1. 实现改写书写格式的新if语句：

```

x:=1;

if (0<x)

   x:=x+1;

   x:= x\*x;

else

  x:=x+2;

  x:= x\*x\*x;

```

2. 增加for循环；

```

for fact := x downto 1 do

fact := fact \* x;

x:=x+1;

enddo;

for fact := 1 to x do

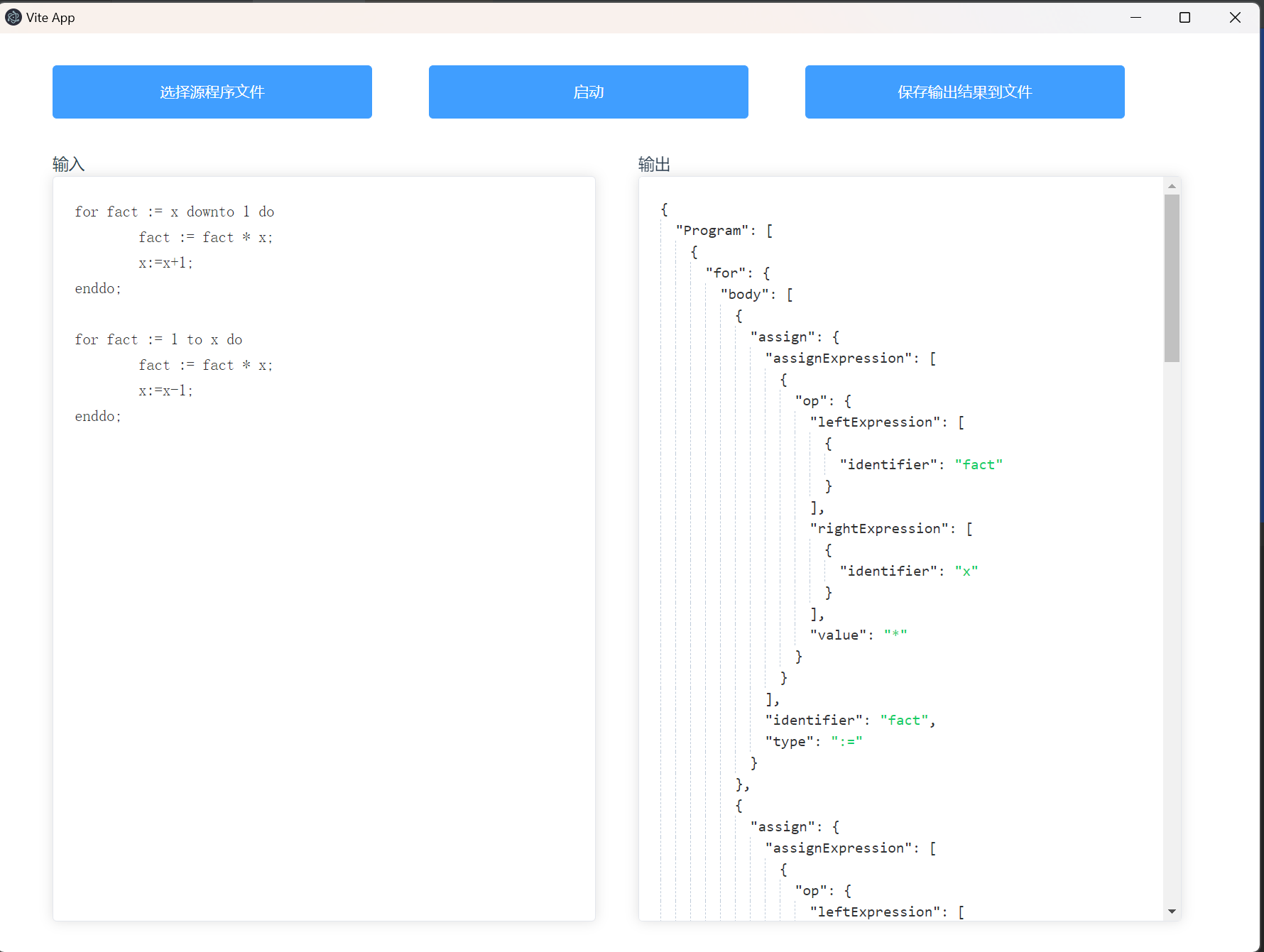
fact := fact \* x;

x:=x-1;

enddo;

