蘭州大學
LANZHOU UNIVERSITY

语义分析器

授课老师: 燕昊
课程: 编译原理实验
班级: 计算机基地班
姓名: 王子豪
学号: 320200945691

June 5, 2023

目录

1 源语言及目标语言的介绍

1.1 源语言介绍

- 源语言文法定义如下：

```
1  PROG     →   {  DECLS  STMTS  }
2
3  DECLS    →   DECLS  DECL    |   empty
4  DECL     →   int  NAMES;  |  bool  NAMES;
5  NAMES    →   NAMES ,  NAME  |  NAME
6  NAME     →   id
7
8  STMTS    →   STMTS  STMT  |   STMT
9  STMT     →   id  =  EXPR;    |   id := BOOL;
10 STMT     →   if  id   then  STMT
11 STMT     →   if   id   then  STMT  else STMT
12 STMT     →   while   id  do  STMT
13 STMT     →   {  STMTS   STMT  }
14 STMT     →   read  id;
15 STMT     →   write  id;
```

- 其中算术表达式使用运算符优先级，并没有严格区分 *bool* 形与 *int* 形。

1.2 目标语言介绍

目标语言使用四元组的形式

$$(result, num1, operator, num2)$$

- result 表示结果存放的位置
- num1 表示第一个运算数
- operator 表示操作类型
- num2 表示第二个运算符

一些特殊操作四元表达式形式如下：

- (a, none, read, none) -> read a;
- (cond1, =, t1, none) -> cond1 = t1;
- (15, cond1, je, 0) -> 如果 cond1 == 0，跳转到 15。
- (4, none, jmp, none) -> 立即跳转到 4。
- (t1, a, >=, b) -> 若 a>= b, 则 t1=1 或 t1 为真。

2 关键算法

根据源语言的特性，关键算法主要体现在运算表达式及跳转语句上（while, if）

2.1 运算表达式

在中缀转后缀的过程中直接进行运算，由于这些运算都是中间运算且为基本运算，直接输出每次运算对应的四元表达式即可。每次运算的结果都存储在 t_n 中，并将 t_n 入栈。

代码体现如下：

```
string num2 = st2.top(); st2.pop();
string num1 = st2.top(); st2.pop();
string res = "t" + to_string(++cnt_st2);
st2.push(res);
Node temp3 = op.top(); op.pop();
quat.push_back({res, num1, temp3.val, num2});
pfix.push_back(temp3);
```

2.2 跳转语句

采用链表回填的方法，由于每个判断条件都为 id，所以实现起来并不复杂。

每次进入 $while$ 或 if 语句时，开两个数组分别记录所有需要跳转到真假出口的位置，对于 $while$ 语句还需记录入口编号。读到真假出口时回填两个数组即可。

代码体现如下：

```
vector<int> T, F;//真假出口
int ent = quat.size();//入口编号
...
...
string res, num1(Val), op("je"), num2("0");//等于0跳转到假出口
quat.push_back({res, num1, op, num2});//此时的res为空值，等待回填
F.push_back(quat.size() - 1);//记录需要回填的编号
...
...
开始回填
for (int i = 0; i < F.size(); ++i)
{
        quat[F[i]].result = to_string(num);
}
```

3 测试结果

3.1 样例 1

- input1

```
//program 1: add two numbers.
{
        int a, b, c  ;
        a = 1;
        b = 2;
        c = a + b ;
}
```

- output1

```
0:  (a, =, 1, none)
1:  (b, =, 2, none)
2:  (t1, a, +, b)
3:  (c, =, t1, none)
```

3.2 样例 2

- input2

```
//program 2: read  in and add two numbers,
//then print the result .
{
        int a, b, c  ;
        read a ;
        read b ;
        c = a + b ;
        write c  ;
}
```

