《编译原理》实验报告

学　　院　计算机工程与科学学院

组　　号　10 组

实验题号　实验四

日　　期　2025 年 05 月 14 日

学号	姓名	主要工作	贡献因子
22122792	王潇	实验代码与报告	0.25
22121037	钱鹏程	实验代码	0.25
22121639	何勇乐	实验代码与报告	0.25
22121630	汪江豪	实验代码	0.25

《编译原理》实验报告

一、 实验题目

根据 PL/0 语言文法，在已有的语法分析程序中添加语义处理部分，重点实现表达式的语义分析与计算功能。在实验三的语法分析器基础上扩展语义子程序，确保能够正确处理经过语法分析后的合法表达式，并输出其计算值。

二、 实验内容设计与实现

本程序使用 SLR(1)分析法对表达式进行语义值计算，通过画出状态迁移图可以得到如表1所示的状态迁移表。每个产生式对应的语义动作如表2所示。

	+	-	*	/	(	)	id	int	#	S	A	B
0	S13	S14			S10		S8	S9		1	18	17
1	S3	S2							ACC			
2					S10		S8	S9			20	17
3					S10		S8	S9			4	17
4	r4	r4	S6	S5		r4			r4			
5					S10		S8	S9				19
6					S10		S8	S9				7
7	r7	r7	r7	r7		r7			r7			
8	r9	r9	r9	r9		r9			r9			
9	r10	r10	r10	r10		r10			r10			
10	S13	S14			S10		S8	S9		11	18	17
11	S3	S2				S12						
12	r11	r11	r11	r11		r11			r11			
13					S10		S8	S9			15	17
14					S10		S8	S9			16	17
15	r1	r1	S6	S5		r1			r1			
16	r2	r2	S6	S5		r2			r2			
17	r6	r6	r6	r6		r6			r6			
18	r3	r3	S6	S5		r3			r3			
19	r8	r8	r8	r8		r8			r8			
20	r5	r5	S6	S5		r5			r5			

表 1　状态迁移表

r1	S.val = A.val
r2	S.val = -A.val
r3	S.val = A.val
r4	S.val = S1.val + A.val
r5	S.val = S1.val - A.val
r6	A.val = B.val
r7	A.val = B.val * B.val
r8	A.val = B.val / B.val
r9	B.val = NUM
r10	B.val = IDENT(-100)
r11	B.val = S.val

表 2　语义动作

doReduction 函数的核心功能是在 SLR(1)语法分析过程中根据当前的归约规则执行相应的语义动作。该函数的设计充分体现了语法分析与语义处理的有机结合，使得在识别表达式结构的同时能够完成对表达式值的计算。

函数接收分析栈、状态迁移表、当前使用的文法规则编号为参数，在每次归约时，它会依据不同的产生式编号进入对应的处理逻辑，完成从栈顶取出操作数、构造新符号、更新语义值、压入新状态等一系列操作。该程序使用的是 SLR(1)分析方法，因此每一步归约都必须依据状态迁移表找到下一个状态，并据此更新分析栈中的内容，确保整个分析过程的状态流转正确无误。

当操作数为标识符时，函数不会立即进行数值计算，而是保留其符号名；而当操作数为常量时，则直接参与运算并生成新的语义值。这种设计不仅保证了程序在面对纯数字表达式时可以正确求值，也使得其在遇到变量标识符时仍能继续进行语法分析而不中断，从而兼顾了语法检查与语义处理的需求。

代码 1

```
...
if (isdigit(tokensQueue.front().at(0))) {
    action =
      stm.getCellValue(to_string(AnalysisStack.top().state),
"int");
} else if (isalpha(tokensQueue.front().at(0))) { // 标识符的符号记
为id
    action =
      stm.getCellValue(to_string(AnalysisStack.top().state),
"id");
    cout << "IDENT at position " << length_origin -
tokensQueue.size() << endl;
    cout << "only SYNTAX CHECKING is performed\n"
    exist_id = true;
} else { // 运算符、括号直接找动作
    action =
stm.getCellValue(to_string(AnalysisStack.top().state),
tokensQueue.front());
}
...
```

代码 2　语法错误处理

```
  ...
  if (action == "") { // 语法错误
      cout << "Syntax ERROR!\n";
      cout << "Unexcepted \"" << tokensQueue.front() << "\" at
position "
           << length_origin - tokensQueue.size() << endl;
      cout << endl;
      string filePath = "test.txt";
      ifstream inputFile(filePath);
      string line;
      while (std::getline(inputFile, line)) {
          std::cout << line << std::endl;
      }
      inputFile.close();
      for (int i = 0; i <= length_origin - tokensQueue.size();
i++) {
          cout << "^";
      }
      cout << endl;
      break;
  }
  ...
```