```

这个功能测试的是对的，但是我代码实现里是要求enddo后面是需要加分号的，和给的测试代码有一点不一样，这里我稍微调整了一下测试代码。



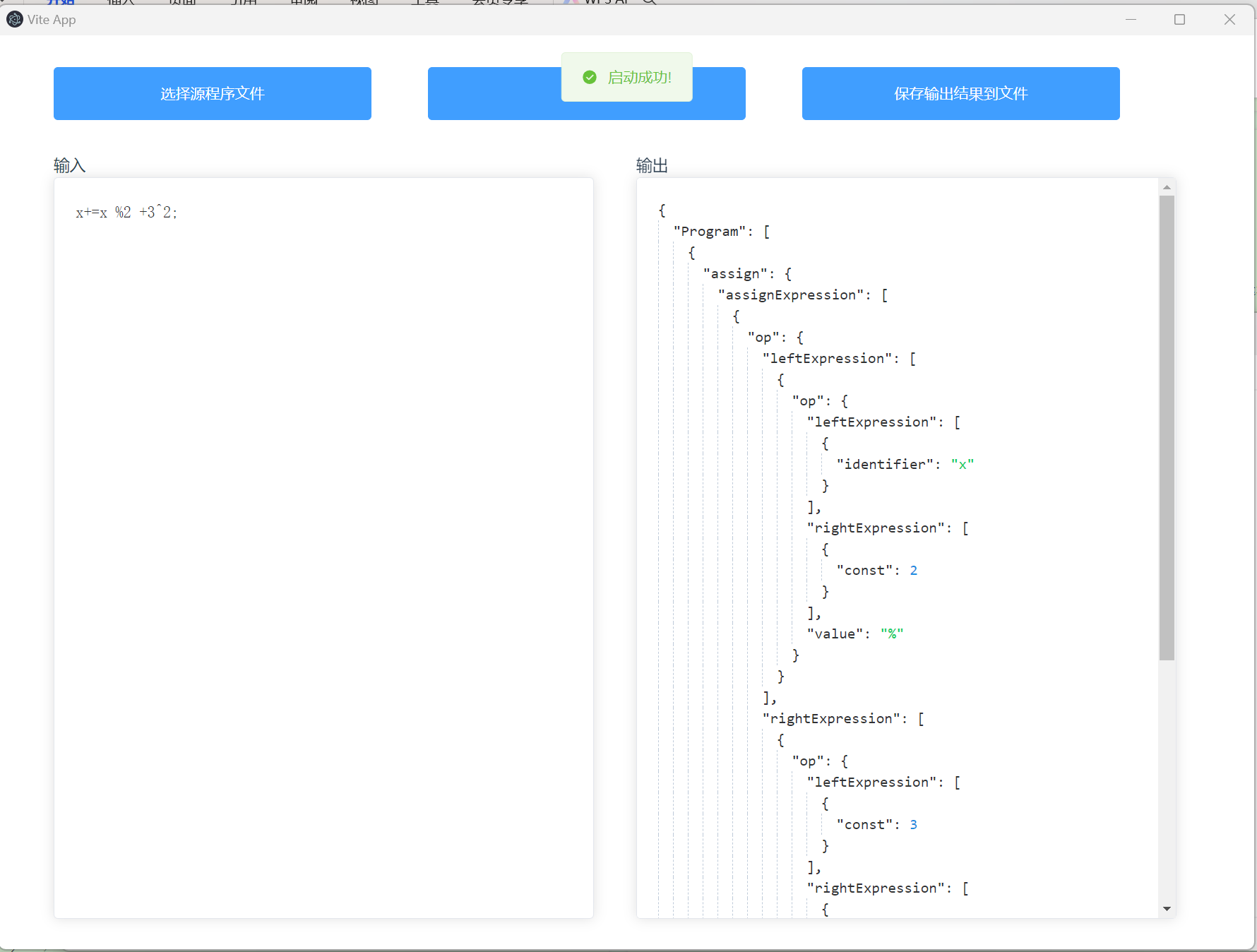
3. 扩充算术表达式的运算符号：+= 加法赋值运算符号（类似于C语言的+=）、求余%、乘方^，

```

x+=x %2 +3^2;

