实验三 测试报告

20203231004 计科1班 李达良

本实验tiny语法规则

program->stmt-sequence

stmt-sequence->stmt-sequence;statement |statement

statement->if-stmt |repeat-stmt |assign-stmt | read-stmt| write-stmt | plusassign-stmt | for-stmt | regex-stmt

if_stmt-->if(exp) [stmt-sequence [[else stmt-sequence]]]（黄色的[]代表不是EBNF语法的[]）

repeat-stmt->repeat stmt-sequence until exp

assign-stmt->identifier := exp

plusassign-stmt ->identifier += exp

read-stmt->read identifier

write-stmt->write exp

for-stmt-->for identifier:=simple-exp to simple-exp do stmt-sequence enddo | for identifier:=simple-exp downto simple-exp do stmt-sequence enddo

exp -> exp orop orexp | orexp

orop -> |

orexp -> orexp andop andexp | andexp

andop -> &

andexp -> simple-exp comparison-op simple-exp | simple-exp

comparison-op -> < | > | <= | >= | <>

simple-exp -> simple-exp addop term | term

addop -> + | -

term -> term mulop notexp | notexp

mulop -> * | / | %

notexp -> notop (power | notexp) | power

notop -> ~

power -> power powop factor| factor

powop -> ^

factor -> (exp) | number | identifier

regex-stmt->identifier ::= regex_exp

regex_exp-> regex_exp rorop andreg | andreg

rorop -> |

andreg -> andreg randop topreg | topreg

randop -> &

topreg -> topreg topop | reg_factor

topop -> # | ?

reg_factor -> (regex_exp) | ideifier | number。

该报告最终结果：完美解析tiny程序的语法树

if-stmt

测试程序：

```
if(1)[write fact];
if(1)[write fact]
else[x:=1]
```

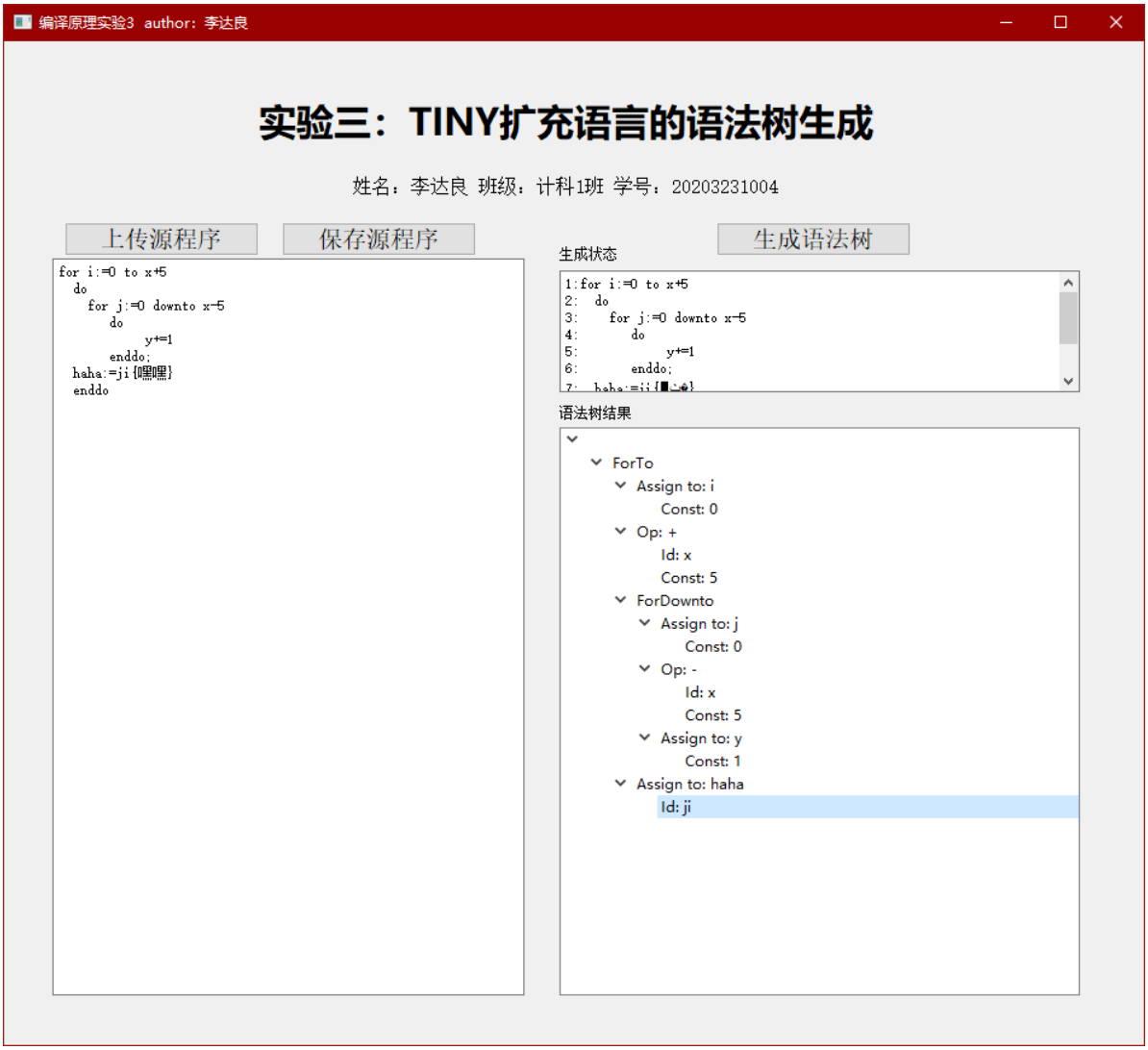
结果：

for

测试程序：

```
for i:=0 to x+5
  do
    for j:=0 downto x-5
       do
            y+=1
       enddo;
       haha:=ji{嘿嘿}
  enddo
```

