한국외국어대학교
HANKUK UNIVERSITY OF FOREIGN STUDIES

프로그래밍 과제 3

HUFS

■ 담 당 교 수 : 김희철 교수님
■ 제 출 일 : 4/01
■ 학 과 : 컴퓨터전자시스템공학부
■ 학 번 : 201902520
■ 성 명 : 이상윤

contents

문제 기술

1. 한 줄로 입력된 괄호들 (,), {, }, [,]에 대하여, 모든 괄호가 짝이 맞는 경우 1을 출력하고 그렇지 않으면 0을 출력한다.

입력 예 1	출력 예 1
()[{a b}]	1

입력 예 2	출력 예 2
()[{bb a}] {()]()	0

입력 예 3	출력 예 3
()[{a b c}]) e {()]()	0

입력 예 4	출력 예 4
[()]{[{a}] e f {()}()	0

2. 한 줄로 입력된 괄호들 (,), {, }, [,]에 대하여, 모든 괄호가 짝이 맞는 경우 1을 출력하고 그렇지 않으면 괄호검사 알고리즘에 의하여 처음으로 발견된 (짝을 짓지 못한 괄호) 오류에 대하여, 입력 문자열에서 이 괄호의 위치(index)를 출력한다.

)에 대응하는 (가 없을 경우: error1
}에 대응하는 {가 없을 경우: error2
]에 대응하는 [가 없을 경우: error3
(에 대응하는)가 없을 경우: error4
{에 대응하는 }가 없을 경우: error5
[에 대응하는]가 없을 경우: error6

입력 예 1	출력 예 1
()[{a}]	1

입력 예 2	출력 예 2
()[{a}] {()]()	12 error3

입력 예 3	출력 예 3
()[{a}]) {()]()	7 error1

입력 예 4	출력 예 4
[()]{[{a}] {()}()	4 error5

3. 여러 줄로 력된 괄호들 (,), {, }, [,]에 대하여, 모든 괄호가 짝이 맞는 경우 1을 출력하고 그렇지 않으면 괄호검사 알고리즘에 의하여 처음으로 발견된 (짝을 짓지 못한 괄호) 오류에 대하여, 다음 중 하나를 출력한다

)에 대응하는 (가 없을 경우 error 1:) at position yy in line xx
}에 대응하는 {가 없을 경우 error 2: } at position yy in line xx
]에 대응하는 [가 없을 경우 error 3:] at position yy in line xx
(에 대응하는)가 없을 경우 error 4: (at position yy in line xx
{에 대응하는 }가 없을 경우 error 5: { at position yy in line xx
[에 대응하는]가 없을 경우 error 6: (at position yy in line xx

입력 예 1	출력 예 1
() [] (() a(b) (d{ e} g() [e fg)gh { }	error 1:) at position 4 in line 5

입력 예 2	출력 예 2
(())[{	error 5: { at position 6 in line 1

자료구조 및 알고리즘 설명

1. stack을 이용합니다. 먼저 stack을 사용하기 위해 클래스를 구현합니다. stack은 제일 나중에 들어온 값이 나갈 때 제일 먼저 나가듯이 괄호도 제일 나중에 열린괄호가 제일 먼저 닫히는 성질이 있습니다. 문자열의 문자들을 하나씩 비교해가면서 여는괄호를 stack par에 push하고 닫는괄호가 나왔을 때 par에 있던 여는괄호를 pop해서 둘이 짝이 맞는지 검사를 합니다(검사할 때 par이 비어있는 경우에는 닫는괄호에 대한 여는괄호가 앞에 없다는 뜻으로 0을 출력합니다). 검사가 끝난 후에는 par에 여는괄호가 남아있는지 확인합니다. 여는 괄호가 남아있지 않는 경우에는 짝이 다 맞다는 뜻이므로 1을 출력합니다. 만약 남아있다면 짝에 맞는 닫는 괄호가 없다는 뜻이므로 0을 출력합니다.

2. 1번문제와 원리는 같습니다. 오류 위치를 표기하기 위해 index값을 저장하는 stack(index)을 따로 만듭니다. 여는괄호를 발견했을 때 par에 push 해줄 뿐만 아니라 index값도 stack index에 pop해줍니다. error 1,2,3은 닫는괄호가 나왔지만 par에서 pop했던 여는 괄호가 닫는괄호와 짝이 맞지 않거나 par에서 pop해야할 여는 괄호가 없는 경우 닫는괄호의 모양에 따라 error1,2,3,이 출력됩니다. 이때 push했던 index값도 같이 pop해줍니다. 검사가 끝난 뒤 여는괄호 stack에서 남아있는 괄호가 있다면 괄호 모양에 따라 error4,5,6이 출력됩니다. 그리고 가장 늦게 열린 괄호가 가장 먼저 닫히는데 닫는괄호가 나오지 않았으므로 첫 오류는 가장 늦게 열린 괄호에서 발견하게 됩니다. 그래서 남아있는 par와 index에서 pop해서 괄호모양에 따른 에러와 에러위치를 출력합니다.