```

这个功能测试的也是正确的。



4. 扩充扩充比较运算符号：>(大于)、<=(小于等于)、>=(大于等于)、<>(不等于)等运算符号

```

if (x>0) x:=1;

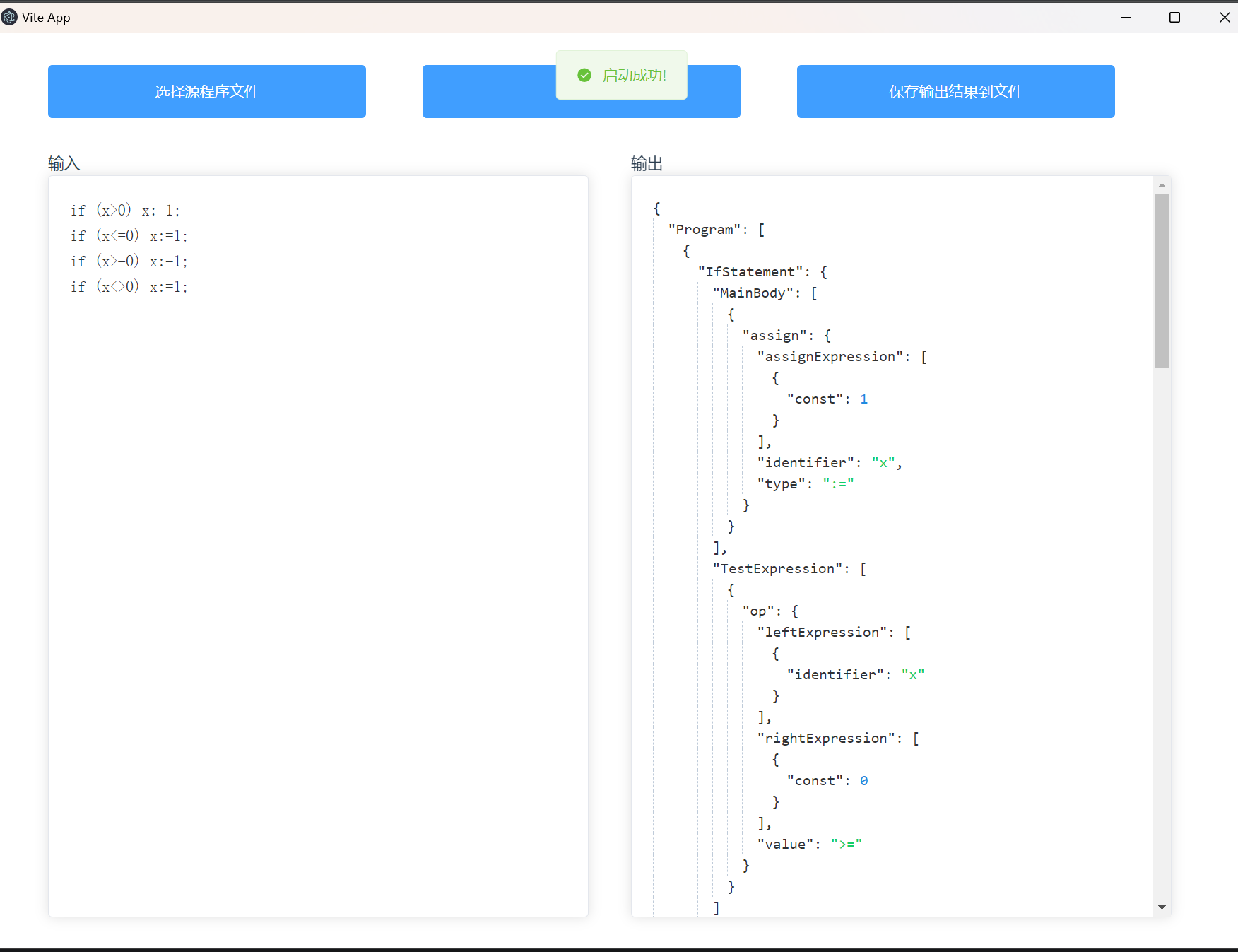
if (x<=0) x:=1;

if (x>=0) x:=1;

if (x<>0) x:=1;

