프로그래머스 Level2

연번	이름	메모	풀이1	풀이2
1				
2				
3				
4				
5				
6				
7				
8				
9				
10				
11				
12				
13				
14				
15				
16				
17				
18				
19				
20				
21				
22				
23				
24				
25				
26				
27				
28				
29				
30				
31				
32				
33				
34				

주식가격

문제 설명
초 단위로 기록된 주식가격이 담긴 배열 prices가 매개변수로 주어질 때, 가격이 떨어지지 않은 기간은 몇 초인지를 return 하도록 solution 함수를 완성하세요.

제한사항
prices의 각 가격은 1 이상 10,000 이하인 자연수입니다.
prices의 길이는 2 이상 100,000 이하입니다.
입출력 예
prices return
[1, 2, 3, 2, 3] [4, 3, 1, 1, 0]
입출력 예 설명
1초 시점의 ₩1은 끝까지 가격이 떨어지지 않았습니다.
2초 시점의 ₩2은 끝까지 가격이 떨어지지 않았습니다.
3초 시점의 ₩3은 1초뒤에 가격이 떨어집니다. 따라서 1초간 가격이 떨어지지 않은 것으로 봅니다.
4초 시점의 ₩2은 1초간 가격이 떨어지지 않았습니다.
5초 시점의 ₩3은 0초간 가격이 떨어지지 않았습니다.
※ 공지 - 2019년 2월 28일 지문이 리뉴얼되었습니다.

쇠막대기

제 설명
여러 개의 쇠막대기를 레이저로 절단하려고 합니다. 효율적인 작업을 위해서 쇠막대기를 아래에서 위로 겹쳐 놓고, 레이저를 위에서 수직으로 발사하여 쇠막대기들을 자릅니다. 쇠막대기와 레이저의 배치는 다음 조건을 만족합니다.

- 쇠막대기는 자신보다 긴 쇠막대기 위에만 놓일 수 있습니다.
- 쇠막대기를 다른 쇠막대기 위에 놓는 경우 완전히 포함되도록 놓되, 끝점은 겹치지 않도록 놓습니다.
- 각 쇠막대기를 자르는 레이저는 적어도 하나 존재합니다.
- 레이저는 어떤 쇠막대기의 양 끝점과도 겹치지 않습니다.

아래 그림은 위 조건을 만족하는 예를 보여줍니다. 수평으로 그려진 굵은 실선은 쇠막대기이고, 점은 레이저의 위치, 수직으로 그려진 점선 화살표는 레이저의 발사 방향입니다.

image0.png

이러한 레이저와 쇠막대기의 배치는 다음과 같이 괄호를 이용하여 왼쪽부터 순서대로 표현할 수 있습니다.

(a) 레이저는 여는 괄호와 닫는 괄호의 인접한 쌍 '()'으로 표현합니다. 또한 모든 '()'는 반드시 레이저를 표현합니다.
(b) 쇠막대기의 왼쪽 끝은 여는 괄호 '('로, 오른쪽 끝은 닫힌 괄호 ')'로 표현됩니다.

위 예의 괄호 표현은 그림 위에 주어져 있습니다.
쇠막대기는 레이저에 의해 몇 개의 조각으로 잘리는데, 위 예에서 가장 위에 있는 두 개의 쇠막대기는 각각 3개와 2개의 조각으로 잘리고, 이와 같은 방식으로 주어진 쇠막대기들은 총 17개의 조각으로 잘립니다.

쇠막대기와 레이저의 배치를 표현한 문자열 arrangement가 매개변수로 주어질 때, 잘린 쇠막대기 조각의 총 개수를 return 하도록 solution 함수를 작성해주세요.

제한사항
arrangement의 길이는 최대 100,000입니다.
arrangement의 여는 괄호와 닫는 괄호는 항상 쌍을 이룹니다.

입출력 예

arrangement	return
()(((()())(())(()))(())	17

입출력 예 설명
문제에 나온 예와 같습니다.

다리를 지나는 트럭

문제 설명

트럭 여러 대가 강을 가로지르는 일 차선 다리를 정해진 순으로 건너려 합니다. 모든 트럭이 다리를 건너려면 최소 몇 초가 걸리는지 알아내야 합니다. 트럭은 1초에 1만큼 움직이며, 다리 길이는 bridge_length이고 다리는 무게 weight까지 견딥니다.

※ 트럭이 다리에 완전히 오르지 않은 경우, 이 트럭의 무게는 고려하지 않습니다.

예를 들어, 길이가 2이고 10kg 무게를 견디는 다리가 있습니다. 무게가 [7, 4, 5, 6]kg인 트럭이 순서대로 최단 시간 안에 다리를 건너려면 다음과 같이 건너야 합니다.

경과 시간		다리를 지난 트럭	다리를 건너는 트럭	대기 트럭
0	[]	[]	[7,4,5,6]	
1~2	[]	[7]	[4,5,6]	
3	[7]	[4]	[5,6]	
4	[7]	[4,5]	[6]	
5	[7,4]	[5]	[6]	
6~7	[7,4,5]	[6]	[]	
8	[7,4,5,6]	[]	[]	

따라서, 모든 트럭이 다리를 지나려면 최소 8초가 걸립니다.

solution 함수의 매개변수로 다리 길이 bridge_length, 다리가 견딜 수 있는 무게 weight, 트럭별 무게 truck_weights가 주어집니다. 이때 모든 트럭이 다리를 건너려면 최소 몇 초가 걸리는지 return 하도록 solution 함수를 완성하세요.

제한 조건

bridge_length는 1 이상 10,000 이하입니다.

weight는 1 이상 10,000 이하입니다.

truck_weights의 길이는 1 이상 10,000 이하입니다.

모든 트럭의 무게는 1 이상 weight 이하입니다.

입출력 예

bridge_length	weight	truck_weights	return
2	10	[7,4,5,6]	8
100	100	[10]	101
100	100	[10,10,10,10,10,10,10,10,10,10]	110

출처

카카오 프렌즈 컬러링북

문제 설명
카카오 프렌즈 컬러링북
출판사의 편집자인 어피치는 네오에게 컬러링북에 들어갈 원화를 그려달라고 부탁하여 여러 장의 그림을 받았다. 여러 장의 그림을 난이도 순으로 컬러링북에 넣고 싶었던 어피치는 영역이 많으면 색칠하기가 까다로워 어려워진다는 사실을 발견하고 그림의 난이도를 영역의 수로 정의하였다. (영역이란 상하좌우로 연결된 같은 색상의 공간을 의미한다.)

그림에 몇 개의 영역이 있는지와 가장 큰 영역의 넓이는 얼마인지 계산하는 프로그램을 작성해보자.

alt text

위의 그림은 총 12개 영역으로 이루어져 있으며, 가장 넓은 영역은 어피치의 얼굴면으로 넓이는 120이다.

입력 형식
입력은 그림의 크기를 나타내는 m과 n, 그리고 그림을 나타내는 m × n 크기의 2차원 배열 picture로 주어진다. 제한조건은 아래와 같다.