该函数在处理每个文法规则时都遵循统一的流程：先弹出已匹配的符号，计算新的语义值和生成新的符号，再查找下一状态并将新的状态和符号压入分析栈。这一流程的高度一致性使得代码结构清晰、逻辑严密，同时也便于扩展和调试。例如，在处理加法、减法、乘法、除法等二元运算时，函数均按照“取两个操作数、执行运算、压入结果”的模式进行操作，而在处理括号或单目负号等一元结构时，则只需调整操作数的数量和运算顺序即可。这种统一性与灵活性相结合的设计提升了代码的可维护性和复用性。

代码 3　移进与规约

```
...
else if (action == "ACC") { // 语法全部正确
    cout << "Syntax CORRECT!!!" << endl;
    if (!exist_id) {
        cout << "value is:" << AnalysisStack.top().value <<
endl;
    }
    break;
} else if (action.at(0) == 'S') {                // 做移进
    if (isdigit(tokensQueue.front().at(0))) { // 要移进的是数字

AnalysisStack.push(AnalysisStackElement(extractNumber(action),
          tokensQueue.front(), tokensQueue.front(),
          tokensQueue.front()));
    } else if (isalpha(tokensQueue.front().at(0))) { // 移进标识符

AnalysisStack.push(AnalysisStackElement(extractNumber(action),
          tokensQueue.front(), "-100", tokensQueue.front()));
    } else { // 移进的是运算符或者括号，没有语义值
        AnalysisStack.push(AnalysisStackElement(
          extractNumber(action), tokensQueue.front(), "-"));
    }
    tokensQueue.pop();
} else if (action.at(0) == 'r') {             // 归约
    int production = extractNumber(action); // 归约用的式子的序号
    if (!doReduction(AnalysisStack, stm, production,
quadrupleQueue)) {
        break;
    }
}
else {
    cout << "UNKNOWN!" << endl;
    break;
}
...
```