```

这个功能测试无误。



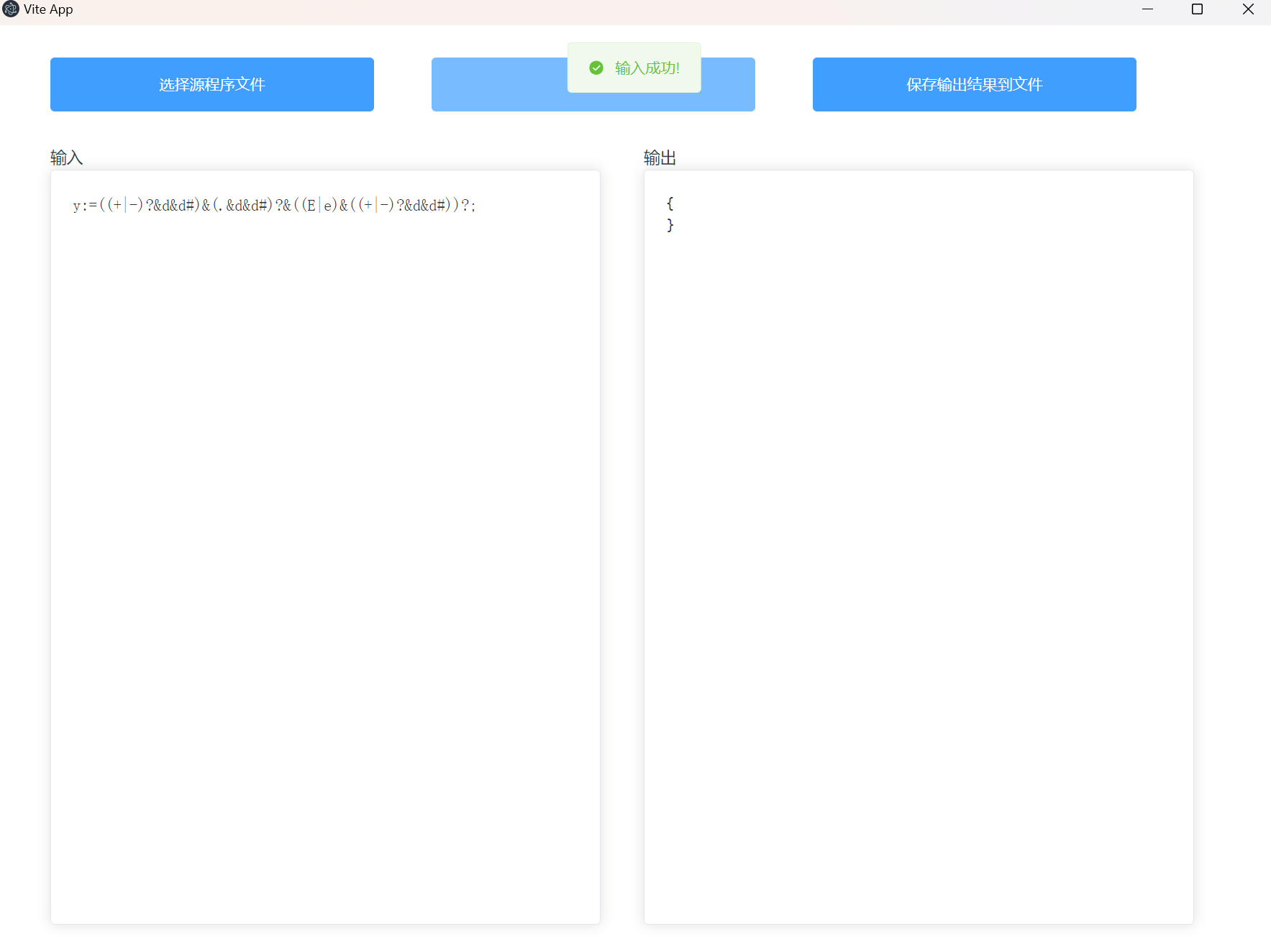
5. 增加正则表达式，其支持的运算符号有：  或(|)  、连接(&)、闭包(#)、括号( ) 、可选运算符号（？）和基本正则表达式。

```

y:=((+|-)?&d&d#)&(.&d&d#)?&((E|e)&((+|-)?&d&d#))?;