1 <= m, n <= 100
picture의 원소는 0 이상 2^31 - 1 이하의 임의의 값이다.
picture의 원소 중 값이 0인 경우는 색칠하지 않는 영역을 뜻한다.
출력 형식
리턴 타입은 원소가 두 개인 정수 배열이다. 그림에 몇 개의 영역이 있는지와 가장 큰 영역은 몇 칸으로 이루어져 있는지를 리턴한다.

예제 입출력

m	n	picture	answer
6	4	[[1, 1, 1, 0], [1, 2, 2, 0], [1, 0, 0, 1], [0, 0, 0, 1], [0, 0, 0, 3], [0, 0, 0, 3]]	[4, 5]

예제에 대한 설명
예제로 주어진 그림은 총 4개의 영역으로 구성되어 있으며, 왼쪽 위의 영역과 오른쪽의 영역은 모두 1로 구성되어 있지만 상하좌우로 이어져있지 않으므로 다른 영역이다. 가장 넓은 영역은 왼쪽 위 1이 차지하는 영역으로 총 5칸이다.

조이스틱

문제 설명
조이스틱으로 알파벳 이름을 완성하세요. 맨 처음엔 A로만 이루어져 있습니다.
ex) 완성해야 하는 이름이 세 글자면 AAA, 네 글자면 AAAA

조이스틱을 각 방향으로 움직이면 아래와 같습니다.

▲ - 다음 알파벳
▼ - 이전 알파벳 (A에서 아래쪽으로 이동하면 Z로)
◀ - 커서를 왼쪽으로 이동 (첫 번째 위치에서 왼쪽으로 이동하면 마지막 문자에 커서)
▶ - 커서를 오른쪽으로 이동
예를 들어 아래의 방법으로 JAZ를 만들 수 있습니다.

- 첫 번째 위치에서 조이스틱을 위로 9번 조작하여 J를 완성합니다.
- 조이스틱을 왼쪽으로 1번 조작하여 커서를 마지막 문자 위치로 이동시킵니다.
- 마지막 위치에서 조이스틱을 아래로 1번 조작하여 Z를 완성합니다.
따라서 11번 이동시켜 "JAZ"를 만들 수 있고, 이때가 최소 이동입니다.
만들고자 하는 이름 name이 매개변수로 주어질 때, 이름에 대해 조이스틱 조작 횟수의 최솟값을 return 하도록 solution 함수를 만드세요.

제한 사항
name은 알파벳 대문자로만 이루어져 있습니다.
name의 길이는 1 이상 20 이하입니다.
입출력 예

name	return
JEROEN	56
JAN	23

출처

※ 공지 - 2019년 2월 28일 테스트케이스가 추가되었습니다.

더 맵게

문제 설명
매운 것을 좋아하는 Leo는 모든 음식의 스코빌 지수를 K 이상으로 만들고 싶습니다. 모든 음식의 스코빌 지수를 K 이상으로 만들기 위해 Leo는 스코빌 지수가 가장 낮은 두 개의 음식을 아래와 같이 특별한 방법으로 섞어 새로운 음식을 만듭니다.

섞은 음식의 스코빌 지수 = 가장 맵지 않은 음식의 스코빌 지수 + (두 번째로 맵지 않은 음식의 스코빌 지수 * 2)
Leo는 모든 음식의 스코빌 지수가 K 이상이 될 때까지 반복하여 섞습니다.
Leo가 가진 음식의 스코빌 지수를 담은 배열 scoville과 원하는 스코빌 지수 K가 주어질 때, 모든 음식의 스코빌 지수를 K 이상으로 만들기 위해 섞어야 하는 최소 횟수를 return 하도록 solution 함수를 작성해주세요.

제한 사항
scoville의 길이는 1 이상 1,000,000 이하입니다.
K는 0 이상 1,000,000,000 이하입니다.
scoville의 원소는 각각 0 이상 1,000,000 이하입니다.
모든 음식의 스코빌 지수를 K 이상으로 만들 수 없는 경우에는 -1을 return 합니다.
입출력 예

scoville	K	return
[1, 2, 3, 9, 10, 12]	7	2

입출력 예 설명
스코빌 지수가 1인 음식과 2인 음식을 섞으면 음식의 스코빌 지수가 아래와 같이 됩니다.
새로운 음식의 스코빌 지수 = 1 + (2 * 2) = 5
가진 음식의 스코빌 지수 = [5, 3, 9, 10, 12]

스코빌 지수가 3인 음식과 5인 음식을 섞으면 음식의 스코빌 지수가 아래와 같이 됩니다.
새로운 음식의 스코빌 지수 = 3 + (5 * 2) = 13
가진 음식의 스코빌 지수 = [13, 9, 10, 12]

모든 음식의 스코빌 지수가 7 이상이 되었고 이때 섞은 횟수는 2회입니다.

큰 수 만들기

문제 설명
어떤 숫자에서 k개의 수를 제거했을 때 얻을 수 있는 가장 큰 숫자를 구하려 합니다.

예를 들어, 숫자 1924에서 수 두 개를 제거하면 [19, 12, 14, 92, 94, 24] 를 만들 수 있습니다. 이 중 가장 큰 숫자는 94 입니다.

문자열 형식으로 숫자 number와 제거할 수의 개수 k가 solution 함수의 매개변수로 주어집니다. number에서 k 개의 수를 제거했을 때 만들 수 있는 수 중 가장 큰 숫자를 문자열 형태로 return 하도록 solution 함수를 완성하세요.

제한 조건
number는 1자리 이상, 1,000,000자리 이하인 숫자입니다.
k는 1 이상 number의 자릿수 미만인 자연수입니다.
입출력 예

number	k	return
1924	2	94
1231234	3	3234
4177252841	4	775841

출처

[2020카카오공채] 괄호 변환

문제 설명
카카오에 신입 개발자로 입사한 콘은 선배 개발자로부터 개발역량 강화를 위해 다른 개발자가 작성한 소스 코드를 분석하여 문제점을 발견하고 수정하라는 업무 과제를 받았습니다. 소스를 컴파일하여 로그를 보니 대부분 소스 코드 내 작성된 괄호가 개수는 맞지만 짝이 맞지 않은 형태로 작성되어 오류가 나는 것을 알게 되었습니다.
수정해야 할 소스 파일이 너무 많아서 고민하던 콘은 소스 코드에 작성된 모든 괄호를 뽑아서 올바른 순서대로 배치된 괄호 문자열을 알려주는 프로그램을 다음과 같이 개발하려고 합니다.

용어의 정의
'(' 와 ')' 로만 이루어진 문자열이 있을 경우, '(' 의 개수와 ')' 의 개수가 같다면 이를 균형잡힌 괄호 문자열이라고 부릅니다.
그리고 여기에 '('와 ')'의 괄호의 짝도 모두 맞을 경우에는 이를 올바른 괄호 문자열이라고 부릅니다.
예를 들어, "(()))("와 같은 문자열은 균형잡힌 괄호 문자열 이지만 올바른 괄호 문자열은 아닙니다.
반면에 "(())()"와 같은 문자열은 균형잡힌 괄호 문자열 이면서 동시에 올바른 괄호 문자열 입니다.