- output2

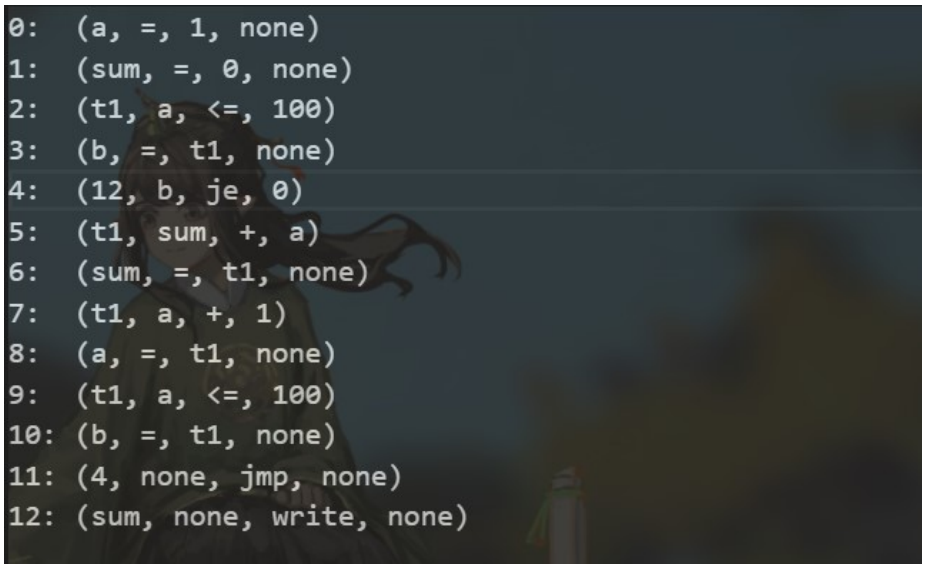
```
0:  (a, none, read, none)
1:  (b, none, read, none)
2:  (t1, a, +, b)
3:  (c, =, t1, none)
4:  (c, none, write, none)
```

3.3 样例 3

- input3

```
/*program 3: add numbers from 1 to 100
 *and print the result.
 */
{
        int a , sum ;
        bool  b ;
        a = 1 ;
        sum = 0 ;
        b := a <= 100 ;
        while b do
        {
                sum = sum + a ;
                a = a + 1 ;
                b := a <= 100 ;
        }
        write  sum  ;
}
```

