괄호 변환

문제 요약

‘콘’은 다른 개발자가 작성한 소스 코드를 분석하여 문제점을 발견하고 수정하라는 업무 과제를 받음

대부분 소스 코드 내 작성된 괄호가 개수는 맞는데
짝이 맞지 않아서 오류가 발생하네...

수정해야 할 소스 파일이 너무 많은데???
소스코드에 작성된 모든 괄호를 뽑아서 올바른 순서대로 배치된 괄호
문자열을 알려주는 프로그램을 개발해서 한번에 해결하자

문제설명

매개변수 설명

- p는 ‘(‘ 와 ‘)’로만 이루어진 문자열이며 길이는 2이상 1,000 이하인 짝수
- 문자열 p를 이루는 ‘(‘ 와 ‘)’의 개수는 항상 같음
- 만약 p가 이미 “올바른 괄호 문자열 " 이라면 그대로 return 하면 됨

용어의 정의

- ‘(‘ 와 ‘)’의 개수가 같다 = 균형 잡힌 괄호 문자 (매개변수로 ‘(‘,’)’의 개수가 같은 것이 들어오기 때문에 디폴트로 균형 잡힌 괄호 문자)
- 균형 잡힌 괄호 문자에서 괄호의 짝도 모두 맞다 = 올바른 괄호 문자
- 예
  - “(()))(“ -> “균형 잡힌 괄호 문자열”이지만, “올바른 괄호 문자열 " 은 아님
  - “(())()” ->“균형 잡힌 괄호 문자열”이면서 동시에, “올바른 괄호 문자열 " 임

문제설명

용어의 정의

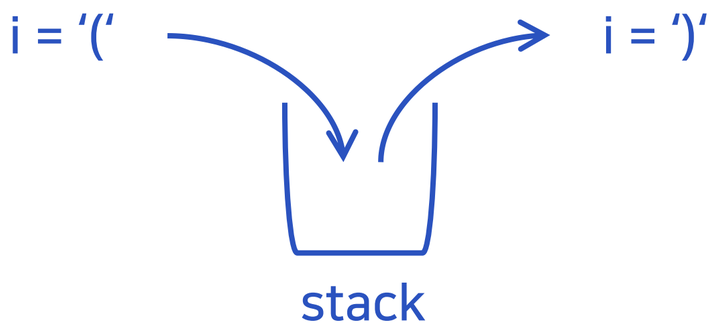
‘(‘와 ‘)’로만 이루어진 문자열 w가 “균형 잡힌 괄호 문자열“ 이라면 다음과 같은 과정을 통해 “올바른 괄호 문자열 " 로 변환할 수 있음

1. 입력이 빈 문자열인 경우, 빈 문자열을 반환
2. 문자열 w를 두 “균형 잡힌 괄호 문자열“ u,v로 분리
   단, “균형 잡힌 괄호 문자열 " 로 더 이상 분리할 수 없어야 하며, v는 빈 문자열이 될 수 있음
3. 문자열 u가 “올바른 괄호 문자열“이라면 문자열 v에 대해 1단계부터 다시 수행
   3-1. 수행한 결과 문자열을 u에 이어 붙인 후 반환
4. 문자열 u가 “올바른 괄호 문자열 " 이 아니라면 아래 과정을 수행
   4-1. 빈 문자열에 첫 번째 문자로 ‘(‘를 붙임
   4-2. 문자열 v에 대해 1단계부터 재귀적으로 수행한 결과 문자열을 이어 붙임
   4-3. ‘)’를 다시 붙임
   4-4. u의 첫번째와 마지막 문자를 제거하고, 나머지 문자열의 괄호 방향을 뒤집어서 뒤에 붙음
   4-5. 생성된 문자열을 반환

코드 설명

```python
# 주어진 괄호 문자열이 올바른지 여부를 확인하는 함수
def check(s):
    stack = []  # 스택을 사용하여 '('를 저장

    for i in s:  # 문자열의 각 문자를 순회
        if i == '(':  # 여는 괄호인 경우
            stack.append(i)  # 스택에 추가
        else:  # 닫는 괄호인 경우
            if len(stack) == 0:  # 스택이 비어있다면 짝이 맞지 않음
                return False
            stack.pop()  # 스택에서 여는 괄호 '('를 제거 (짝이 맞는 경우)
    return True  # 스택이 비어 있으면 올바른 괄호 문자열이니 True출력
```




```python
# 주어진 문자열을 균형잡힌 괄호 문자열 u, v로 분리하는 함수
def divide(s):
    left, right = 0, 0  # 왼쪽 괄호 '('와 오른쪽 괄호 ')'의 개수를 카운트
    for i in range(len(s)):
        if s[i] == '(':  # 여는 괄호가 나오면
            left += 1  # 왼쪽 괄호 개수 증가
        else:  # 닫는 괄호가 나오면
            right += 1  # 오른쪽 괄호 개수 증가

        # 여는 괄호와 닫는 괄호의 개수가 같아지는 순간, 균형잡힌 문자열을 찾음
        if left == right:
            # s[:i+1]: 0부터 i까지의 부분 문자열을 반환 (여기서 +1은 i번째 문자를 포함하기 위해서임)
            # 즉, s의 처음부터 i번째 문자까지를 u로, 나머지 부분을 v로 나누는 것
            return s[:i+1], s[i+1:]  # u, v를 반환
```

'(( ))) ('

→ Left = 2, right = 2

S[:2] = '(( ))', s[3:] = ') ('

코드 설명

1. 입력이 빈 문자열인 경우, 빈 문자열을 반환
2. 문자열 w를 두 “균형 잡힌 괄호 문자열“ u,v로 분리
   단, “균형 잡힌 괄호 문자열 " 로 더 이상 분리할 수 없어야 하며, v는 빈 문자열이 될 수 있음
3. 문자열 u가 “올바른 괄호 문자열“이라면 문자열 v에 대해 1단계부터 다시 수행

   3-1. 수행한 결과 문자열을 u에 이어 붙인 후 반환
4. 문자열 u가 “올바른 괄호 문자열 " 이 아니라면 아래 과정을 수행

   4-1. 빈 문자열에 첫 번째 문자로 ‘(‘를 붙임

   4-2. 문자열 v에 대해 1단계부터 재귀적으로 수행한 결과 문자열을 이어 붙임

   4-3. ‘)’를 다시 붙임

   4-4. u의 첫번째와 마지막 문자를 제거하고, 나머지 문자열의 괄호 방향을 뒤집어서 뒤에 붙음

   4-5. 생성된 문자열을 반환

```python
# 올바른 괄호 문자열로 변환하는 함수
def solution(p):
    #1
    if not p:
        return ''

    # 2
    u, v = divide(p)

    # 3, 3-1
    if check(u):  # u가 올바른 괄호 문자열이면
        return u + solution(v)  # u는 그대로 두고 v에 대해 재귀적으로 처리한 후 u에 붙임

    # 4-1 ~ 4-3
    else:
        # u가 올바르지 않은 경우
        answer = '('  # 빈 문자열에 '('를 붙임
        answer += solution(v)  # v를 재귀적으로 처리한 결과를 붙임
        answer += ')'  # 다시 ')'를 붙임


        # 4-4, 4-5
        for s in u[1:len(u)-1]:  # u의 첫 문자와 마지막 문자를 제외한 부분만 확인
            if s == '(':  # '('를 ')'로 바꾸고
                answer += ')'
            else:  # ')'를 '('로 바꿈
                answer += '('

        return answer  # 변환된 결과를 반환
```