```

这个测试出现了问题，代码应该哪里是进入了死循环，无法显示结果到前端。

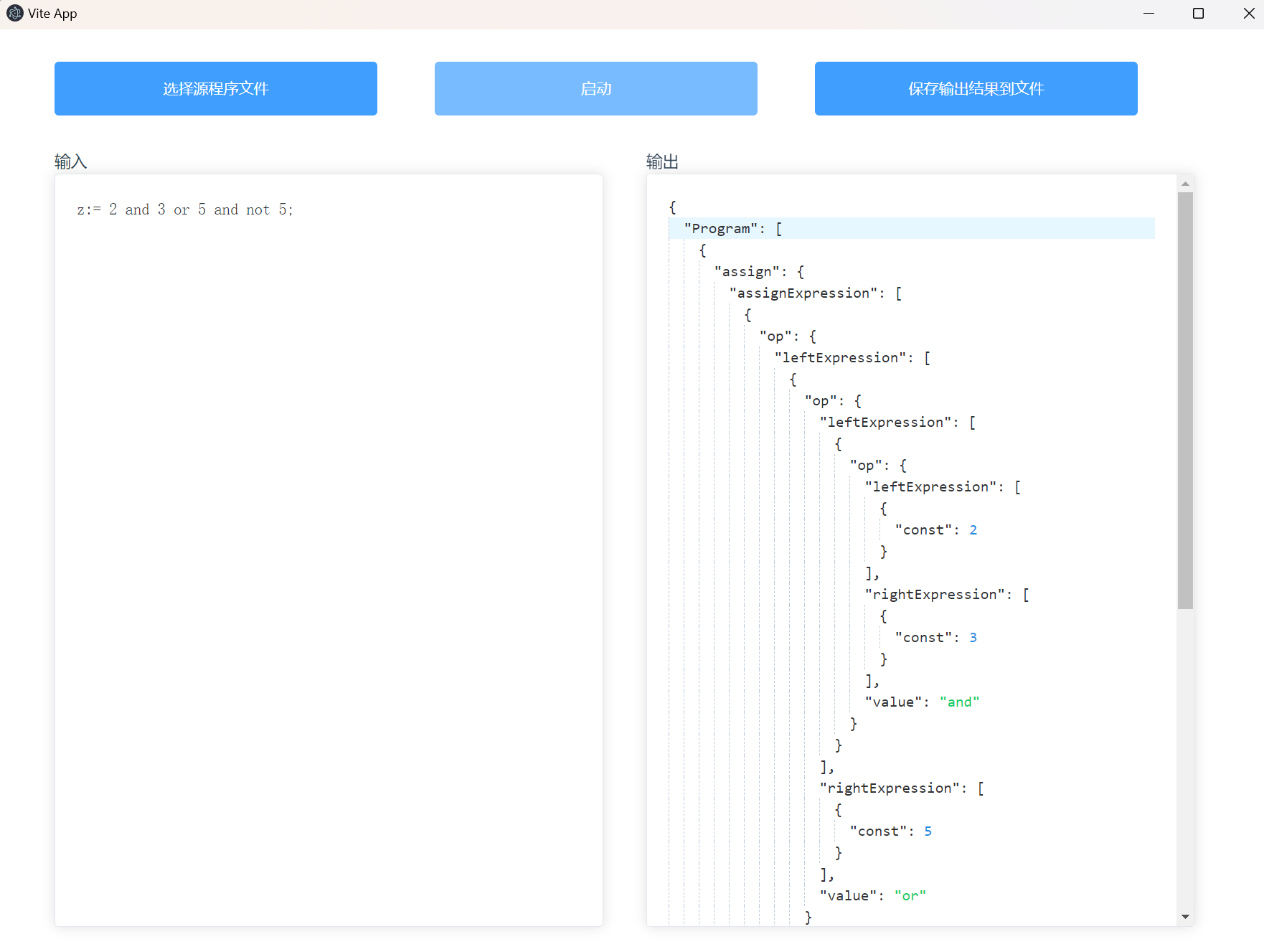


6. 增加位运算表达式，其支持的位运算符号有 and(与)、or（或）、 not(非），如果对位运算不熟悉，可以参考C/C++的位运算。

```

z:= 2 and 3 or 5 and not 5;