- output3

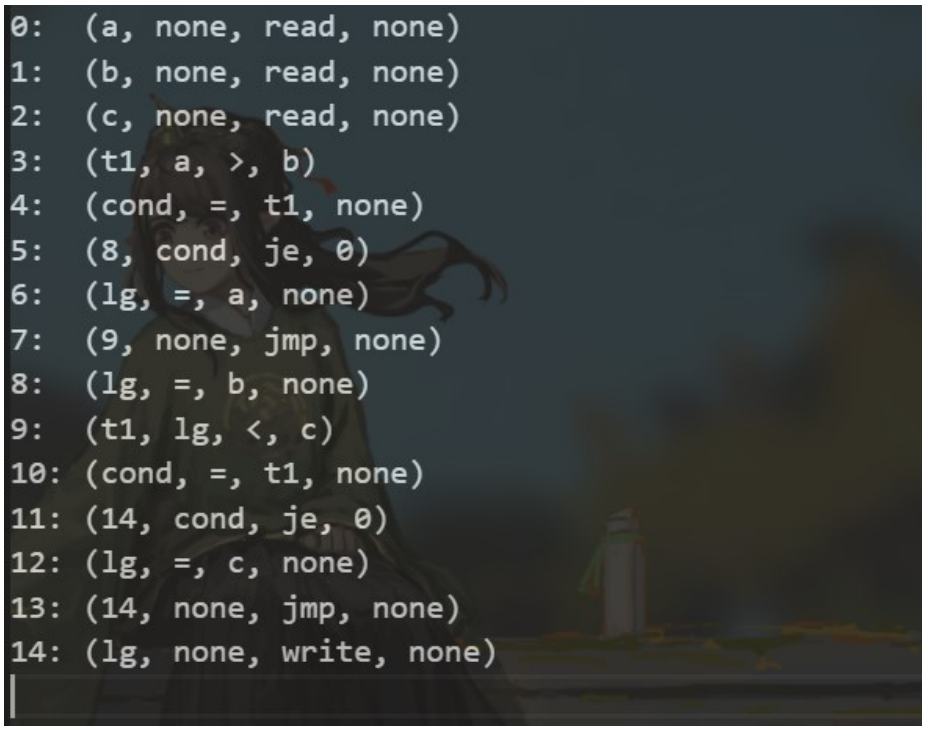


3.4 样例 4

- input4

```
//program 4: input 3 numbers, find the largest
//one, and output it .
{
        int a,b,c;
        int lg;
        bool cond;

        read  a;        read  b;        read  c;

        cond := a > b  ;
        if  cond  then  lg = a ;
        else  lg = b ;

        cond := lg < c ;
        if  cond  then  lg = c ;

        write lg  ;
}
```

- output4

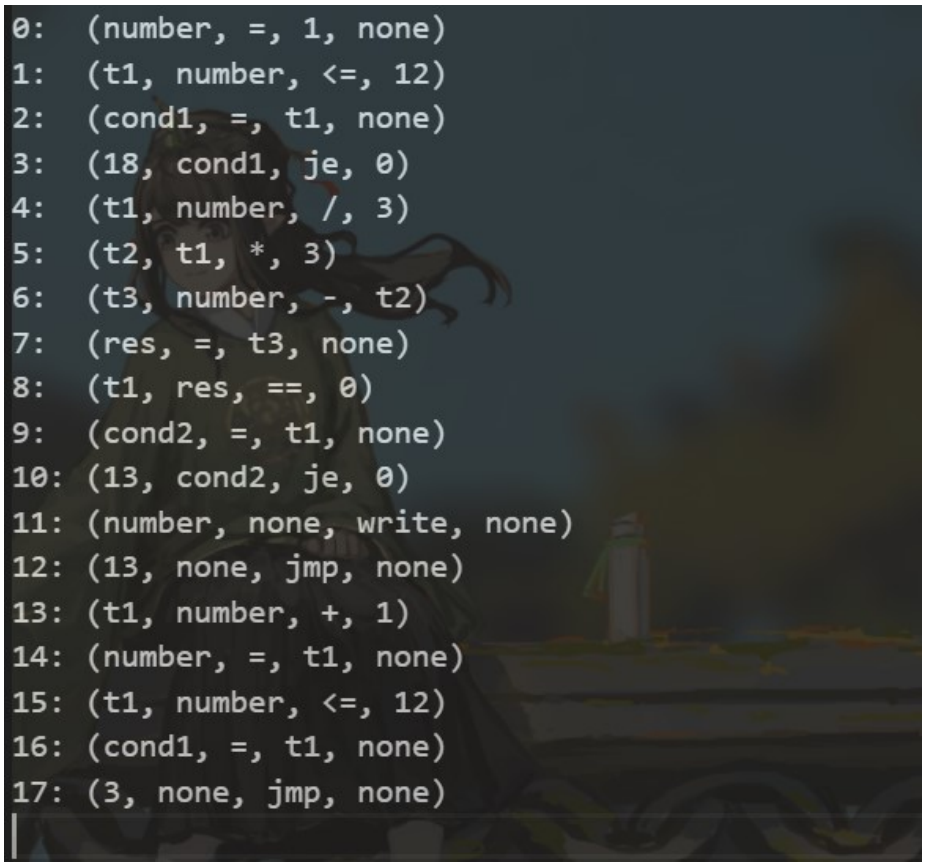


3.5 样例 5

- input5

```
/* program 5: find all numbers which is
 * divisible by 3  between 1 and 12 .
 */
{
        int  number, res ;
        bool  cond1, cond2  ;

        number = 1 ;
        cond1 := number <= 12 ;
        while cond1 do
        {
                res = number - ( number / 3 ) * 3 ;
                cond2 := res == 0  ;
                if cond2 then write number ;
                number = number + 1 ;
                cond1 := number <= 12 ;
        }
}
```