函数还包含了完善的错误处理机制，一旦在查找下一状态时返回空字符串，即表示无法继续归约，函数将输出语法错误信息并返回 false，从而及时终止整

个分析过程。这种机制增强了程序的鲁棒性，使其在面对非法输入时能够做出明确反应，提高了程序的实用性。

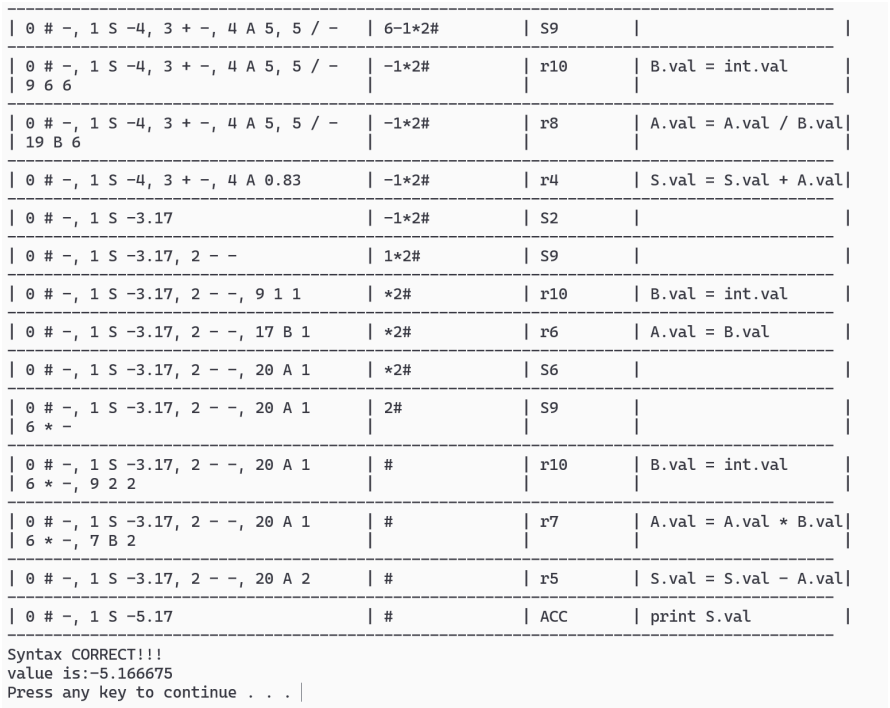
三、 实验数据测试

3.1. 测试 1

输入如下。

```
1/2/3*(-2*(-3)*(-4))+5/6-1*2
```

从输出结果可以看见，表达式结果是 −5.166675。



```
-------------------------------------------------------------------------------------------
| 0 # -, 1 S -4, 3 + -, 4 A 5, 5 / -   | 6-1*2#          | S9        |                       |
-------------------------------------------------------------------------------------------
| 0 # -, 1 S -4, 3 + -, 4 A 5, 5 / -   | -1*2#           | r10       | B.val = int.val       |
| 9 6 6                                |                 |           |                       |
-------------------------------------------------------------------------------------------
| 0 # -, 1 S -4, 3 + -, 4 A 5, 5 / -   | -1*2#           | r8        | A.val = A.val / B.val|
| 19 B 6                               |                 |           |                       |
-------------------------------------------------------------------------------------------
| 0 # -, 1 S -4, 3 + -, 4 A 0.83       | -1*2#           | r4        | S.val = S.val + A.val|
-------------------------------------------------------------------------------------------
| 0 # -, 1 S -3.17                     | -1*2#           | S2        |                       |
-------------------------------------------------------------------------------------------
| 0 # -, 1 S -3.17, 2 - -              | 1*2#            | S9        |                       |
-------------------------------------------------------------------------------------------
| 0 # -, 1 S -3.17, 2 - -, 9 1 1       | *2#             | r10       | B.val = int.val       |
-------------------------------------------------------------------------------------------
| 0 # -, 1 S -3.17, 2 - -, 17 B 1      | *2#             | r6        | A.val = B.val         |
-------------------------------------------------------------------------------------------
| 0 # -, 1 S -3.17, 2 - -, 20 A 1      | *2#             | S6        |                       |
-------------------------------------------------------------------------------------------
| 0 # -, 1 S -3.17, 2 - -, 20 A 1      | 2#              | S9        |                       |
| 6 * -                                |                 |           |                       |
-------------------------------------------------------------------------------------------
| 0 # -, 1 S -3.17, 2 - -, 20 A 1      | #               | r10       | B.val = int.val       |
| 6 * -, 9 2 2                         |                 |           |                       |
-------------------------------------------------------------------------------------------
| 0 # -, 1 S -3.17, 2 - -, 20 A 1      | #               | r7        | A.val = A.val * B.val|
| 6 * -, 7 B 2                         |                 |           |                       |
-------------------------------------------------------------------------------------------
| 0 # -, 1 S -3.17, 2 - -, 20 A 2      | #               | r5        | S.val = S.val - A.val|
-------------------------------------------------------------------------------------------
| 0 # -, 1 S -5.17                     | #               | ACC       | print S.val           |
-------------------------------------------------------------------------------------------
Syntax CORRECT!!!
value is:-5.166675
Press any key to continue . . .
```

