串。

② 实验输出



```
***** 三地址码 *****
t1 = a * 1
t2 = d / 2
t3 = t1 - t2
t4 = b + t3
a = t4
***** 符号表 *****
名字      符号种类  类型      地址      扩展属性指针
d          变量        int        0          NULL
b          变量        int        4          NULL
a          变量        int        8          NULL
```