- output5



3.6 样例 6

- input6

```
//program 6: no decls in program .
//NO errors in Syntax Checking.
//Undeclared id errors should be reported in Symantic Checking.
{
        a = -1 ;
        b = a - 1;
        wirte a;
}
```

- output6

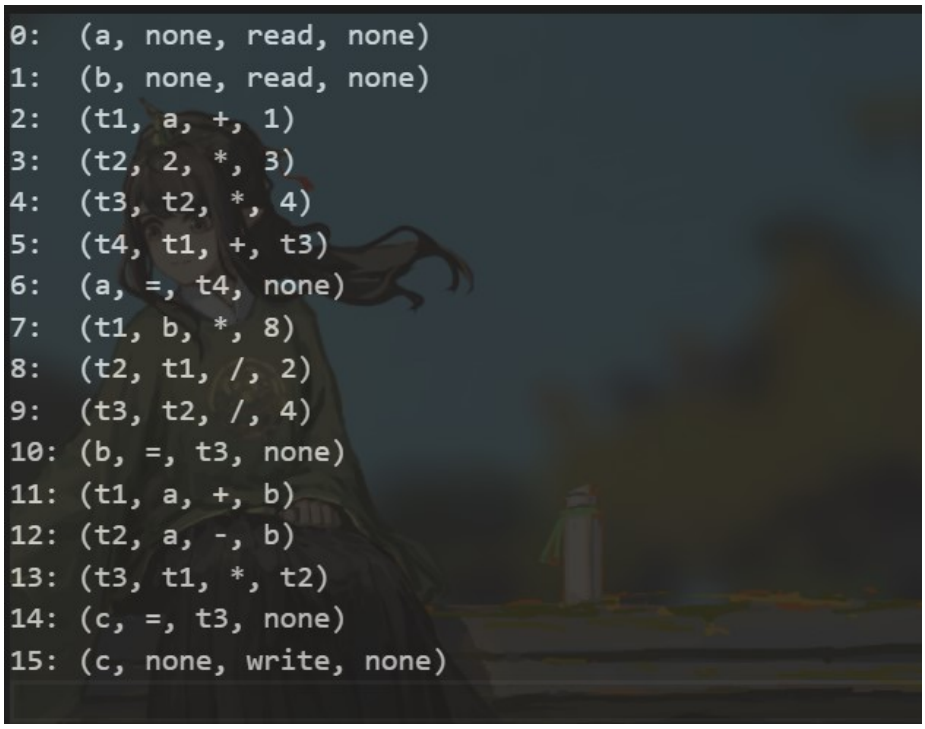
```
欢迎使用词法分析器，请选择输入方式:
1:从文件中输入  2:从终端中输入  3:退出
1
<{, delimiter>
<a, identifier1>
<=, op>
<-1, integer>
<;, delimiter>
<b, identifier2>
<=, op>
<a, identifier1>
<-, op>
<1, integer>
<;, delimiter>
<wirte, identifier3>
<a, identifier1>
<;, delimiter>
<}, delimiter>
未声明标识符a
PS E:\C++\compilation_principle>
```

3.7 样例 7

- input7

```
//program 7: read  in two numbers and do some calculations.
{
        int a, b, c  ;
        read a ;
        read b ;

        //Test Expr().
        a = a + 1 + 2 * 3 * 4 ;
        b = b * 8 / 2 / 4 ;
        c = ( a + b ) * ( a - b ) ;

        write c  ;
}
```

