递归下降子程序分析法

E21514033

何铭春

1.源代码：

import sys

#spring为一个列表，保存了要分析的句子

global spring

spring=[]

# spring=['(', 'a', ',', 'a', ')']

global lookahead

# lookahead=spring[0]

#counter用于控制列表的访问

global counter

counter=1

#flag用于判断是否一个句子完全被分析，例如aa可能在判断开头的“a”之后就退出了，不会输出error

global flag

flag=0

#当最后一个句子元素被MatchToken那就输出success，并且flag=1表示整个句子都被分析

def getToken():

global lookahead

global counter

global spring

global flag

if counter==len(spring):

flag=1

print("success")

else:

lookahead=spring[counter]

counter=counter+1

#错误那就退出，不错那就取得下一个字符

def MatchToken(word):

if lookahead!=word:

print("error!")

sys.exit()

else:

getToken()

#S的分析程序

def ParseS():

print("S->",end='')

if lookahead=="a":

MatchToken("a")

elif lookahead=="^":

MatchToken("^")

elif lookahead=="(":

MatchToken("(")

ParseT()

MatchToken(")")

else:

print("error")

sys.exit()

#T的分析程序

def ParseT():

print("T->",end='')

ParseS()

ParseT7()

#T'的分析程序

def ParseT7():

if lookahead==",":

MatchToken(",")

ParseS()

ParseT7()

elif lookahead==")":

pass

else:

print("error")

sys.exit()

springStr=input("请输入要分析的句子(不输入使用默认句子：(a,a))：")

if springStr=="":

spring=['(', 'a', ',', 'a', ')']

else:

for i in springStr:

spring.append(i)

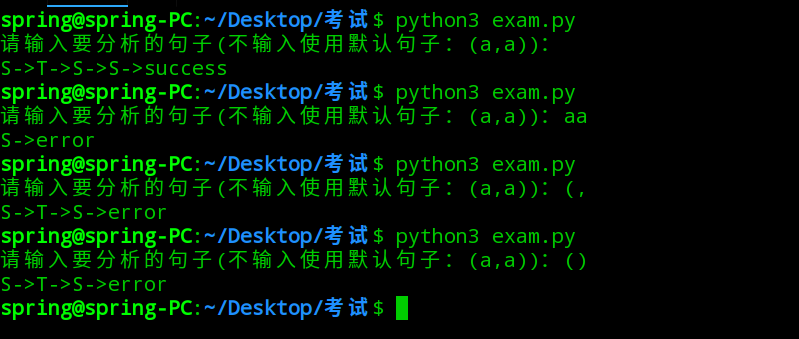
lookahead=spring[0]

ParseS()

if flag==0:

print("error")

2.实验截图：



3.实验心得

①利用列表保存要分析的句子元素，不同于字符串，不受类似a’这种占两个字符位，但是算作一个句子元素影响。

②本程序设定的结束条件是，最后一个句子元素被MatchToken到，MatchToken会调用getToken函数，访问下一个句子元素。此时判断是否还有下个句子元素，如果没有那句子就符合语法。不是最后一个元素，那就读取下一个。

③按照书上写完程序后，测试时发现输入aa可以通过程序，分析后发现存在S->a的语法，输入a开头的任何句子都会通过程序。我尝试在ParseS中添加检查下一个句子元素是否存在的代码，但是那样的话正常的句子也会收到影响。我最后添加了一个标志flag，flag在程序一开始运行时被置为0。如果经过完整的分析成功输出，那么flag会被置为1。而类似aa的句子则因为flag==0使程序报错。