'(' 와 ')' 로만 이루어진 문자열 w가 균형잡힌 괄호 문자열 이라면 다음과 같은 과정을 통해 올바른 괄호 문자열로 변환할 수 있습니다.

1. 입력이 빈 문자열인 경우, 빈 문자열을 반환합니다.
2. 문자열 w를 두 "균형잡힌 괄호 문자열" u, v로 분리합니다. 단, u는 "균형잡힌 괄호 문자열"로 더 이상 분리할 수 없어야 하며, v는 빈 문자열이 될 수 있습니다.
3. 문자열 u가 "올바른 괄호 문자열" 이라면 문자열 v에 대해 1단계부터 다시 수행합니다.
   3-1. 수행한 결과 문자열을 u에 이어 붙인 후 반환합니다.
4. 문자열 u가 "올바른 괄호 문자열"이 아니라면 아래 과정을 수행합니다.
   4-1. 빈 문자열에 첫 번째 문자로 '('를 붙입니다.
   4-2. 문자열 v에 대해 1단계부터 재귀적으로 수행한 결과 문자열을 이어 붙입니다.
   4-3. ')'를 다시 붙입니다.
   4-4. u의 첫 번째와 마지막 문자를 제거하고, 나머지 문자열의 괄호 방향을 뒤집어서 뒤에 붙입니다.
   4-5. 생성된 문자열을 반환합니다.
균형잡힌 괄호 문자열 p가 매개변수로 주어질 때, 주어진 알고리즘을 수행해 올바른 괄호 문자열로 변환한 결과를 return 하도록 solution 함수를 완성해 주세요.

매개변수 설명
p는 '(' 와 ')' 로만 이루어진 문자열이며 길이는 2 이상 1,000 이하인 짝수입니다.
문자열 p를 이루는 '(' 와 ')' 의 개수는 항상 같습니다.
만약 p가 이미 올바른 괄호 문자열이라면 그대로 return 하면 됩니다.
입출력 예

p result
"(()())()" "(()())()"
")(" "()"
"()))((()" "()(())()"
입출력 예에 대한 설명
입출력 예 #1
이미 올바른 괄호 문자열 입니다.
문자열 분리

입출력 예 #2

두 문자열 u, v로 분리합니다.
u = ")("
v = ""
u가 올바른 괄호 문자열이 아니므로 다음과 같이 새로운 문자열을 만듭니다.
v에 대해 1단계부터 재귀적으로 수행하면 빈 문자열이 반환됩니다.
u의 앞뒤 문자를 제거하고, 나머지 문자의 괄호 방향을 뒤집으면 ""이 됩니다.
따라서 생성되는 문자열은 "(" + "" + ")" + ""이며, 최종적으로 "()"로 변환됩니다.
입출력 예 #3

두 문자열 u, v로 분리합니다.
u = "()"
v = "))((()"
문자열 u가 올바른 괄호 문자열이므로 그대로 두고, v에 대해 재귀적으로 수행합니다.
다시 두 문자열 u, v로 분리합니다.
u = "))(("
v = "()"
u가 올바른 괄호 문자열이 아니므로 다음과 같이 새로운 문자열을 만듭니다.
v에 대해 1단계부터 재귀적으로 수행하면 "()"이 반환됩니다.
u의 앞뒤 문자를 제거하고, 나머지 문자의 괄호 방향을 뒤집으면 "()"이 됩니다.
따라서 생성되는 문자열은 "(" + "()" + ")" + "()"이며, 최종적으로 "(())()"를 반환합니다.
처음에 그대로 둔 문자열에 반환된 문자열을 이어 붙이면 "()" + "(())()" = "()(())()"가 됩니다.

소수 찾기

문제 설명
한자리 숫자가 적힌 종이 조각이 흩어져있습니다. 흩어진 종이 조각을 붙여 소수를 몇 개 만들 수 있는지 알아내려 합니다.

각 종이 조각에 적힌 숫자가 적힌 문자열 numbers가 주어졌을 때, 종이 조각으로 만들 수 있는 소수가 몇 개인지 return 하도록 solution 함수를 완성해주세요.

제한사항
numbers는 길이 1 이상 7 이하인 문자열입니다.
numbers는 0~9까지 숫자만으로 이루어져 있습니다.
013은 0, 1, 3 숫자가 적힌 종이 조각이 흩어져있다는 의미입니다.
입출력 예

numbers	return
17	3
011	2

입출력 예 설명
예제 #1
[1, 7]으로는 소수 [7, 17, 71]를 만들 수 있습니다.

예제 #2
[0, 1, 1]으로는 소수 [11, 101]를 만들 수 있습니다.

11과 011은 같은 숫자로 취급합니다.
출처

짝지어 제거하기

문제 설명

짝지어 제거하기는, 알파벳 소문자로 이루어진 문자열을 가지고 시작합니다. 먼저 문자열에서 같은 알파벳이 2개 붙어 있는 짝을 찾습니다. 그다음, 그 둘을 제거한 뒤, 앞뒤로 문자열을 이어 붙입니다. 이 과정을 반복해서 문자열을 모두 제거한다면 짝지어 제거하기가 종료됩니다. 문자열 S가 주어졌을 때, 짝지어 제거하기를 성공적으로 수행할 수 있는지 반환하는 함수를 완성해 주세요. 성공적으로 수행할 수 있으면 1을, 아닐 경우 0을 리턴해주면 됩니다.

예를 들어, 문자열 S = baabaa 라면

b *aa* baa → *bb* aa → *aa* →

의 순서로 문자열을 모두 제거할 수 있으므로 1을 반환합니다.

제한사항

- 문자열의 길이 : 1,000,000이하의 자연수
- 문자열은 모두 소문자로 이루어져 있습니다.

입출력 예

s	**result**
baabaa	1
cdcd	0

입출력 예 설명

입출력 예 #1

위의 예시와 같습니다.

입출력 예 #2

문자열이 남아있지만 짝지어 제거할 수 있는 문자열이 더 이상 존재하지 않기 때문에 0을 반환합니다.

JadenCase 문자열 만들기

문제 설명

JadenCase란 모든 단어의 첫 문자가 대문자이고, 그 외의 알파벳은 소문자인 문자열입니다. 문자열 s가 주어졌을 때, s를 JadenCase로 바꾼 문자열을 리턴하는 함수, solution을 완성해주세요.

제한 조건

- s는 길이 1 이상인 문자열입니다.
- s는 알파벳과 공백문자(" ")로 이루어져 있습니다.
- 첫 문자가 영문이 아닐때에는 이어지는 영문은 소문자로 씁니다. (첫번째 입출력 예 참고)

입출력 예

s	**return**
3people unFollowed me	3people Unfollowed Me
for the last week	For The Last Week

행렬의 곱셈

문제 설명

2차원 행렬 arr1과 arr2를 입력받아, arr1에 arr2를 곱한 결과를 반환하는 함수, solution을 완성해주세요.