3. 여러줄을 입력받기 위해 while반복문을 사용했고 입력이 공백이면 루프를 끝내도록 하였습니다. 입력받은 문자열을 한줄씩 list 'sen'에 append하였고 실행시간을 단축시키기 위해서 input대신 sys.stdin.readline()메소드를 사용하였습니다. sen[1][3]이면 두 번째 문자열의 index값이 3인 문자를 얻을 수 있다는 것을 이용하여서 for문을 돌려 한글자씩 비교할 수 있게 구현했습니다. 또 몇번째줄 몇번째 글자인지 얻기위해 line,raw스택을 만들고 여는괄호를 par에 append할 때 같이 값을 각각 append하여 4,5,6에러에 대하여 에러를 출력할 때 에러 위치로 이용할 수 있도록 하였습니다. error1,2,3은 닫는괄호에서 발생하므로 그냥 현재 index값에 각각 1을 더하여 오류위치를 출력할 수 있고, error4,5,6은 여는괄호에서 발생하므로 line과raw에서 pop한 index값을 1씩 더해서 몇번째줄 몇번째 글자에서 오류가 발생했는지 출력하도록 하였습니다.

느낀 점

이번 과제에서 입력받은 문자열에서 괄호가 알맞게 짝을 이루어 열고 닫혔는지 검사하는 프로그램을 구현했습니다. 스택의 last in first out의 특성을 이용하여 이번 과제를 프로그래밍 할 때 효율적으로 프로그래밍할 수 있었습니다. 코드를 짜면서 어떤 자료구조를 이용하면 효율적으로 짤 수 있는지 또 자료를 기준에 맞추어 잘 정렬하여 효율적인 프로그램을 만드는 것의 중요성을 조금 알게 되는 것 같습니다. 앞으로는 좋은 코드를 짜기위해 많은 노력을 하겠습니다. 문제를 푸는 도중에 처음에는 오류가 여러개인 문자열을 입력하면 첫 오류는 스택의 젤 밑에서(제일 먼저 발견한 여는괄호) 발생할거라고 생각해봤지만 테스트를 여러번 거치면서 그것이 아니라 스택의 가장 위에서 발생한다는 것을 깨달았습니다. 3번문제에서 sys.stdin.readline()메소드를 사용하면 input()보다 실행속도가 빠르다는 것을 이해했습니다. 문자열을 입력하면 맨 뒤에 ₩n이 따라온다는 것도 보았습니다. 여러줄을 입력 받을 때 입력받은 줄이 몇줄인지도 모르는데 어떻게 반복문을 쓸까 의문을 갖고 있었는데 지인의 도움으로 입력값이 없다면 반복을 종료하는 방식으로 해결하려고 했습니다. 하지만 개행문자 때문에 무한루프가 돌고 rstrip('₩n')으로 개행문자를 제거하면 몇 개의 답이 틀렸다는 것을 보았습니다. 그래서 수요일 수업시간에 알려주신대로 sys.stdin.readlines()을 사용하여 여러줄을 입력받고 ctrl+d키로 입력을 끝낼 수 있는 방법을 알게되었습니다.

프로그램 코드

1.

```
class Stack :     #stack을 사용하기 위한 클래스  구현
   def __init__(self ):
     self .items =[]
   def isEmpty (self ):
     return len (self .items )==0
   def pop (self ):
     try :
       return self .items .pop ()
     except SyntaxError :
       print ("stack is empty")
   def push (self , item ):
     self .items .append (item )
str =input ()
def parentheses_bal (string ):
   par = Stack ()
   openPa ="([{"
   closePa =")]}"


   for ch in string :
     if ch in openPa :     #여는 괄호는 스택에 push
       par .push (ch )
     elif ch in closePa :
       if par .isEmpty ():   #닫는 괄호가 있는데 여는괄호가 앞에 없으면 0리턴
         return 0
       else :    #여는괄호가 있지만 닫는괄호가 여는괄호와 매칭이 되지 않는다면
0리턴
         opench =par .pop ()
         if ( ch ==')'and opench !='(' ) or ( ch =='}'and opench !='{' ) \
           or ( ch ==']'and opench !='[' ):
           return 0

   if par .isEmpty ():    #남아있는 여는괄호가 있는지 검사 스택이 비어있지 않
으면 짝이 맞지 않으므로 0리턴
     return 1
   else :
     return 0
print (parentheses_bal (str ))
```