图 1　正例结果

3.2. 测试 2

输入如下。

```
-5*(+8)/9-4*(5*1*2)/(8/(-8))
```

计算结果为 35.555556。

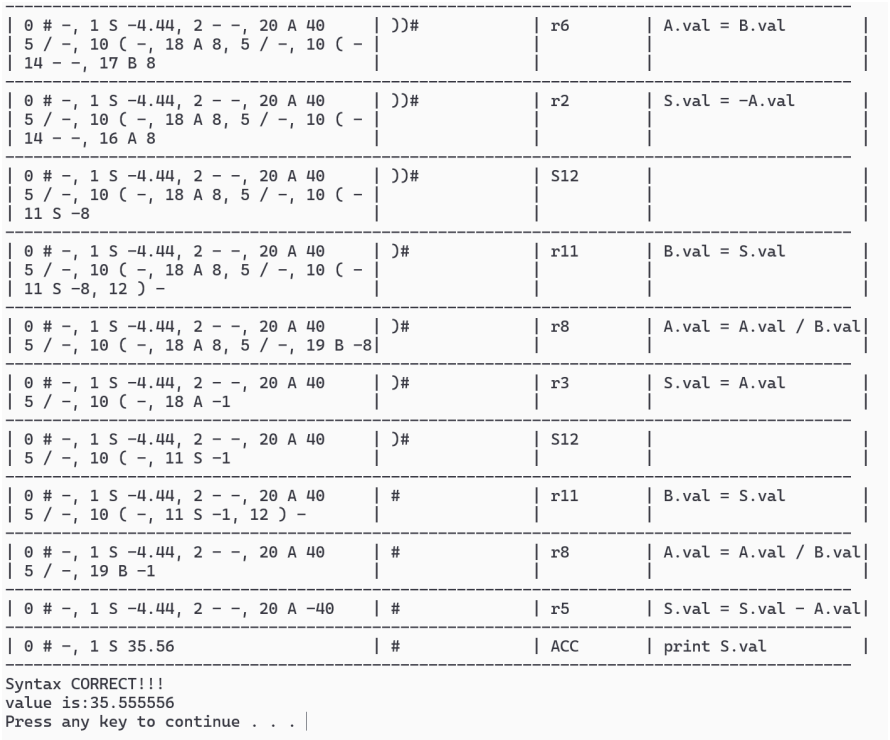


图 2　正例结果

3.3. 测试 3

输入如下，存在未知符号“@”。

```
1.  -1+2*3*(7/(5+6)-8)/9+(-5)@5
```

可以从输出中发现当前出现了错误，无法计算求值，且每一步都输出了当前的状态和符号。

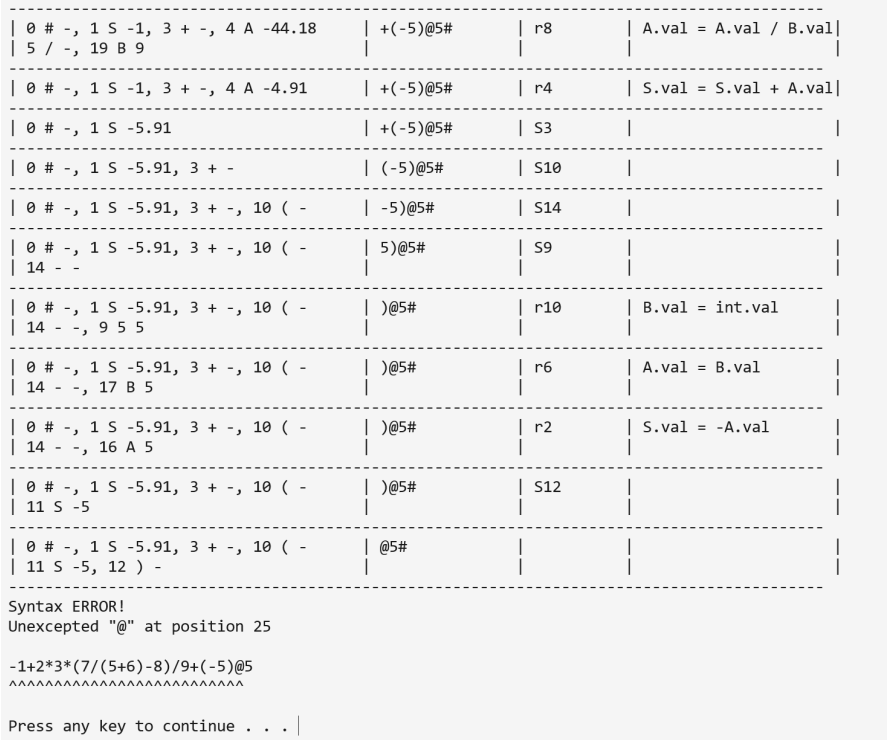


图 3　负例结果

3.4. 测试 4

输入如下，存在未定义符号“*-”。

```
1.  -5/8/9*-9+(5*(6+7)*7)-8/(5+6)
```

输出如下，程序在遇到错误时会输出当前的状态和符号以及发生错误的位置

```
-------------------------------------------------------------------------------------
| 0 # -, 14 - -, 16 A 5                | /8/9*-9+(5*(6+7| S5       |                    |
|                                      | )*7)-8/(5+6)#  |          |                    |
-------------------------------------------------------------------------------------
| 0 # -, 14 - -, 16 A 5, 5 / -         | 8/9*-9+(5*(6+7)| S9       |                    |
|                                      | *7)-8/(5+6)#   |          |                    |
-------------------------------------------------------------------------------------
| 0 # -, 14 - -, 16 A 5, 5 / -, 9 8 8  | /9*-9+(5*(6+7)*| r10      | B.val = int.val    |
|                                      | 7)-8/(5+6)#    |          |                    |
-------------------------------------------------------------------------------------
| 0 # -, 14 - -, 16 A 5, 5 / -, 19 B 8 | /9*-9+(5*(6+7)*| r8       | A.val = A.val / B.val|
|                                      | 7)-8/(5+6)#    |          |                    |
-------------------------------------------------------------------------------------
| 0 # -, 14 - -, 16 A 0.62             | /9*-9+(5*(6+7)*| S5       |                    |
|                                      | 7)-8/(5+6)#    |          |                    |
-------------------------------------------------------------------------------------
| 0 # -, 14 - -, 16 A 0.62, 5 / -      | 9*-9+(5*(6+7)*7| S9       |                    |
|                                      | )-8/(5+6)#     |          |                    |
-------------------------------------------------------------------------------------
| 0 # -, 14 - -, 16 A 0.62, 5 / -      | *-9+(5*(6+7)*7)| r10      | B.val = int.val    |
| 9 9 9                                | -8/(5+6)#      |          |                    |
-------------------------------------------------------------------------------------
| 0 # -, 14 - -, 16 A 0.62, 5 / -      | *-9+(5*(6+7)*7)| r8       | A.val = A.val / B.val|
| 19 B 9                               | -8/(5+6)#      |          |                    |
-------------------------------------------------------------------------------------
| 0 # -, 14 - -, 16 A 0.07             | *-9+(5*(6+7)*7)| S6       |                    |
|                                      | -8/(5+6)#      |          |                    |
-------------------------------------------------------------------------------------