제한 조건

- 행렬 arr1, arr2의 행과 열의 길이는 2 이상 100 이하입니다.
- 행렬 arr1, arr2의 원소는 -10 이상 20 이하인 자연수입니다.
- 곱할 수 있는 배열만 주어집니다.

입출력 예

arr1	**arr2**	**return**
[[1, 4], [3, 2], [4, 1]]	[[3, 3], [3, 3]]	[[15, 15], [15, 15], [15, 15]]
[[2, 3, 2], [4, 2, 4], [3, 1, 4]]	[[5, 4, 3], [2, 4, 1], [3, 1, 1]]	[[22, 22, 11], [36, 28, 18], [29, 20, 14]]

최솟값 만들기

문제 설명

길이가 같은 배열 A, B 두개가 있습니다. 각 배열은 자연수로 이루어져 있습니다.

배열 A, B에서 각각 한 개의 숫자를 뽑아 두 수를 곱합니다. 이러한 과정을 배열의 길이만큼 반복하며, 두 수를 곱한 값을 누적하여 더합니다. 이때 최종적으로 누적된 값이 최소가 되도록 만드는 것이 목표입니다. (단, 각 배열에서 k번째 숫자를 뽑았다면 다음에 k번째 숫자는 다시 뽑을 수 없습니다.)

예를 들어 A = [1, 4, 2] , B = [5, 4, 4] 라면

- A에서 첫번째 숫자인 1, B에서 두번째 숫자인 5를 뽑아 곱하여 더합니다. (누적된 값 : 0 + 5(1x5) = 5)
- A에서 두번째 숫자인 4, B에서 세번째 숫자인 4를 뽑아 곱하여 더합니다. (누적된 값 : 5 + 16(4x4) = 21)
- A에서 세번째 숫자인 2, B에서 첫번째 숫자인 4를 뽑아 곱하여 더합니다. (누적된 값 : 21 + 8(2x4) = 29)

즉, 이 경우가 최소가 되므로 **29**를 return 합니다.

배열 A, B가 주어질 때 최종적으로 누적된 최솟값을 return 하는 solution 함수를 완성해 주세요.

제한사항

- 배열 A, B의 크기 : 1,000 이하의 자연수
- 배열 A, B의 원소의 크기 : 1,000 이하의 자연수

입출력 예

A	B	answer
[1, 4, 2]	[5, 4, 4]	29
[1,2]	[3,4]	10

입출력 예 설명

입출력 예 #1

문제의 예시와 같습니다.

입출력 예 #2

A에서 첫번째 숫자인 1, B에서 두번째 숫자인 4를 뽑아 곱하여 더합니다. (누적된 값 : 4) 다음, A에서 두번째 숫자인 2, B에서 첫번째 숫자인 3을 뽑아 곱하여 더합니다. (누적된 값 : 4 + 6 = 10)

이 경우가 최소이므로 10을 return 합니다.

최댓값과 최솟값

문제 설명

문자열 s에는 공백으로 구분된 숫자들이 저장되어 있습니다. str에 나타나는 숫자 중 최소값과 최대값을 찾아 이를 (최소값) (최대값)형태의 문자열을 반환하는 함수, solution을 완성하세요.

예를들어 s가 1 2 3 4라면 1 4를 리턴하고, -1 -2 -3 -4라면 -4 -1을 리턴하면 됩니다.

제한 조건

- s에는 둘 이상의 정수가 공백으로 구분되어 있습니다.

입출력 예

s	**return**
1 2 3 4	1 4
-1 -2 -3 -4	-4 -1
-1 -1	-1 -1

포켓몬

문제 설명

당신은 폰켓몬을 잡기 위한 오랜 여행 끝에, 홍 박사님의 연구실에 도착했습니다. 홍 박사님은 당신에게 자신의 연구실에 있는 총 N 마리의 폰켓몬 중에서 N/2마리를 가져가도 좋다고 했습니다.

홍 박사님 연구실의 폰켓몬은 종류에 따라 번호를 붙여 구분합니다. 따라서 같은 종류의 폰켓몬은 같은 번호를 가지고 있습니다. 예를 들어 연구실에 총 4마리의 폰켓몬이 있고, 각 폰켓몬의 종류 번호가 [3번, 1번, 2번, 3번]이라면 이는 3번 폰켓몬 두 마리, 1번 폰켓몬 한 마리, 2번 폰켓몬 한 마리가 있음을 나타냅니다. 이때, 4마리의 폰켓몬 중 2마리를 고르는 방법은 다음과 같이 6가지가 있습니다.

1. 첫 번째(3번), 두 번째(1번) 폰켓몬을 선택
2. 첫 번째(3번), 세 번째(2번) 폰켓몬을 선택
3. 첫 번째(3번), 네 번째(3번) 폰켓몬을 선택
4. 두 번째(1번), 세 번째(2번) 폰켓몬을 선택
5. 두 번째(1번), 네 번째(3번) 폰켓몬을 선택
6. 세 번째(2번), 네 번째(3번) 폰켓몬을 선택

이때, 첫 번째(3번) 폰켓몬과 네 번째(3번) 폰켓몬을 선택하는 방법은 한 종류(3번 폰켓몬 두 마리)의 폰켓몬만 가질 수 있지만, 다른 방법들은 모두 두 종류의 폰켓몬을 가질 수 있습니다. 따라서 위 예시에서 가질 수 있는 폰켓몬 종류 수의 최댓값은 2가 됩니다.

당신은 최대한 다양한 종류의 폰켓몬을 가지길 원하기 때문에, 최대한 많은 종류의 폰켓몬을 포함해서 N/2마리를 선택하려 합니다. N마리 폰켓몬의 종류 번호가 담긴 배열 nums가 매개변수로 주어질 때, N/2마리의 폰켓몬을 선택하는 방법 중, 가장 많은 종류의 폰켓몬을 선택하는 방법을 찾아, 그때의 폰켓몬 종류 번호의 개수를 return 하도록 solution 함수를 완성해주세요.

제한사항

- nums는 폰켓몬의 종류 번호가 담긴 1차원 배열입니다.
- nums의 길이(N)는 1 이상 10,000 이하의 자연수이며, 항상 짝수로 주어집니다.
- 폰켓몬의 종류 번호는 1 이상 200,000 이하의 자연수로 나타냅니다.
- 가장 많은 종류의 폰켓몬을 선택하는 방법이 여러 가지인 경우에도, 선택할 수 있는 폰켓몬 종류 개수의 최댓값 하나만 return 하면 됩니다.

입출력 예

nums	**result**
[3,1,2,3]	2
[3,3,3,2,2,4]	3
[3,3,3,2,2,2]	2

입출력 예 설명

입출력 예 #1

문제의 예시와 같습니다.

입출력 예 #2

6마리의 폰켓몬이 있으므로, 3마리의 폰켓몬을 골라야 합니다.

가장 많은 종류의 폰켓몬을 고르기 위해서는 3번 폰켓몬 한 마리, 2번 폰켓몬 한 마리, 4번 폰켓몬 한 마리를 고르면 되며, 따라서 3을 return 합니다.