结果：



+=

测试程序：

```
a += 2;
b += a;
c += a + b
```

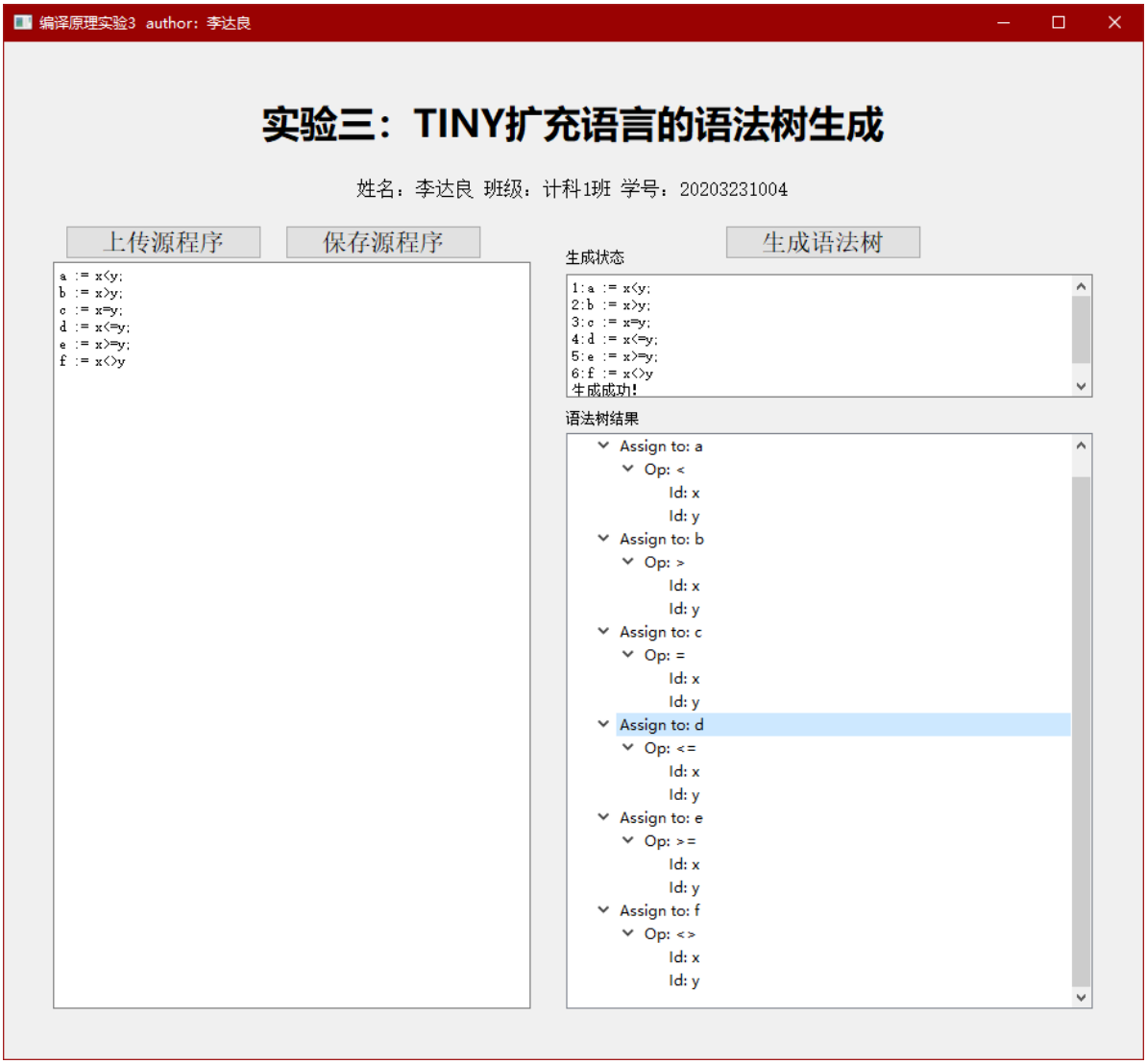
结果：

比较运算符

测试程序：

```
a := x<y;
b := x>y;
c := x=y;
d := x<=y;
e := x>=y;
f := x<>y
```

测试截图：

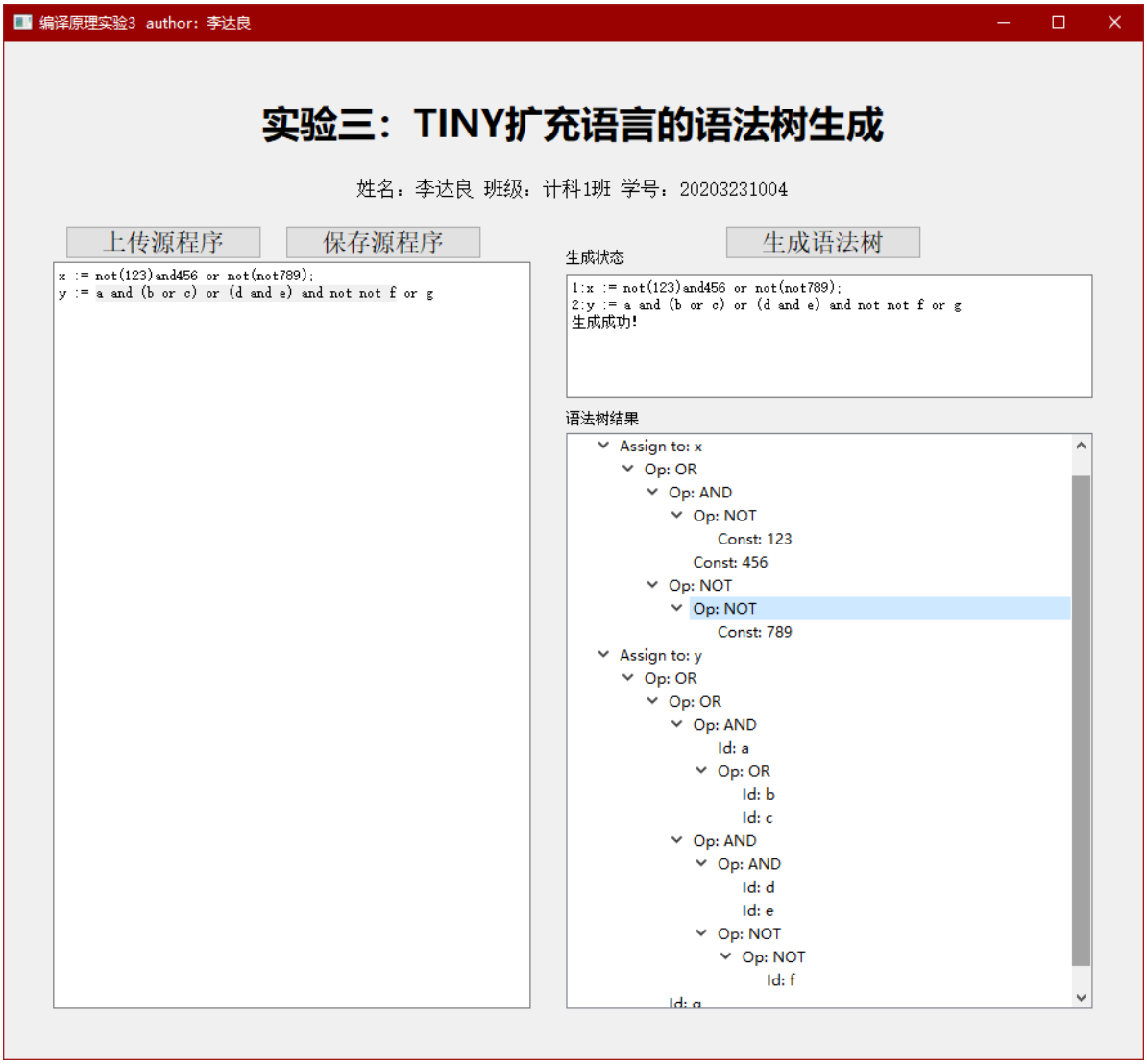


位运算

测试样例

```
x := not(123)and456 or not(not789);
y := a and (b or c) or (d and e) and not not f or g
```