| 0 # -, 14 - -, 16 A 0.07, 6 * -      | -9+(5*(6+7)*7)-|          |                    |
|                                      | 8/(5+6)#       |          |                    |
-------------------------------------------------------------------------------------
Syntax ERROR!
Unexcepted "-" at position 7

-5/8/9*-9+(5*(6+7)*7)-8/(5+6)
^^^^^^^^

Press any key to continue . . .
```

图 4　负例结果

3.5. 测试 5

输入如下，本次输入尝试连除和单括号表达式。

```
1.  +8/2/3-4-2*(-2)/(+2)+4/9
```

```
-------------------------------------------------------------------------------------
| 0 # -, 1 S -2.67, 2 - -, 20 A -2     | +4/9#          | r5       | S.val = S.val - A.val|
-------------------------------------------------------------------------------------
| 0 # -, 1 S -0.67                     | +4/9#          | S3       |                    |
-------------------------------------------------------------------------------------
| 0 # -, 1 S -0.67, 3 + -              | 4/9#           | S9       |                    |
-------------------------------------------------------------------------------------
| 0 # -, 1 S -0.67, 3 + -, 9 4 4       | /9#            | r10      | B.val = int.val    |
-------------------------------------------------------------------------------------
| 0 # -, 1 S -0.67, 3 + -, 17 B 4      | /9#            | r6       | A.val = B.val      |
-------------------------------------------------------------------------------------
| 0 # -, 1 S -0.67, 3 + -, 4 A 4       | /9#            | S5       |                    |
-------------------------------------------------------------------------------------
| 0 # -, 1 S -0.67, 3 + -, 4 A 4, 5 / -| 9#             | S9       |                    |
-------------------------------------------------------------------------------------
| 0 # -, 1 S -0.67, 3 + -, 4 A 4, 5 / -| #              | r10      | B.val = int.val    |
| 9 9 9                                |                |          |                    |
-------------------------------------------------------------------------------------
| 0 # -, 1 S -0.67, 3 + -, 4 A 4, 5 / -| #              | r8       | A.val = A.val / B.val|
| 19 B 9                               |                |          |                    |
-------------------------------------------------------------------------------------
| 0 # -, 1 S -0.67, 3 + -, 4 A 0.44    | #              | r4       | S.val = S.val + A.val|
-------------------------------------------------------------------------------------
| 0 # -, 1 S -0.22                     | #              | ACC      | print S.val        |
-------------------------------------------------------------------------------------
Syntax CORRECT!!!
value is:-0.222223
Press any key to continue . . .
```

图 5　正例结果

四、 实验总结

本次实验的主要任务是实现一个基于 SLR(1)分析法的表达式语义值计算程序。通过对已有的语法分析器进行扩展，我们成功地将语法分析与语义处理结合起来，实现了对表达式的正确求值。

通过本次实验，我们深入理解了 SLR(1)分析法的工作原理，掌握了如何在语法分析过程中进行语义处理。同时，我们也认识到语法分析与语义处理之间的密切关系，以及如何通过合理的设计实现两者的有机结合。