测试结果分析

测试1（正确）：

在此测试重点是在测试程序对var的语法语句能否识别，结果是正确，编译程序能正确判断它是var语句。

测试2（正确）：

在此测试重点是在测试程序对const的语法语句能否识别，结果是正确，编译程序能正确判断它是const语句。

测试3（正确）：

在此测试重点是在测试程序对Procedure的语法语句能否识别，结果是正确，编译程序能正确判断它是Procedure语句。

测试4（正确）：

在此测试重点是在测试程序对Begin和End的语法语句能否识别，结果是正确，编译程序能正确判断它是Begin和End语句。

测试5（正确）：

在此测试重点是在测试程序对write和read的语法语句能否识别，结果是正确，编译程序能正确判断它是write和read语句。

测试6（正确）：

在此测试重点是在测试程序对if和the的语法条件语句能否识别，结果是正确，编译程序能正确判断它是if和the的语法条件语句。

测试7（正确）：

在此测试重点是在测试程序对while和do的语法循环语句能否识别，结果是正确，编译程序能正确判断它是while和do的语法循环语。

测试8（错误）：

在此测试重点是在测试程序对赋值的语法语句能否识别，结果是错误，编译程序不能正确判断它赋值语句，这情况的原因是在设计算法的时候本人没考虑到这情况，所以会出现这错误，程序应该能够判断它是否赋值语句。

测试9（错误）：

在此测试重点是在测试程序对错误处理，结果是错误，编译程序只能正确判断符号的错误，但无法判断关键词的错误，比如是var 写成 vra 的错误，程序无法判断错误，这情况的原因是本人设计程序的时候没有考虑到这情况，在错误处理部分也没有条件判断这情况，程序应该会处理关键词/单词的错误。目前程序还需要在很多方面改进。

本人获取的文法是如此：

<程序>                    ::=  <分程序>.  
<分程序>               ::=  [<变量说明部分>][<常量说明部分>][<过程说明部分>]<复合语句>  
<常量说明部分>  ::=  const<常量定义>{,<常量定义>};  
<常量定义>           ::=  <标识符>=<无符号整数>  
<无符号整数>       ::=  <非零数字>{<数字>}|0  
<标识符>                ::=  <字母>{<字母>|<数字>}  
<变量说明部分>   ::=  var<标识符>{, <标识符>};  
<过程说明部分>   ::=  <过程首部><分程序>{;<过程首部><分程序>};  
<过程首部>            ::=  procedure<标识符>;  
<语句>                    ::=  <赋值语句>|<条件语句>|<当循环语句>|<过程调用语句>|<复合语句>|<读语句>|<写语句>|<空>   
<赋值语句>           ::=  <标识符> := <表达式>  
<表达式>               ::=  [+|-]<项>{<加法运算符><项>}    //[+|-]只作用于第一个<项>

<项>                        ::=  <因子>{<乘法运算符><因子>}  
<因子>                    ::=  <标识符>|<无符号整数>|'(' <表达式>')'  
<加法运算符>       ::=  +|-  
<乘法运算符>       ::=  \*|/  
<条件>                    ::=  <表达式><关系运算符><表达式>|odd<表达式>  
<关系运算符>       ::=  =|<>|<|<=|>|>=  
<条件语句>           ::=  if<条件>then<语句>  
<当循环语句>       ::=  while<条件>do<语句>   
<过程调用语句>  ::=  call<标识符>   
<复合语句>           ::=  begin<语句>{;<语句>}end  
<读语句>               ::=  read'('<标识符>{, <标识符>}')'   
<写语句>               ::=  write'('<表达式>{, <表达式>}')'  
<字母>                    ::=  a|b|c|d...|x|y|z|A|B...|Z   
<数字>                    ::=  0|<非零数字>  
<非零数字>           ::=  1|2|3...|8|9

下面是程序的测试分析结果：



