494.Target Sum

494.Target Sum

Medium

4936196Add to ListShare

You are given an integer array `nums` and an integer `target`.

You want to build an **expression** out of nums by adding one of the symbols `'+'` and `'-'` before each integer in nums and then concatenate all the integers.

- For example, if `nums = [2, 1]`, you can add a `'+'` before `2` and a `'-'` before `1` and concatenate them to build the expression `"+2-1"`.

Return the number of different **expressions** that you can build, which evaluates to `target`.

Example 1:

```
Input: nums = [1,1,1,1,1], target = 3
Output: 5
Explanation: There are 5 ways to assign symbols to make the sum of nums be
target 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
+1 + 1 + 1 + 1 - 1 = 3
```

Example 2:

```
Input: nums = [1], target = 1
Output: 1
```

```
def findTargetSumWays(self, nums: List[int], target: int) -> int:
        setSum = sum(nums)
        if target>setSum:
            return 0
        if(target+setSum)%2!=0 or target+setSum<0:
            return 0
        set1 = (target+setSum)//2
        n = len(nums)
        m = set1
        dp = [[0]*(set1+1) for i in range(n+1)]
        for i in range(n+1):
            for j in range(m+1):
                if i==0 and j==0:
                    dp[i][j]=1
                elif i==0 and j!=0:
                    dp[i][j]=0
                elif j==0:
                    dp[i][j]=1
                else:
                    tar = nums[i-1]
                    if j-tar>=0 and tar!=0:
                        dp[i][j] = dp[i-1][j] + dp[i-1][j-tar]
                    else:
                        dp[i][j] = dp[i-1][j]
        p = nums.count(0)
        return pow(2,p)*dp[n][m]
```

It is very important as we are not having direct implementation here.Just that we are multiplying our answer with $2^{\#\text{ of zeroes in the array}}$

This is because the 0 doesn't affect the sum and we can add +0 or -0 that is 2 ways. So for every subsequent 0 we multiply the answer with 2 or we can also fo it by multplyin the answer with 2 raise to power number of zerpes.