2.

```
class Stack :     #stack을 사용하기 위한 클래스  구현
   def __init__(self ):
     self .items =[]
   def isEmpty (self ):
     return len (self .items )==0
   def pop (self ):
     try :
       return self .items .pop ()
     except SyntaxError :
       print ("stack is empty")
   def push (self , item ):
     self .items .append (item )
str =input ()
def parentheses_bal (string ):
   par = Stack ()
   openPa ="([{"
   closePa =")]}"
   index =Stack ()
   for i in range (len (string )):
     if string [i ] in openPa :   #여는괄호는 스택에 push
       par .push (string [i ])
        index .push (i )      #오류위치를 나중에 출력하기 위해 index스택에
index값 push
     elif string [i ] in closePa :
       if par .isEmpty ():    #닫는괄호가 나왔는데 앞에 여는괄호가 없으면 괄호
모양에 따른 에러 발생
         if string [i ]==')':
           return i , "error1 "
         if string [i ]=='}':
           return i , "error2"
         if string [i ]==']':
           return i , "error3"
       else :#닫는괄호가 있고 앞에 여는괄호가 있지만 모양이 다른경우 닫는괄호의
모양에 따른 에러발생
         opench =par .pop ()
         index .pop ()
         if ( string [i ]==')'and opench !='(' ):
           return i , "error1"
         if ( string [i ]=='}'and opench !='{' ):
           return i , "error2"
         if ( string [i ]==']'and opench !='[' ):
           return i , "error3"
   if par .isEmpty (): #남아있는 여는괄호가 있는지 검사 스택이 비었으면 짝이 맞
음
     i =1
     return i , ""
   else :   #여는괄호가 남아있으면 그에 맞는 닫는괄호가 없으므로 여는괄호의 모양
에 따라 에러발생. 스택에서 제일 나중에 들어온 여는괄호가 먼저 닫혀야 하는데 닫는
괄호가 없으니 첫 오류로 판단하고 해당 index값 pop
```

```
      i =index .pop ()
      p =par .pop ()
      if p =='(':
        return i ,"error4"
      elif p =='{':
        return i , "error5"
      elif p =='[':
        return i ,"error6"


num , error =parentheses_bal (str ) #출력값이 (a,b)이런식으로 나오므로 괄호빼기위한 출력방식
print (num ,error )
```

3.

```
import sys
class Stack :      #stack을 사용하기 위한 클래스 구현
   def __init__(self ):
     self .items =[]
   def isEmpty (self ):
     return len (self .items )==0
   def pop (self ):
     try :
       return self .items .pop ()
     except SyntaxError :
       print ("stack is empty")
   def push (self , item ):
     self .items .append (item )
sen =sys .stdin .readlines ()  #입력받을 여러줄의 문장 list생성 ctrl+D로 종료
   #sys.stdin.readline()메소드를 사용하면 실행시간을 줄일 수 있다
def parentheses_bal (list ):
   par =Stack ()
   openPa ='({['
   closePa =')}]'
   line =Stack ()
   raw =Stack ()

   for i in range (len (list )):    #i번째줄 문장의 j번째 단어가 여는괄호면 par스택에 push
     for j in range (len (list [i ])):
       if list [i ][j ] in openPa :
         par .push (list [i ][j ])
         line .push (i )     #오류위치를 출력하기 위해서 줄과 몇번째 글자인지 각각 line, raw스택에 push
         raw .push (j )
       elif list [i ][j ] in closePa : #닫는괄호가 나왔지만 앞에 여는 괄호가 없는경우 닫는괄호의 모양에따라 error출력
         if par .isEmpty ():
           if list [i ][j ]==')':
             return 'error 1: ) at position',j +1 , 'in line', i +1
```

```
            if list [i ][j ]=='}':
              return 'error 2: } at position',j +1 , 'in line', i +1
            if list [i ][j ]==']':
              return 'error 3: ] at position',j +1 , 'in line', i +1
          else :    #닫는괄호와 최근 여는괄호의 짝이 맞지않을 때 닫는괄호의 모양에 따른 error 출력
            opench =par .pop ()
            l =line .pop ()
            r =raw .pop ()
            if ( list [i ][j ] == ')'and opench !='(':
              return 'error 1: ) at position',j +1 , 'in line', i +1
            if ( list [i ][j ] == '}'and opench !='{':
              return 'error 2: } at position',j +1 , 'in line', i +1
            if ( list [i ][j ] == ']'and opench !='[':
              return 'error 3: ] at position',j +1 , 'in line', i +1
    if par .isEmpty ():   #짝이 맞을경우 1출력
      return 1 ,'','',''
    else :    #닫는괄호가 더이상 나오지 않는경우 남은 여는괄호중 가장 최근에 들어온 여는괄호가 젤 먼저 닫히지 않는 에러 발생하므로 line, raw 여는괄호모양 pop해서 출력
      l =line .pop ()
      r =raw .pop ()
      p =par .pop ()
      if p =='(':
        return 'error 4: ( at position', r +1 , 'in line', l +1
      elif p =='{':
        return 'error 5: { at position', r +1 , 'in line', l +1
      elif p =='[':
        return 'error 6: [ at position', r +1 , 'in line', l +1
error , raw , inline , line =parentheses_bal (sen )
print (error ,raw ,inline ,line )
```

자료구조 3

컴퓨터전자시스템공학부
201902520
이상윤