- output7

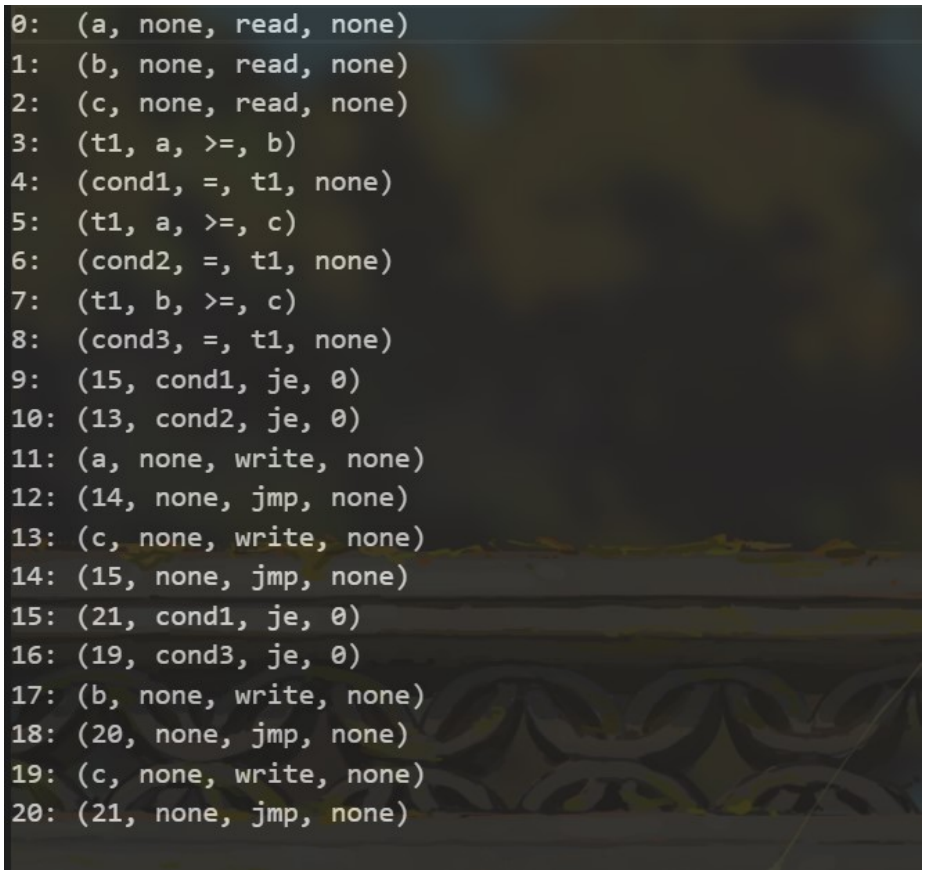


3.8 样例 8

- input8

```
/* program8: input three numbers ,output the largest one.
 * Test Statement : If_then and If_Then_Else .
 */
{
        int a,b,c ;
        bool cond1,cond2,cond3;

        read  a;  read  b;  read  c;

        cond1 := a >= b ;
        cond2 := a >= c ;
        cond3 := b >= c ;

        if cond1 then
                if cond2 then write a ;
                else write c ;
        if cond1 then
                if cond3 then write b ;
                else write c ;
}
```

- output8


0: (a, none, read, none)
1: (b, none, read, none)
2: (c, none, read, none)
3: (t1, a, >=, b)
4: (cond1, =, t1, none)
5: (t1, a, >=, c)
6: (cond2, =, t1, none)
7: (t1, b, >=, c)
8: (cond3, =, t1, none)
9: (15, cond1, je, 0)
10: (13, cond2, je, 0)
11: (a, none, write, none)
12: (14, none, jmp, none)
13: (c, none, write, none)
14: (15, none, jmp, none)
15: (21, cond1, je, 0)
16: (19, cond3, je, 0)
17: (b, none, write, none)
18: (20, none, jmp, none)
19: (c, none, write, none)
20: (21, none, jmp, none)