Python

Difficulty: **Category:** **Successful Submissions:** 54,478+

Product Sum

Write a function that takes in a "special" array and returns its product sum.

A "special" array is a non-empty array that contains either integers or other "special" arrays. The product sum of a "special" array is the sum of its elements, where "special" arrays inside it are summed themselves and then multiplied by their level of depth.

The depth of a "special" array is how far nested it is. For instance, the depth of `[]` is `1`; the depth of the inner array in `[[]]` is `2`; the depth of the innermost array in `[[[]]]` is `3`.

Therefore, the product sum of `[x, y]` is `x + y`; the product sum of `[x, [y, z]]` is `x + 2 * (y + z)`; the product sum of `[x, [y, [z]]]` is `x + 2 * (y + 3z)`.

Sample Input

```
array = [5, 2, [7, -1], 3, [6, [-13, 8], 4]]
```

Sample Output

```
12 // calculated as: 5 + 2 + 2 * (7 - 1) + 3 + 2 * (6 + 3 * (-13 + 8) + 4)
```

Hints

Hint 1

Try using recursion to solve this problem.

Hint 2

Initialize the product sum of the "special" array to 0. Then, iterate through all of the array's elements; if you come across a number, add it to the product sum; if you come across another "special" array, recursively call the productSum function on it and add the returned value to the product sum. How will you handle multiplying the product sums at a given level of depth?

Prompt **Your Solutions** **Custom Output**