문제 풀이

- 이 문제는 괄호가 의미하는 내용이 크게 2가지로 나뉩니다.

1. (), [] 값이 각 각 2와 3인 경우.
2. (()),[[]] 처럼 각 각 2 혹은 3을 곱해줘야 하는 경우.

* 또한, ()와 [] 사이의 연산자는 +가 되며, 이 때, 2번 조건의 곱셈때문에 식 자체에 연산 우선순위가 생겨서 코드가 더욱 복잡해집니다.

* 그리고, 이 문제는 괄호열이 올바른 괄호열인지 확인하는 프로세스도 필요합니다.

문제 풀이

- 코드 자체는 크게 조건 1번과 2번 그리고 올바른 괄호열인지 확인하는 과정으로 구분할 수 있습니다.

1. 괄호열이 올바른지 확인하는 과정.
2. (), []만을 인식하여 2와 3으로 치환하는 과정.
3. 2번 프로세스로 완성된 식(예. ((2)[3]) 에서 괄호 사이의 값들을 2 또는 3배 해주는 과정
4. 마지막으로 값들을 모두 더해 출력해주는 과정.

- 1번 프로세스

  예 : 괄호열 "(()[[]])"이 올바른 괄호열인지 아닌지를 확인하는 과정부터 설명하겠습니다.

  "(" 일 경우 stack에 push를 해줍니다.

idx

"(,(,),[,[,],],)"

STACK = (

- 1번 프로세스

  "(" 일 경우 stack에 push를 해줍니다.

idx

"(,(,),[,[,],],)"

STACK = (, (

- 1번 프로세스

  ")" 일 경우 stack에서 pop을 해준 후, pop한 원소가 "(" 인지를 확인합니다.

idx

"(,(,),[,[,],],)"

pop

STACK = (, (

- 1번 프로세스

  "[" 일 경우 stack에 push를 해줍니다.

idx

"(,(,),[,[,],],)"

STACK = (,[

- 1번 프로세스

  "[" 일 경우 stack에 push를 해줍니다.

idx

"(,(,),[,[,],],)"

STACK = (,[,[

- 1번 프로세스

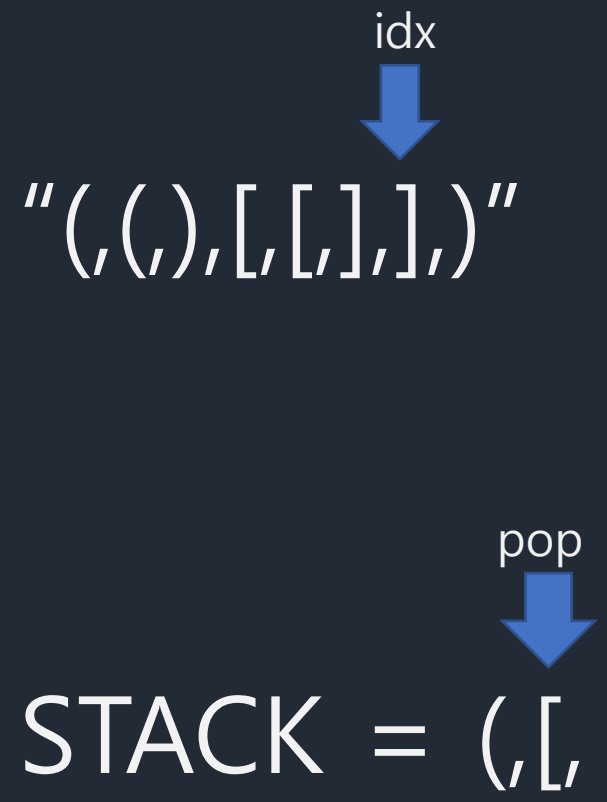
  "]" 일 경우 stack에서 pop을 해준 후, pop한 원소가 "[" 인지를 확인합니다.

"(,(,),[,[,],],)"

STACK = (,[,[

- 1번 프로세스

  "]" 일 경우 stack에서 pop을 해준 후, pop한 원소가 "[" 인지를 확인합니다.



- 1번 프로세스

  ")" 일 경우 stack에서 pop을 해준 후, pop한 원소가 "(" 인지를 확인합니다.

idx

"(,(,),[,[,],],)"

pop

STACK = (

- 1번 프로세스

  이 과정 중에, STACK이 비어서 pop을 못하거나, 비교하는 과정에서 서로 다른 짝의 괄호열이거나,

  맨 마지막에 STACK이 비지 않으면 올바르지 않은 괄호열입니다.

- 2번 프로세스
  ( ), [ ]를 2와 3으로 치환해주는 프로세스입니다.

  (가 나오면 STACK에 push해줍니다.

  idx

"(,(,),[,[,],],)"

STACK1 = (

- 2번 프로세스

  또다시 (가 나오면 STACK에 (를 push해줍니다.

  idx

  “(,(,),[,[,],],)”

  STACK = (, (

- 2번 프로세스

)가 나오면 {STACK의 마지막 idx가 (인지를 확인한 후} STACK에 (를 pop한 후에 2를 push합니다.

idx

"(,(,),[,[,],],)"

Pop & 2push

STACK = (, (

- 2번 프로세스

  [가 나오면 STACK에 [를 Push 해줍니다.

idx

"(,(,),[,[,],],)"

STACK = (, 2, [

- 2번 프로세스

  [가 나오면 STACK에 [를 Push 해줍니다.

idx

"(,(,),[,[,],],)"

STACK = (, 2, [, [

- 2번 프로세스

]가 나오면 {STACK에 마지막 idx가 [인지를 확인 한 후,} [를 Pop하고 3을 push 해줍니다.

idx

"(,(,),[,[,],],)"

Pop & 3push

STACK = (, 2, [, [

- 2번 프로세스

  ]가 나오면 {STACK에 마지막 idx가 [인지를 확인 한 후,} "["가 아니기 때문에 "]"를 push해줍니다.

"(,(,),[,[,],],)"

STACK = (, 2, [, 3,]

- 2번 프로세스

)가 나오면 {STACK에 마지막 idx가 "("인지를 확인 한 후,} (["가 아니기 때문에 ")"를 push해줍니다.

"(,(,),[,[,],],)"

STACK = (, 2, [, 3,],)

- 이로써 치환해주는 작업이 끝났습니다.

- 3번 프로세스

이제 괄호 안에 값을 곱해줍니다.
이 때 A(B + C) = AB + AC의 원리를 이용합니다.
(가 나오면 STACK에 push합니다.

idx

"(, 2, [, 3,],)"

STACK = (

- 3번 프로세스

  숫자가 나오면 STACK에 push해줍니다.

idx

"(, 2, [, 3,],)"

STACK = (, 2

- 3번 프로세스

  [가 나오면 스킵합니다.

idx

"(, 2, [, 3,],)"

STACK = (, 2, [

- 3번 프로세스

  숫자가 나오면 STACK에 push합니다.

idx

"(, 2, [, 3,],)"

STACK = (, 2, [, 3

- 3번 프로세스

  ]가 나오면 STACK을 반대로 돌면서 '['을 pop할 때까지, 3을 곱해서 더해줍니다.

"(, 2, [, 3,],)"

STACK = (, 2, 9

- 3번 프로세스

)가 나오면 STACK을 반대로 돌면서 '('을 pop할 때까지, 2를 곱해서 더해줍니다.

"(, 2, [, 3,],)"

STACK = 22