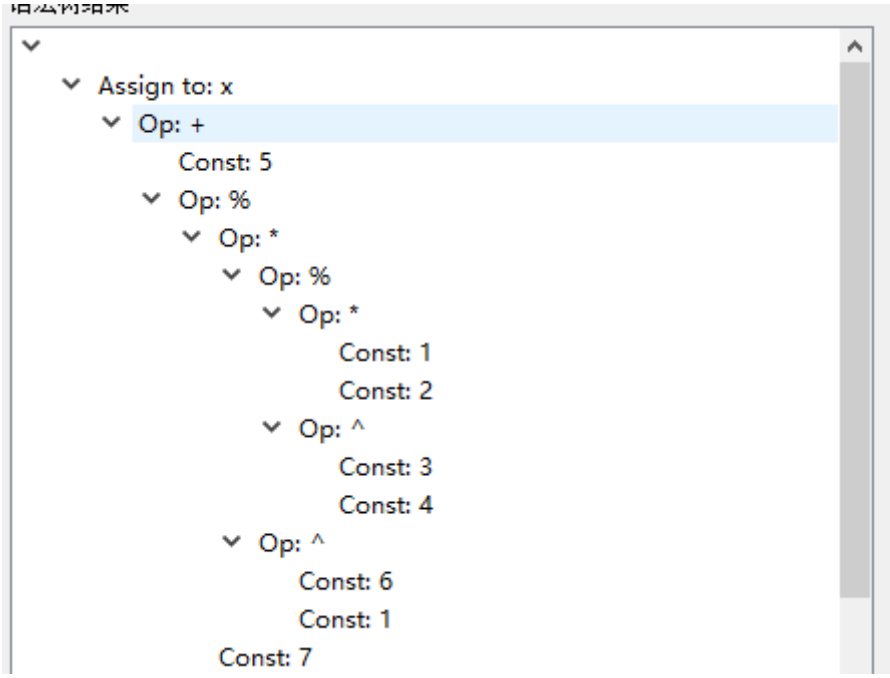
结果：

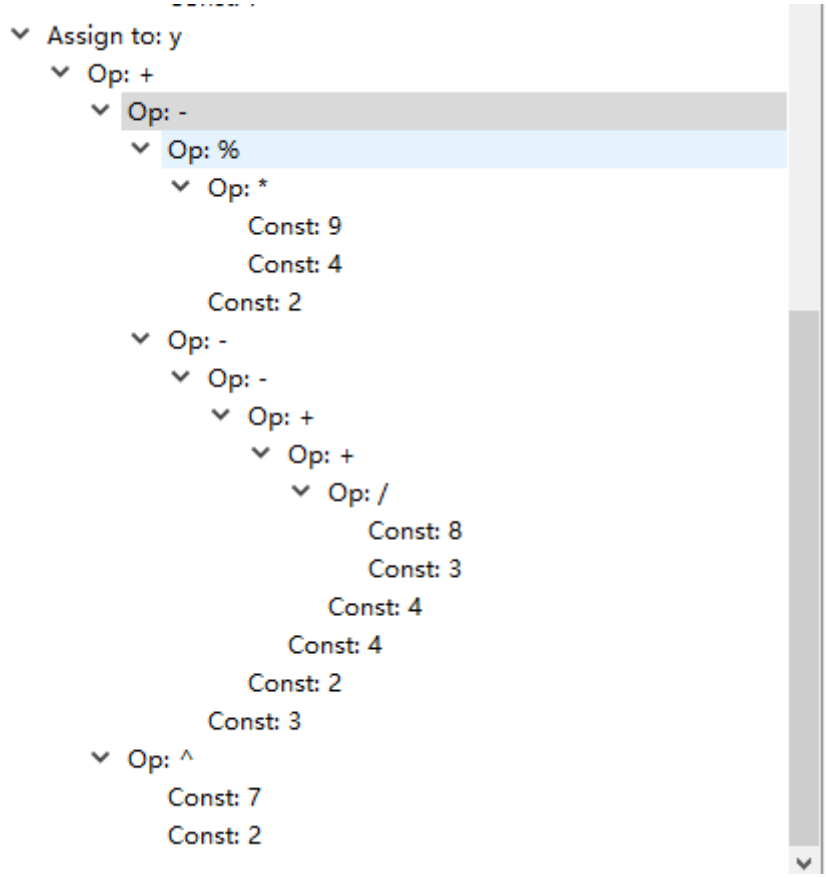


运算符号

```
x:=(5+(((1*2%3^4)))*6^1%7);
y:= 9*4%2-(8/3+4+4-2-3)+7^2
```

结果：



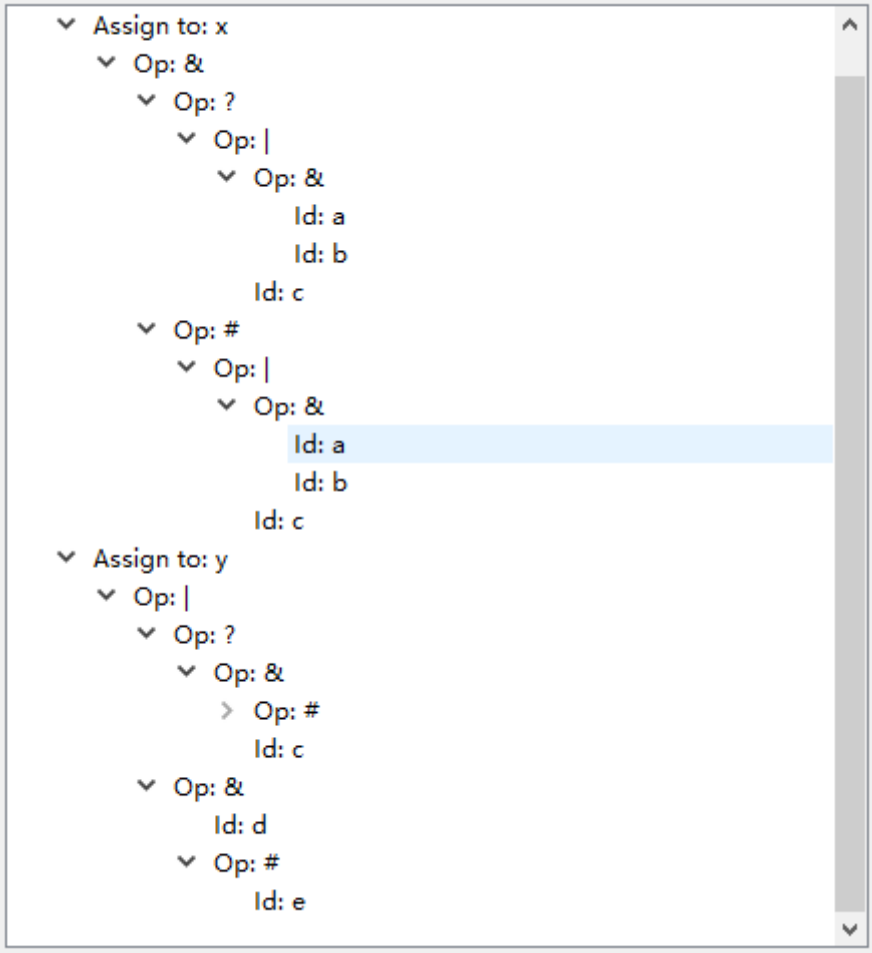


正则表达式

```
x::=(a&b|c)?&(a&b|c)#;
y::=((a&b)#&c)?|d&e#
```

结果：

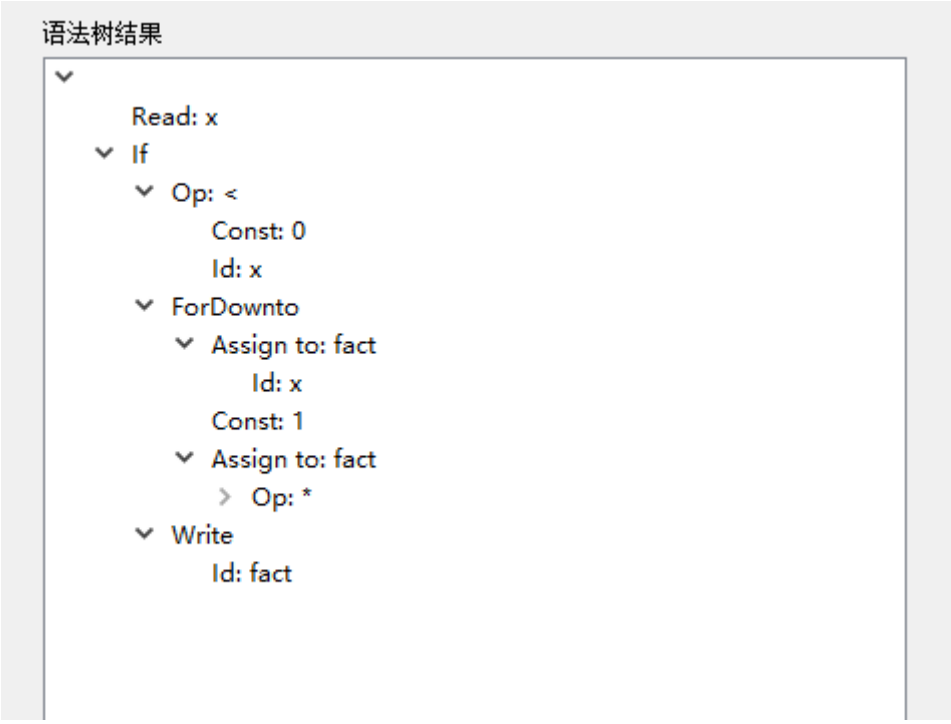
语法树结果



题目样例1

```
{ Sample program
  in TINY language -
  computes factorial
}
read x; { input an integer }
if  (0<x) { don't compute if x <= 0 }
  [for  fact := x downto 1 do
    fact := fact * x
  enddo;
  write fact]  { output factorial of x }
```

结果:



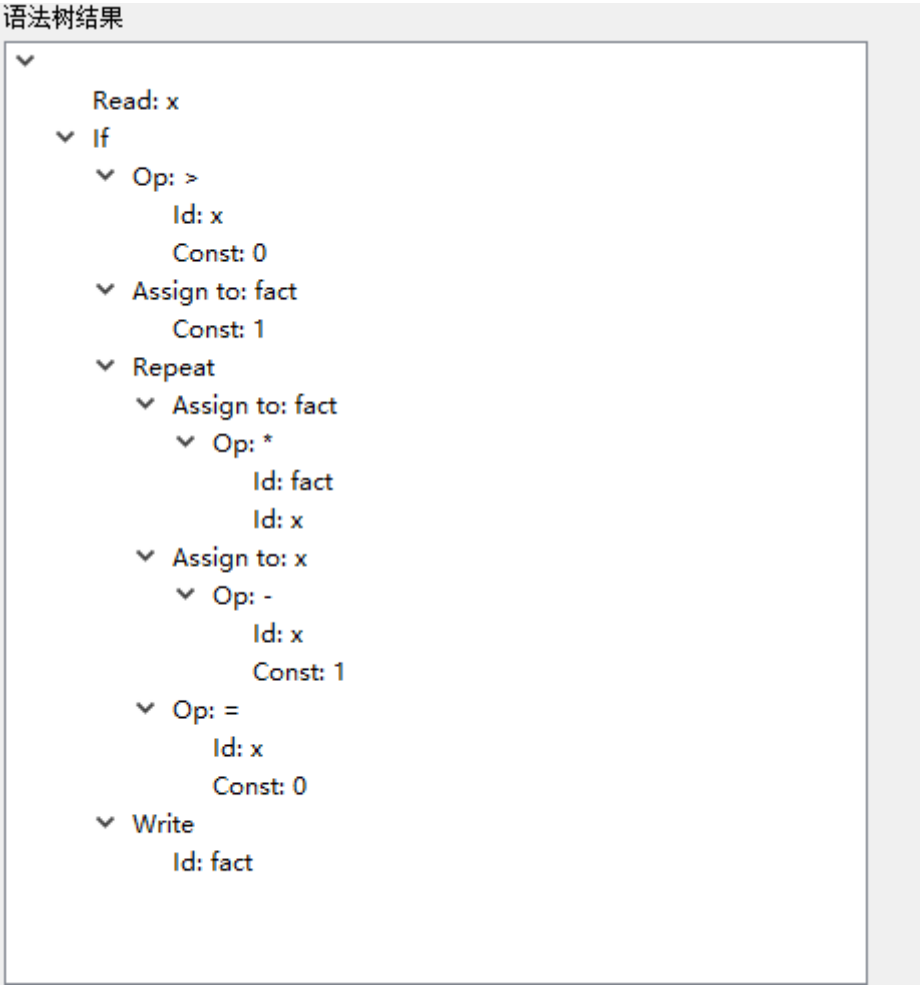
题目样例2

```
{ Sample program
  in TINY language -
  computes factorial
}
read x; { input an integer }

if  (x>0) { don't compute if x <= 0 }
  [fact := 1;
  repeat
    fact := fact * x;
    x := x - 1
  until x = 0;
  write fact  { output factorial of x }]
```

结果:

语法树结果



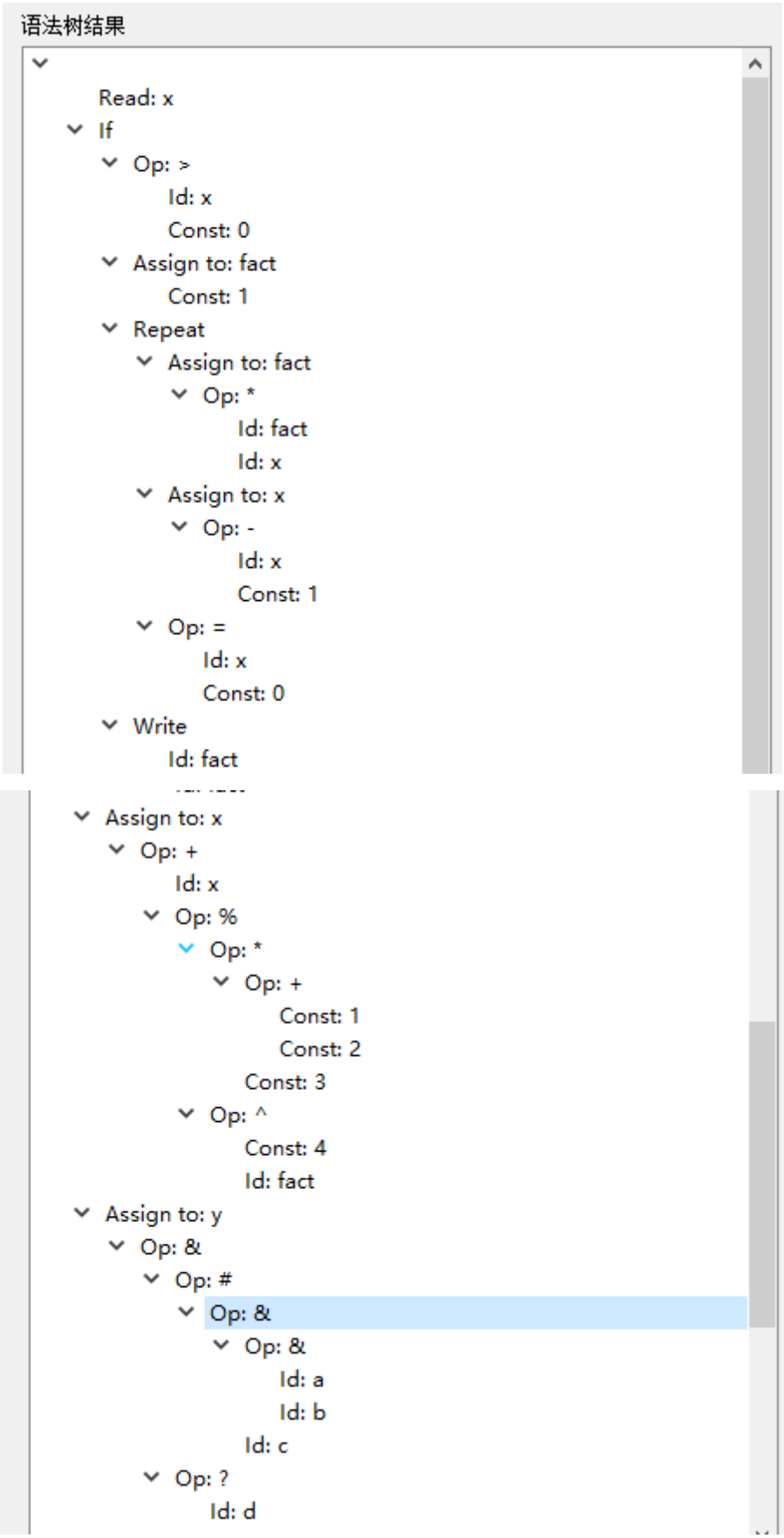
综合测试1

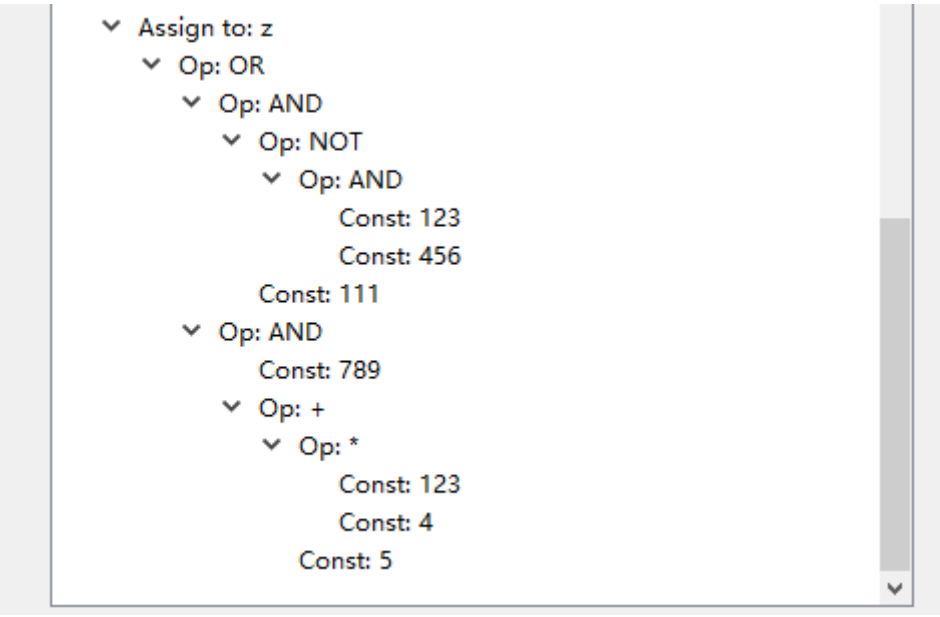
```
{ Sample program
  in TINY language -
  computes factorial
}
{test}
read x; { input an integer }

if (x>0) { don't compute if x <= 0 }
  [fact := 1;
  repeat
    fact := fact * x;
    x := x - 1
  until x = 0;
  write fact  { output factorial of x }
];
  x+=(1+2)*3%4^(fact);
  y::=(a&b&c)#&d?;
  z:= not (123 and 456 ) and 111 or 789 and 123 * 4 +5
```

结果:

语法树结果


Read: x
If
Op: >
Id: x
Const: 0
Assign to: fact
Const: 1
Repeat
Assign to: fact
Op: *
Id: fact
Id: x
Assign to: x
Op: -
Id: x
Const: 1
Op: =
