

测试数据 1:

```
read x; { input an integer }

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

文件(F) 分析(A) 帮助(H)

read x; { input an integer }
if x>0 then { don't compute if x <= 0 }
 fact := 1;
 repeat
 fact := fact * x;
 --x;
 until x = 0;
 write fact { output factorial of x }
end

词法分析结果

语法分析结果

语法树可视化

说明	文件	行	列
if语句缺少'	未命名	3	5
非法的表达式因子	未命名	3	6
if语句缺少'	未命名	3	7
赋值语句缺少':='	未命名	4	3
非法的表达式因子	未命名	4	8
期望语句	未命名	4	11
期望语句	未命名	10	1

按照规范修改后:

```
read x; { input an integer }

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

文件(F) 分析(A) 帮助(H)

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

词法分析结果 语法分析结果 语法树可视化

改变样式

program

stmt-sequence

read_stmt

identifier: x

if_stmt

condition

comparison: >

identifier: x

number: 0

then

stmt-sequence

assign_stmt

identifier: fact

number: 1

repeat_stmt

body

stmt-sequence

assign_stmt

identifier: fact

binary_op: *

identifier: fact

identifier: x

dec_stmt

operator: --

identifier: x

condition

comparison: =

identifier: x

number: 0

write_stmt

identifier: fact

测试数据 2:

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

文件(F) 分析(A) 帮助(H)

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

词法分析结果 语法分析结果 语法树可视化

说明	文件	行	列
语句缺少'	未命名	6	5
非法的表达式因子	未命名	6	6
语句缺少'	未命名	6	7
赋值语句缺少:='	未命名	7	3
缺少右括号	未命名	7	13
期望语句	未命名	7	16
赋值语句缺少:='	未命名	7	20
期望语句	未命名	7	27
多余的符号: write	未命名	9	3

语法分析完成, 发现9个错误

按照规范修改后:

```

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

```

The screenshot shows a compiler interface with a menu bar (文件(F), 分析(A), 帮助(H)) and a toolbar. The main window is divided into two panes. The left pane displays the source code of a 'Sample program' in TINY language, which computes a factorial. The right pane shows the '词法分析结果' (Lexical Analysis Results) tab, which displays a hierarchical parse tree for the input code. The tree structure is as follows:

- program
 - stmt-sequence
 - read_stmt
 - identifier: x
 - if_stmt
 - condition
 - comparison: <
 - number: 0
 - identifier: x
 - then
 - stmt-sequence
 - for_stmt
 - init
 - for_init
 - identifier: fact
 - number: 1
 - condition
 - comparison: >
 - identifier: x
 - number: 0
 - update
 - dec_stmt
 - operator: --
 - identifier: x
 - body
 - stmt-sequence
 - assign_stmt
 - identifier: fact
 - binary_op: *
 - identifier: fact

At the bottom of the interface, a status bar indicates '语法分析完成, 未发现错误' (Syntax analysis completed, no errors found).

测试数据 3:

```

read count;
result := 1;
pattern ::= (a|b)#;

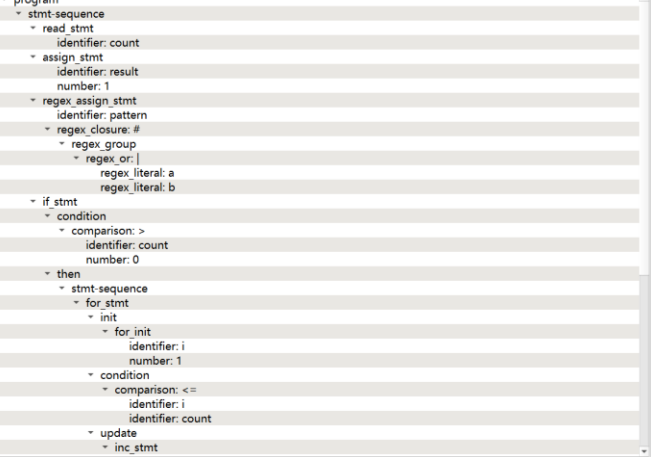
if(count > 0)
  for(i := 1; i <= count; ++i;)
    result := result * i;
    parity := i % 2;
  write result;
else
  write 0;

```

```
read count;
result := 1;
pattern := (a|b)#;

if(count > 0)
for(i := 1; i <= count; ++i)
    result := result * i;
    parity := i % 2;
write result;
else
write 0;
```

改变样式



语法分析完成，未发现错误