입출력 예 #3

6마리의 폰켓몬이 있으므로, 3마리의 폰켓몬을 골라야 합니다.

가장 많은 종류의 폰켓몬을 고르기 위해서는 3번 폰켓몬 한 마리와 2번 폰켓몬 두 마리를 고르거나, 혹은 3번 폰켓몬 두 마리와 3번 폰켓몬 한 마리를 고르면 됩니다. 따라서 최대 고를 수 있는 폰켓몬 종류의 수는 2입니다.

숫자의 표현

문제 설명

Finn은 요즘 수학공부에 빠져 있습니다. 수학 공부를 하던 Finn은 자연수 n을 연속한 자연수들로 표현하는 방법이 여러개라는 사실을 알게 되었습니다. 예를들어 15는 다음과 같이 4가지로 표현 할 수 있습니다.

- 1 + 2 + 3 + 4 + 5 = 15
- 4 + 5 + 6 = 15
- 7 + 8 = 15
- 15 = 15

자연수 n이 매개변수로 주어질 때, 연속된 자연수들로 n을 표현하는 방법의 수를 return하는 solution를 완성해주세요.

제한사항

- n은 10,000 이하의 자연수 입니다.

입출력 예

n	**result**
15	4

입출력 예 설명

입출력 예#1

문제의 예시와 같습니다.

땅따먹기

문제 설명

땅따먹기 게임을 하려고 합니다. 땅따먹기 게임의 땅(land)은 총 N행 4열로 이루어져 있고, 모든 칸에는 점수가 쓰여 있습니다. 1행부터 땅을 밟으며 한 행씩 내려올 때, 각 행의 4칸 중 한 칸만 밟으면서 내려와야 합니다. **단, 땅따먹기 게임에는 한 행씩 내려올 때, 같은 열을 연속해서 밟을 수 없는 특수 규칙이 있습니다.**

예를 들면,

1	2	3	5
5	6	7	8
4	3	2	1

로 땅이 주어졌다면, 1행에서 네번째 칸 (5)를 밟았으면, 2행의 네번째 칸 (8)은 밟을 수 없습니다.

마지막 행까지 모두 내려왔을 때, 얻을 수 있는 점수의 최대값을 return하는 solution 함수를 완성해 주세요. 위 예의 경우, 1행의 네번째 칸 (5), 2행의 세번째 칸 (7), 3행의 첫번째 칸 (4) 땅을 밟아 16점이 최고점이 되므로 16을 return 하면 됩니다.

제한사항

- 행의 개수 N : 100,000 이하의 자연수
- 열의 개수는 4개이고, 땅(land)은 2차원 배열로 주어집니다.
- 점수 : 100 이하의 자연수

입출력 예

land	answer
[[1,2,3,5],[5,6,7,8],[4,3,2,1]]	16

입출력 예 설명

입출력 예 #1

문제의 예시와 같습니다.

다음 큰 숫자

문제 설명

자연수 n이 주어졌을 때, n의 다음 큰 숫자는 다음과 같이 정의 합니다.

- 조건 1. n의 다음 큰 숫자는 n보다 큰 자연수 입니다.
- 조건 2. n의 다음 큰 숫자와 n은 2진수로 변환했을 때 1의 갯수가 같습니다.
- 조건 3. n의 다음 큰 숫자는 조건 1, 2를 만족하는 수 중 가장 작은 수 입니다.

예를 들어서 78(1001110)의 다음 큰 숫자는 83(1010011)입니다.

자연수 n이 매개변수로 주어질 때, n의 다음 큰 숫자를 return 하는 solution 함수를 완성해주세요.

제한 사항

- n은 1,000,000 이하의 자연수 입니다.

입출력 예

n	**result**
78	83
15	23

입출력 예 설명

입출력 예#1

문제 예시와 같습니다.

입출력 예#2

15(1111)의 다음 큰 숫자는 23(10111)입니다.

단체사진 찍기

문제 설명

단체사진 찍기

가을을 맞아 카카오프렌즈는 단체로 소풍을 떠났다. 즐거운 시간을 보내고 마지막에 단체사진을 찍기 위해 카메라 앞에 일렬로 나란히 섰다. 그런데 각자가 원하는 배치가 모두 달라 어떤 순서로 설지 정하는데 시간이 오래 걸렸다. 네오는 프로도와 나란히 서기를 원했고, 튜브가 뿜은 불을 맞은 적이 있던 라이언은 튜브에게서 적어도 세 칸 이상 떨어져서 서기를 원했다. 사진을 찍고 나서 돌아오는 길에, 무지는 모두가 원하는 조건을 만족하면서도 다르게 서는 방법이 있지 않았을까 생각해보게 되었다. 각 프렌즈가 원하는 조건을 입력으로 받았을 때 모든 조건을 만족할 수 있도록 서는 경우의 수를 계산하는 프로그램을 작성해보자.

입력 형식

입력은 조건의 개수를 나타내는 정수 n과 n개의 원소로 구성된 문자열 배열 data로 주어진다. data의 원소는 각 프렌즈가 원하는 조건이 N~F=0과 같은 형태의 문자열로 구성되어 있다. 제한조건은 아래와 같다.

- 1 <= n <= 100
- data의 원소는 다섯 글자로 구성된 문자열이다. 각 원소의 조건은 다음과 같다.
    - 첫 번째 글자와 세 번째 글자는 다음 8개 중 하나이다. {A, C, F, J, M, N, R, T} 각각 어피치, 콘, 프로도, 제이지, 무지, 네오, 라이언, 튜브를 의미한다. 첫 번째 글자는 조건을 제시한 프렌즈, 세 번째 글자는 상대방이다. 첫 번째 글자와 세 번째 글자는 항상 다르다.
    - 두 번째 글자는 항상 ~이다.
    - 네 번째 글자는 다음 3개 중 하나이다. {=, <, >} 각각 같음, 미만, 초과를 의미한다.
    - 다섯 번째 글자는 0 이상 6 이하의 정수의 문자형이며, 조건에 제시되는 간격을 의미한다. 이때 간격은 두 프렌즈 사이에 있는 다른 프렌즈의 수이다.

출력 형식

모든 조건을 만족하는 경우의 수를 리턴한다.

예제 입출력

n	data	answer
2	[N~F=0, R~T>2]	3648
2	[M~C₩<2, C~M>1]	0

예제에 대한 설명

첫 번째 예제는 문제에 설명된 바와 같이, 네오는 프로도와의 간격이 0이기를 원하고 라이언은 튜브와

의 간격이 2보다 크기를 원하는 상황이다.
두 번째 예제는 무지가 콘과의 간격이 2보다 작기를 원하고, 반대로 콘은 무지와의 간격이 1보다 크기를 원하는 상황이다. 이는 동시에 만족할 수 없는 조건이므로 경우의 수는 0이다.