Id: x
Const: 0
Write
Id: fact
Assign to: x
Op: +
Id: x
Op: %
Op: *
Op: +
Const: 1
Const: 2
Const: 3
Op: ^
Const: 4
Id: fact
Assign to: y
Op: &
Op: #
Op: &
Op: &
Id: a
Id: b
Id: c
Op: ?
Id: d



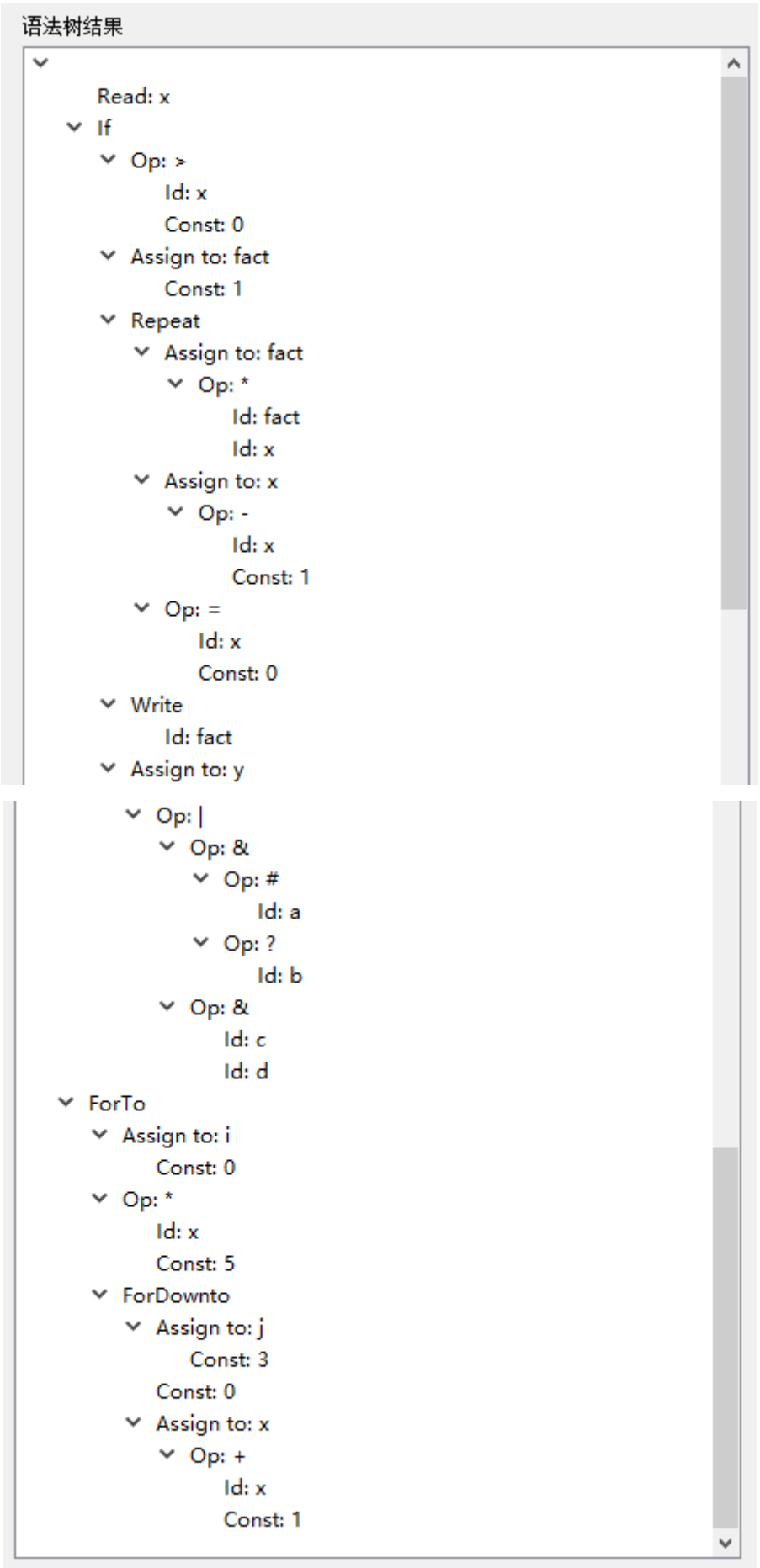
综合测试2

```
{ Sample program
  in TINY language -
  computes factorial
}
read x; { input an integer }

if  (x>0) { don't compute if x <= 0 }
  [fact := 1;
  repeat
    fact := fact * x;
    x := x - 1
  until x = 0;
  write fact]  { output factorial of x }
else
  [y::=a#&b?|c&(d)];{test}
for i := 0 to x*5
  do
for j :=3 downto 0
do
     x+=1{这里是两层的for循环}
enddo
enddo
```