```  
这个功能测试的无误。



7. 混合功能测试

```

x:=5;

if( x>=1)

for fact := x downto 1 do

fact := fact \* x;

if (fact<>5) y:=10; else y:=20;

for z := 1 to y do

x:= 2 +3 and 3\*4 or 5%6^2 and not 5 \* (7+9);

x+=1-x

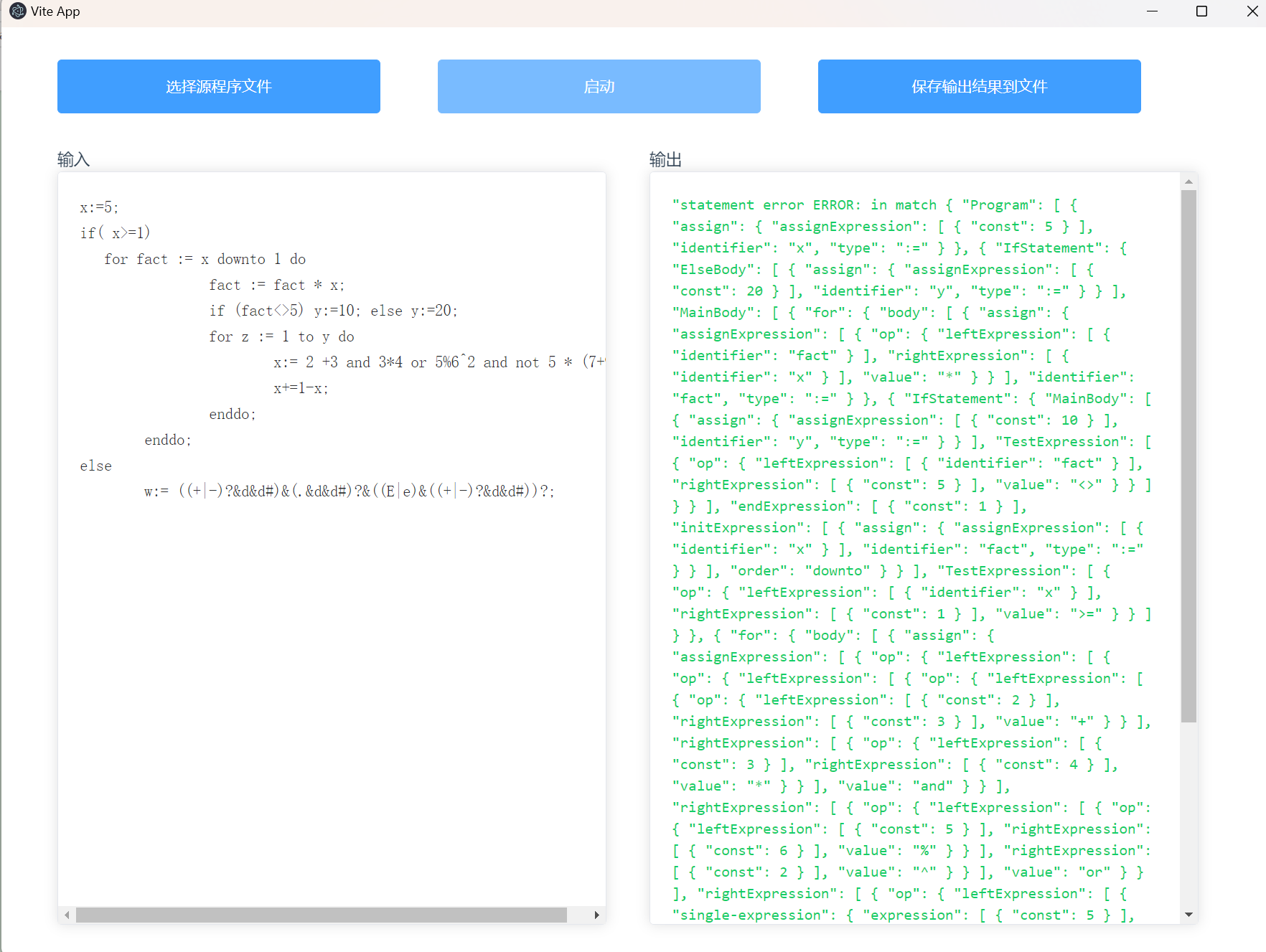
enddo;

enddo;

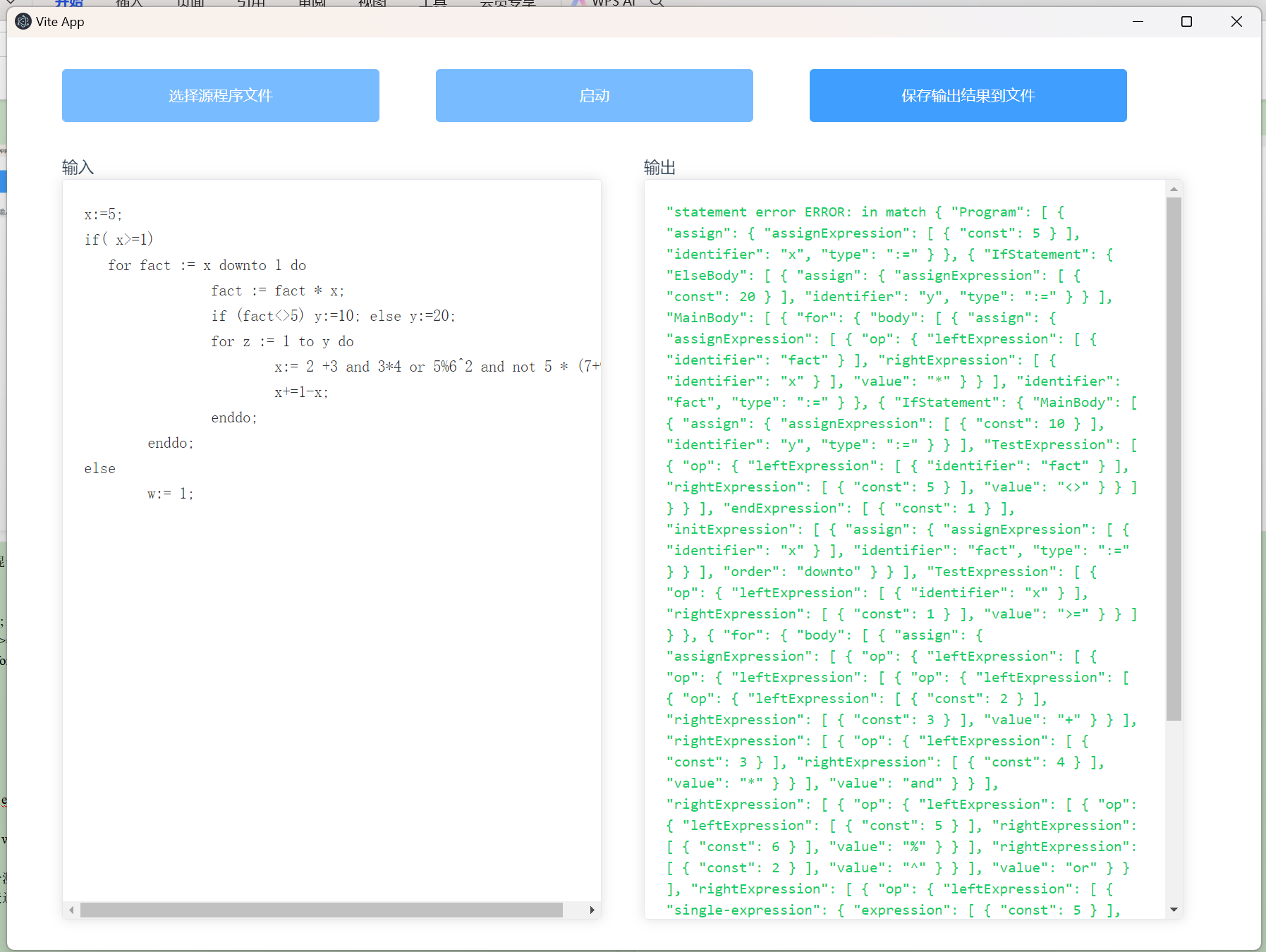
else

w:= ((+|-)?&d&d#)&(.&d&d#)?&((E|e)&((+|-)?&d&d#))?