올바른 괄호

괄호 짝짓기

문제 설명

괄호가 바르게 짝지어졌다는 것은 '(' 문자로 열렸으면 반드시 짝지어서 ')' 문자로 닫혀야 한다는 뜻입니다. 예를 들어

- ()() 또는 (())() 는 올바른 괄호입니다.
-)()(또는 (()(는 올바르지 않은 괄호입니다.

'(' 또는 ')' 로만 이루어진 문자열 s가 주어졌을 때, 문자열 s가 올바른 괄호이면 true를 return 하고, 올바르지 않은 괄호이면 false를 return 하는 solution 함수를 완성해 주세요.

제한사항

- 문자열 s의 길이 : 100,000 이하의 자연수
- 문자열 s는 '(' 또는 ')' 로만 이루어져 있습니다.

입출력 예

s	**answer**
()()	true
(())()	true
)()(	false
(()(	false

입출력 예 설명

입출력 예 #1,2,3,4

문제의 예시와 같습니다.

가장 큰 정사각형 찾기

문제 설명

1와 0로 채워진 표(board)가 있습니다. 표 1칸은 1 x 1 의 정사각형으로 이루어져 있습니다. 표에서 1로 이루어진 가장 큰 정사각형을 찾아 넓이를 return 하는 solution 함수를 완성해 주세요. (단, 정사각형이란 축에 평행한 정사각형을 말합니다.)

예를 들어

1	**2**	**3**	**4**
0	1	1	1
1	1	1	1
1	1	1	1
0	0	1	0

가 있다면 가장 큰 정사각형은

1	**2**	**3**	**4**
0	1	1	1
1	1	1	1
1	1	1	1
0	0	1	0

가 되며 넓이는 9가 되므로 9를 반환해 주면 됩니다.

제한사항

- 표(board)는 2차원 배열로 주어집니다.
- 표(board)의 행(row)의 크기 : 1,000 이하의 자연수
- 표(board)의 열(column)의 크기 : 1,000 이하의 자연수
- 표(board)의 값은 1또는 0으로만 이루어져 있습니다.

입출력 예

board	**answer**
[[0,1,1,1],[1,1,1,1],[1,1,1,1],[0,0,1,0]]	9
[[0,0,1,1],[1,1,1,1]]	4

입출력 예 설명

입출력 예 #1

위의 예시와 같습니다.

입출력 예 #2

| 0 | 0 | 1 | 1 |

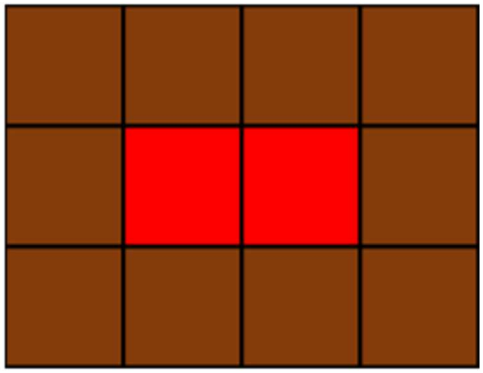
| 1 | 1 | 1 | 1 |

로 가장 큰 정사각형의 넓이는 4가 되므로 4를 return합니다.

카펫

문제 설명

Leo는 카펫을 사러 갔다가 아래 그림과 같이 중앙에는 빨간색으로 칠해져 있고 테두리 1줄은 갈색으로 칠해져 있는 격자 모양 카펫을 봤습니다.



Leo는 집으로 돌아와서 아까 본 카펫의 빨간색과 갈색으로 색칠된 격자의 개수는 기억했지만, 전체 카펫의 크기는 기억하지 못했습니다.

Leo가 본 카펫에서 갈색 격자의 수 brown, 빨간색 격자의 수 red가 매개변수로 주어질 때 카펫의 가로, 세로 크기를 순서대로 배열에 담아 return 하도록 solution 함수를 작성해주세요.

제한사항

- 갈색 격자의 수 brown은 8 이상 5,000 이하인 자연수입니다.
- 빨간색 격자의 수 red는 1 이상 2,000,000 이하인 자연수입니다.
- 카펫의 가로 길이는 세로 길이와 같거나, 세로 길이보다 깁니다.

입출력 예

brown	red	return
10	2	[4, 3]
8	1	[3, 3]
24	24	[8, 6]

출처

※ 공지 - 2020년 2월 3일 테스트케이스가 추가되었습니다.

타겟 넘버

문제 설명

n개의 음이 아닌 정수가 있습니다. 이 수를 적절히 더하거나 빼서 타겟 넘버를 만들려고 합니다. 예를 들어 [1, 1, 1, 1, 1]로 숫자 3을 만들려면 다음 다섯 방법을 쓸 수 있습니다.

-1+1+1+1+1 = 3

+1-1+1+1+1 = 3

+1+1-1+1+1 = 3

+1+1+1-1+1 = 3

+1+1+1+1-1 = 3

사용할 수 있는 숫자가 담긴 배열 numbers, 타겟 넘버 target이 매개변수로 주어질 때 숫자를 적절히 더하고 빼서 타겟 넘버를 만드는 방법의 수를 return 하도록 solution 함수를 작성해주세요.

제한사항

- 주어지는 숫자의 개수는 2개 이상 20개 이하입니다.
- 각 숫자는 1 이상 50 이하인 자연수입니다.
- 타겟 넘버는 1 이상 1000 이하인 자연수입니다.

입출력 예

numbers	**target**	**return**
[1, 1, 1, 1, 1]	3	5

입출력 예 설명

문제에 나온 예와 같습니다.

라면 공장

문제 설명

라면 공장에서는 하루에 밀가루를 1톤씩 사용합니다. 원래 밀가루를 공급받던 공장의 고장으로 앞으로 k일 이후에야 밀가루를 공급받을 수 있기 때문에 해외 공장에서 밀가루를 수입해야 합니다.

해외 공장에서는 향후 밀가루를 공급할 수 있는 날짜와 수량을 알려주었고, 라면 공장에서는 운송비를 줄이기 위해 최소한의 횟수로 밀가루를 공급받고 싶습니다.

현재 공장에 남아있는 밀가루 수량 stock, 밀가루 공급 일정(dates)과 해당 시점에 공급 가능한 밀가루 수량(supplies), 원래 공장으로부터 공급받을 수 있는 시점 k가 주어질 때, 밀가루가 떨어지지 않고 공장을 운영하기 위해서 최소한 몇 번 해외 공장으로부터 밀가루를 공급받아야 하는지를 return 하도록 solution 함수를 완성하세요.

dates[i]에는 i번째 공급 가능일이 들어있으며, supplies[i]에는 dates[i] 날짜에 공급 가능한 밀가루 수량이 들어 있습니다.