结果:

语法树结果

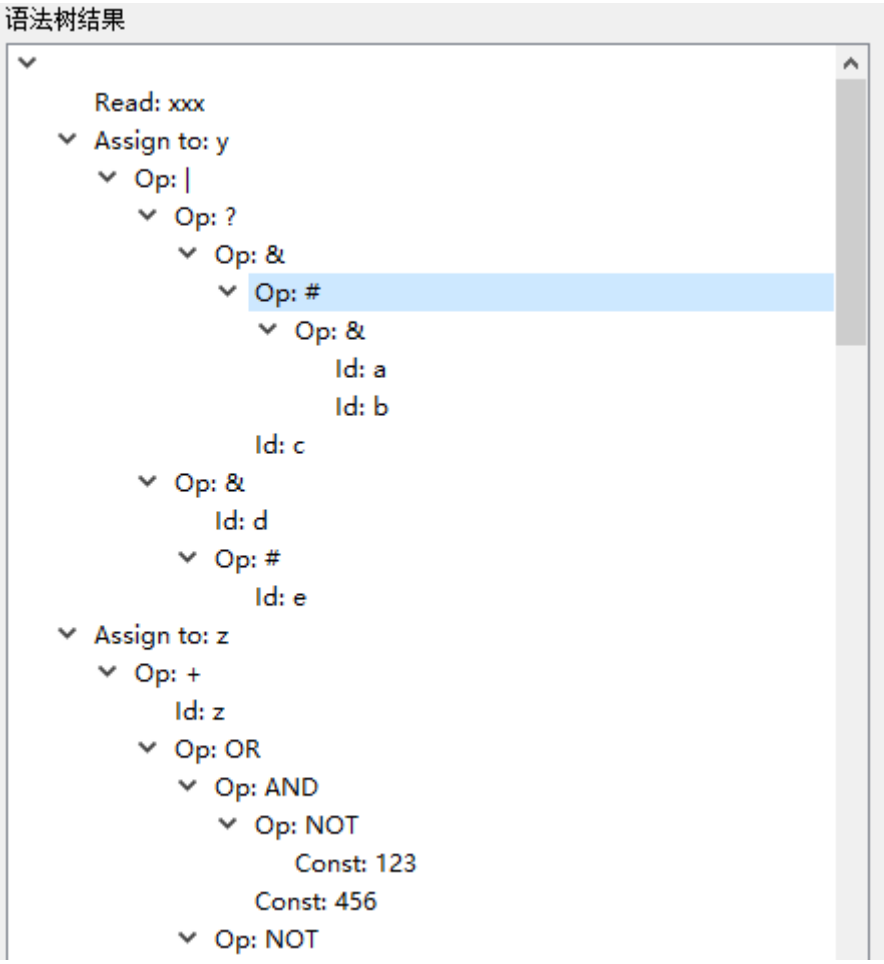


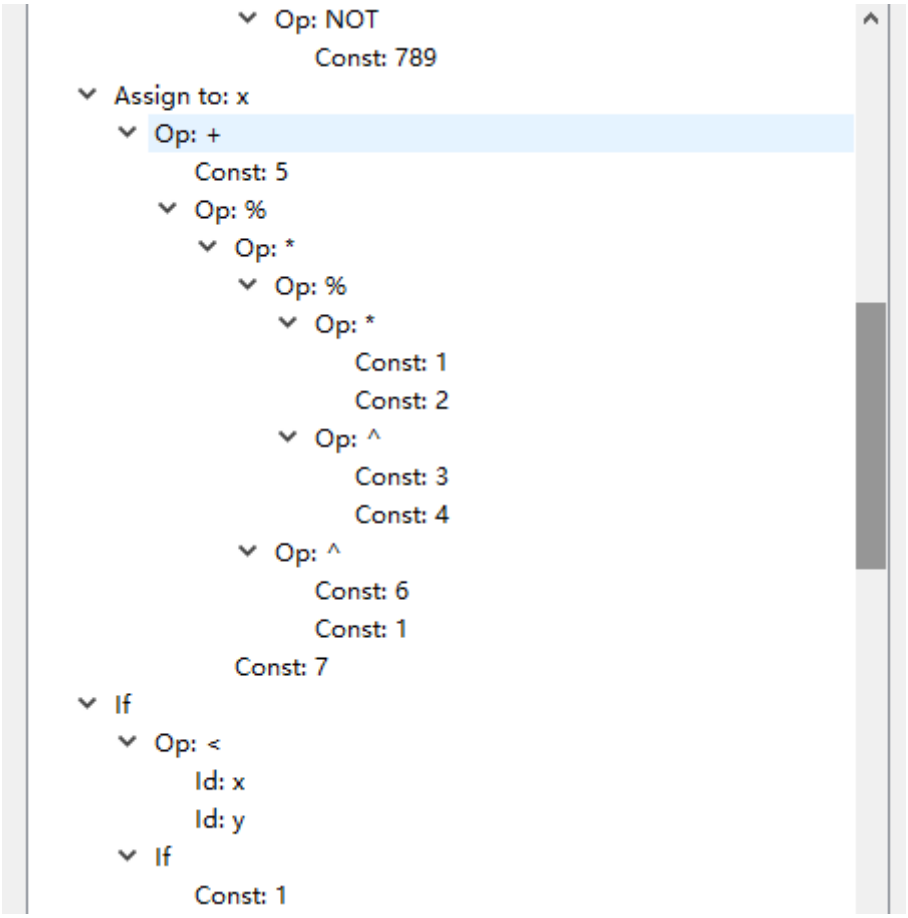
综合测试3

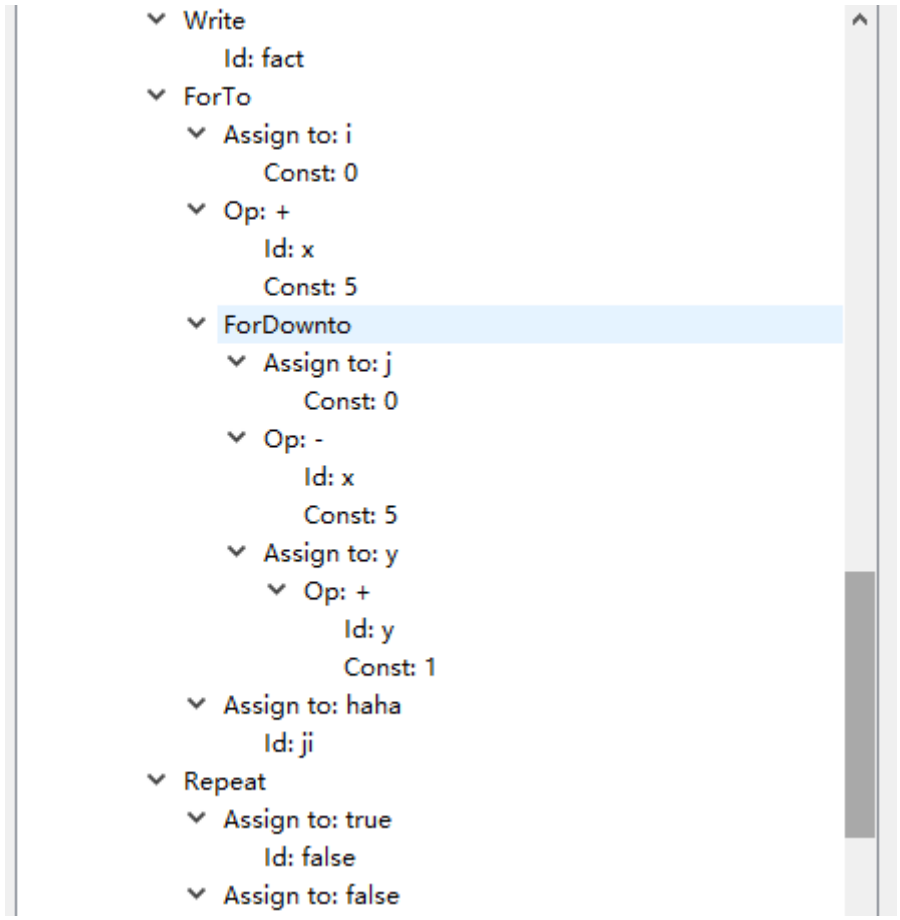
```
{预计这是一个很难的测试样例，你准备好了吗}
read xxx;{}
```

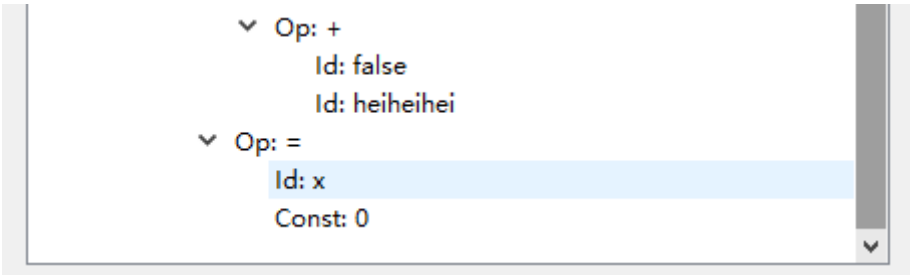
```
y::=((a&b)#&c)?|d&e#;
z+=not(123)and456 or not(not789);
x:=(5+(((1*2%3^4)))*6^1%7);
if (x<y)[
    if(1)
        [write fact]
    else
        [for i:=0 to x+5
            do
                for j:=0 downto x-5
                    do
                        y+=1
                    enddo;
                haha:=ji{嘿嘿}
            enddo;
        repeat
            true:=false;
            false+=heiheihei
        until x=0]
]
```

结果:




Op: NOT
Const: 789
Assign to: x
Op: +
Const: 5
Op: %
Op: *
Op: %
Op: *
Const: 1
Const: 2
Op: ^
Const: 3
Const: 4
Op: ^
Const: 6
Const: 1
Const: 7
If
Op: <
Id: x
Id: y
If
Const: 1


Write
Id: fact
ForTo
Assign to: i
Const: 0
Op: +
Id: x
Const: 5
ForDownto
Assign to: j
Const: 0
Op: -
Id: x
Const: 5
Assign to: y
Op: +
Id: y
Const: 1
Assign to: haha
Id: ji
Repeat
Assign to: true
Id: false
Assign to: false


Op: +
Id: false
Id: heiheihei
Op: =
Id: x
Const: 0