```  
这个测试也出现了问题，前端显示了Error，返回的数据是异常的。应该还是我写的代码正则表达式判断的有问题。



但是如果去掉正则表达式那一行代码，发现还是有点问题。



经过debug源程序，发现是在if\_stmt函数实现里面出现了bug，我是考虑了有else的if，如果if没有else，但是我在if里面匹配了分号，但是外面在stmt\_sequence中也匹配了分号，这里就会出现重复匹配分号的问题。如果稍微改一下源程序在有else的时候if里面要匹配分号，没有else的时候if就不匹配结尾的分号就可以了。

下面是修改后的代码实现。

```

TreeNode \*stmt\_sequence() {

TreeNode \*t = statement();

TreeNode \*p = t;

while (currentTokenType != ENDFILE &&

currentTokenType != ELSE &&

currentTokenType != UNTIL &&

currentTokenType != ENDDO) { //标志代码段的结束

TreeNode \*q;

// 下面是第二次发现的if语句的bug，加上下面的if判断就解决了bug

if (currentTokenType != ELSE) {

match(SEMI);

}

q = statement();

if (q != nullptr) {

if (t == nullptr) {

t = p = q;

} else {

p->sibling = q;

p = q;

}

}

}

return t;

}

```

因此最后评分要扣除一个正则表达式测试的12分，然后最终混合测试中正则表达式的4分和其中一个if语句的6分，也就是自评分数为78分。