제한사항

- stock에 있는 밀가루는 오늘(0일 이후)부터 사용됩니다.
- stock과 k는 2 이상 100,000 이하입니다.
- dates의 각 원소는 1 이상 k 이하입니다.
- supplies의 각 원소는 1 이상 1,000 이하입니다.
- dates와 supplies의 길이는 1 이상 20,000 이하입니다.
- k일 째에는 밀가루가 충분히 공급되기 때문에 k-1일에 사용할 수량까지만 확보하면 됩니다.
- dates에 들어있는 날짜는 오름차순 정렬되어 있습니다.
- dates에 들어있는 날짜에 공급되는 밀가루는 작업 시작 전 새벽에 공급되는 것을 기준으로 합니다. 예를 들어 9일째에 밀가루가 바닥나더라도, 10일째에 공급받으면 10일째에는 공장을 운영할 수 있습니다.
- 밀가루가 바닥나는 경우는 주어지지 않습니다.

입출력 예

stock	dates	supplies	k	result
4	[4,10,15]	[20,5,10]	30	2

입출력 예 설명

- 현재 밀가루가 4톤 남아 있기 때문에 오늘과 1일 후~3일 후까지 사용하고 나면 모든 밀가루를 다 사용합니다. 따라서 4일 후에는 반드시 밀가루를 공급받아야 합니다.
- 4일째 공급받고 나면 15일 이후 아침에는 9톤의 밀가루가 남아있게 되고, 이때 10톤을 더 공급받으면 19톤이 남아있게 됩니다. 15일 이후부터 29일 이후까지 필요한 밀가루는 15톤이므로 더 이상의 공급은 필요 없습니다.
- 따라서 총 2회의 밀가루를 공급받으면 됩니다.

※ 공지 - 2019년 2월 28일 테스트케이스가 추가되었습니다.

위장

문제 설명

스파이들은 매일 다른 옷을 조합하여 입어 자신을 위장합니다.

예를 들어 스파이가 가진 옷이 아래와 같고 오늘 스파이가 동그란 안경, 긴 코트, 파란색 티셔츠를 입었다면 다음날은 청바지를 추가로 입거나 동그란 안경 대신 검정 선글라스를 착용하거나 해야 합니다.

종류	**이름**
얼굴	동그란 안경, 검정 선글라스
상의	파란색 티셔츠
하의	청바지
겉옷	긴 코트

스파이가 가진 의상들이 담긴 2차원 배열 clothes가 주어질 때 서로 다른 옷의 조합의 수를 return 하도록 solution 함수를 작성해주세요.

제한사항

- clothes의 각 행은 [의상의 이름, 의상의 종류]로 이루어져 있습니다.
- 스파이가 가진 의상의 수는 1개 이상 30개 이하입니다.
- 같은 이름을 가진 의상은 존재하지 않습니다.
- clothes의 모든 원소는 문자열로 이루어져 있습니다.
- 모든 문자열의 길이는 1 이상 20 이하인 자연수이고 알파벳 소문자 또는 '_' 로만 이루어져 있습니다.
- 스파이는 하루에 최소 한 개의 의상은 입습니다.

입출력 예

clothes	**return**
[[yellow_hat, headgear], [blue_sunglasses, eyewear], [green_turban, headgear]]	5
[[crow_mask, face], [blue_sunglasses, face], [smoky_makeup, face]]	3

입출력 예 설명

예제 #1

headgear에 해당하는 의상이 yellow_hat, green_turban이고 eyewear에 해당하는 의상이 blue_sunglasses이므로 아래와 같이 5개의 조합이 가능합니다.

1. yellow_hat
2. blue_sunglasses
3. green_turban
4. yellow_hat + blue_sunglasses
5. green_turban + blue_sunglasses

예제 #2

face에 해당하는 의상이 crow_mask, blue_sunglasses, smoky_makeup이므로 아래와 같이 3개의 조합이 가능합니다.

1. crow_mask // 2. blue_sunglasses // 3. smoky_makeup

구명보트

문제 설명

무인도에 갇힌 사람들을 구명보트를 이용하여 구출하려고 합니다. 구명보트는 작아서 한 번에 최대 **2명** 씩 밖에 탈 수 없고, 무게 제한도 있습니다.

예를 들어, 사람들의 몸무게가 [70kg, 50kg, 80kg, 50kg]이고 구명보트의 무게 제한이 100kg이라면 2번째 사람과 4번째 사람은 같이 탈 수 있지만 1번째 사람과 3번째 사람의 무게의 합은 150kg이므로 구명보트의 무게 제한을 초과하여 같이 탈 수 없습니다.

구명보트를 최대한 적게 사용하여 모든 사람을 구출하려고 합니다.

사람들의 몸무게를 담은 배열 people과 구명보트의 무게 제한 limit가 매개변수로 주어질 때, 모든 사람을 구출하기 위해 필요한 구명보트 개수의 최솟값을 return 하도록 solution 함수를 작성해주세요.

제한사항

- 무인도에 갇힌 사람은 1명 이상 50,000명 이하입니다.
- 각 사람의 몸무게는 40kg 이상 240kg 이하입니다.
- 구명보트의 무게 제한은 40kg 이상 240kg 이하입니다.
- 구명보트의 무게 제한은 항상 사람들의 몸무게 중 최댓값보다 크게 주어지므로 사람들을 구출할 수 없는 경우는 없습니다.

입출력 예

people	**limit**	**return**
[70, 50, 80, 50]	100	3
[70, 80, 50]	100	3

전화번호 목록

문제 설명

전화번호부에 적힌 전화번호 중, 한 번호가 다른 번호의 접두어인 경우가 있는지 확인하려 합니다.
전화번호가 다음과 같을 경우, 구조대 전화번호는 영석이의 전화번호의 접두사입니다.

- 구조대 : 119
- 박준영 : 97 674 223
- 지영석 : 11 9552 4421

전화번호부에 적힌 전화번호를 담은 배열 phone_book 이 solution 함수의 매개변수로 주어질 때, 어떤 번호가 다른 번호의 접두어인 경우가 있으면 false를 그렇지 않으면 true를 return 하도록 solution 함수를 작성해주세요.

제한 사항

- phone_book의 길이는 1 이상 1,000,000 이하입니다.
- 각 전화번호의 길이는 1 이상 20 이하입니다.

입출력 예제

phone_book	**return**
[119, 97674223, 1195524421]	false
[123,456,789]	true
[12,123,1235,567,88]	false

입출력 예 설명

입출력 예 #1

앞에서 설명한 예와 같습니다.

입출력 예 #2

한 번호가 다른 번호의 접두사인 경우가 없으므로, 답은 true입니다.

입출력 예 #3

첫 번째 전화번호, “12”가 두 번째 전화번호 “123”의 접두사입니다. 따라서 답은 false입니다.

알림

2019년 5월 13일, 테스트 케이스가 변경되었습니다. 이로 인해 이전에 통과하던 코드가 더 이상 통과하지 않을 수 있습니다.

출처

가장 큰 수

문제 설명

0 또는 양의 정수가 주어졌을 때, 정수를 이어 붙여 만들 수 있는 가장 큰 수를 알아내 주세요.

예를 들어, 주어진 정수가 [6, 10, 2]라면 [6102, 6210, 1062, 1026, 2610, 2106]를 만들 수 있고, 이중 가장 큰 수는 6210입니다.

0 또는 양의 정수가 담긴 배열 numbers가 매개변수로 주어질 때, 순서를 재배치하여 만들 수 있는 가장 큰 수를 문자열로 바꾸어 return 하도록 solution 함수를 작성해주세요.

제한 사항

- numbers의 길이는 1 이상 100,000 이하입니다.
- numbers의 원소는 0 이상 1,000 이하입니다.
- 정답이 너무 클 수 있으니 문자열로 바꾸어 return 합니다.

입출력 예

numbers	**return**
[6, 10, 2]	6210
[3, 30, 34, 5, 9]	9534330

H-Index

문제 설명

H-Index는 과학자의 생산성과 영향력을 나타내는 지표입니다. 어느 과학자의 H-Index를 나타내는 값인 h를 구하려고 합니다. 위키백과[1]에 따르면, H-Index는 다음과 같이 구합니다.

어떤 과학자가 발표한 논문 n편 중, h번 이상 인용된 논문이 h편 이상이고 나머지 논문이 h번 이하 인용되었다면 h의 최댓값이 이 과학자의 H-Index입니다.

어떤 과학자가 발표한 논문의 인용 횟수를 담은 배열 citations가 매개변수로 주어질 때, 이 과학자의 H-Index를 return 하도록 solution 함수를 작성해주세요.

제한사항

- 과학자가 발표한 논문의 수는 1편 이상 1,000편 이하입니다.
- 논문별 인용 횟수는 0회 이상 10,000회 이하입니다.

입출력 예

citations	**return**
[3, 0, 6, 1, 5]	3

입출력 예 설명

이 과학자가 발표한 논문의 수는 5편이고, 그중 3편의 논문은 3회 이상 인용되었습니다. 그리고 나머지 2편의 논문은 3회 이하 인용되었기 때문에 이 과학자의 H-Index는 3입니다.

※ 공지 - 2019년 2월 28일 테스트 케이스가 추가되었습니다.

[2020카카오공채] 괄호 변환

문제 설명

카카오에 신입 개발자로 입사한 **콘**은 선배 개발자로부터 개발역량 강화를 위해 다른 개발자가 작성한 소스 코드를 분석하여 문제점을 발견하고 수정하라는 업무 과제를 받았습니다. 소스를 컴파일하여 로그를 보니 대부분 소스 코드 내 작성된 괄호가 개수는 맞지만 짝이 맞지 않은 형태로 작성되어 오류가 나는 것을 알게 되었습니다.

수정해야 할 소스 파일이 너무 많아서 고민하던 콘은 소스 코드에 작성된 모든 괄호를 뽑아서 올바른 순서대로 배치된 괄호 문자열을 알려주는 프로그램을 다음과 같이 개발하려고 합니다.

용어의 정의

'(' 와 **')'** 로만 이루어진 문자열이 있을 경우, '(' 의 개수와 ')' 의 개수가 같다면 이를 **균형잡힌 괄호 문자열**이라고 부릅니다.

그리고 여기에 '('와 ')'의 괄호의 짝도 모두 맞을 경우에는 이를 **올바른 괄호 문자열**이라고 부릅니다.

예를 들어, "(()))("와 같은 문자열은 균형잡힌 괄호 문자열 이지만 올바른 괄호 문자열은 아닙니다.

반면에 "(())()"와 같은 문자열은 균형잡힌 괄호 문자열 이면서 동시에 올바른 괄호 문자열 입니다.

'(' 와 ')' 로만 이루어진 문자열 w가 균형잡힌 괄호 문자열 이라면 다음과 같은 과정을 통해 올바른 괄호 문자열로 변환할 수 있습니다.

1. 입력이 빈 문자열인 경우, 빈 문자열을 반환합니다.
2. 문자열 w를 두 "균형잡힌 괄호 문자열" u, v로 분리합니다. 단, u는 "균형잡힌 괄호 문자열"로 더 이상 분리할 수 없어야 하며, v는 빈 문자열이 될 수 있습니다.
3. 문자열 u가 "올바른 괄호 문자열" 이라면 문자열 v에 대해 1단계부터 다시 수행합니다.
   3-1. 수행한 결과 문자열을 u에 이어 붙인 후 반환합니다.
4. 문자열 u가 "올바른 괄호 문자열"이 아니라면 아래 과정을 수행합니다.
   4-1. 빈 문자열에 첫 번째 문자로 '('를 붙입니다.
   4-2. 문자열 v에 대해 1단계부터 재귀적으로 수행한 결과 문자열을 이어 붙입니다.
   4-3. ')'를 다시 붙입니다.
   4-4. u의 첫 번째와 마지막 문자를 제거하고, 나머지 문자열의 괄호 방향을 뒤집어서 뒤에 붙입니다.
   4-5. 생성된 문자열을 반환합니다.

균형잡힌 괄호 문자열 p가 매개변수로 주어질 때, 주어진 알고리즘을 수행해 **올바른 괄호 문자열**로 변환한 결과를 return 하도록 solution 함수를 완성해 주세요.

매개변수 설명

- p는 '(' 와 ')' 로만 이루어진 문자열이며 길이는 2 이상 1,000 이하인 짝수입니다.
- 문자열 p를 이루는 '(' 와 ')' 의 개수는 항상 같습니다.
- 만약 p가 이미 올바른 괄호 문자열이라면 그대로 return 하면 됩니다.

입출력 예

p	result
"(()())()"	"(()())()"
")("	"()"

"()))((()"	"()(())()"

입출력 예에 대한 설명

입출력 예 #1

이미 올바른 괄호 문자열 입니다.

입출력 예 #2

- 두 문자열 u, v로 분리합니다.
  - u = ")("
  - v = ""
- u가 올바른 괄호 문자열이 아니므로 다음과 같이 새로운 문자열을 만듭니다.
  - v에 대해 1단계부터 재귀적으로 수행하면 빈 문자열이 반환됩니다.
  - u의 앞뒤 문자를 제거하고, 나머지 문자의 괄호 방향을 뒤집으면 ""이 됩니다.
  - 따라서 생성되는 문자열은 "(" + "" + ")" + ""이며, 최종적으로 "()"로 변환됩니다.

입출력 예 #3

- 두 문자열 u, v로 분리합니다.
  - u = "()"
  - v = "))((()"
- 문자열 u가 올바른 괄호 문자열이므로 그대로 두고, v에 대해 재귀적으로 수행합니다.
- 다시 두 문자열 u, v로 분리합니다.
  - u = "))(("
  - v = "()"
- u가 올바른 괄호 문자열이 아니므로 다음과 같이 새로운 문자열을 만듭니다.
  - v에 대해 1단계부터 재귀적으로 수행하면 "()"이 반환됩니다.
  - u의 앞뒤 문자를 제거하고, 나머지 문자의 괄호 방향을 뒤집으면 "()"이 됩니다.
  - 따라서 생성되는 문자열은 "(" + "()" + ")" + "()"이며, 최종적으로 "(())()"를 반환합니다.
- 처음에 그대로 둔 문자열에 반환된 문자열을 이어 붙이면 "()" + "(())()" = "()(())()"가 됩니다.