1769. Minimum Number of Operations to Move All Balls to Each Box

Solved

Medium | Topics | Companies | Hint

You have `n` boxes. You are given a binary string `boxes` of length `n`, where `boxes[i]` is `'0'` if the `i^th` box is **empty**, and `'1'` if it contains **one** ball.

In one operation, you can move **one** ball from a box to an adjacent box. Box `i` is adjacent to box `j` if `abs(i - j) == 1`. Note that after doing so, there may be more than one ball in some boxes.

Return an array `answer` of size `n`, where `answer[i]` is the **minimum** number of operations needed to move all the balls to the `i^th` box.

Each `answer[i]` is calculated considering the **initial** state of the boxes.

Example 1:

```
Input: boxes = "110"
Output: [1,1,3]
Explanation: The answer for each box is as follows:
1) First box: you will have to move one ball from the second box to the first box in one operation.
2) Second box: you will have to move one ball from the first box to the second box in one operation.
3) Third box: you will have to move one ball from the first box to the third box in two operations, and move one ball from the second box to the third box in one operation.
```

Python3 | Auto

```python
class Solution:
    def minOperations(self, boxes: str) -> List[int]:
        n = len(boxes)
        ones = []
        res = [0] * n
        for i in range(n):
            if boxes[i] == '1':
                ones.append(i)

        for i in range(n):
            for ball in ones:
                res[i] += abs(ball - i)

        return res
```

Saved

Testcase | Test Result

Accepted Runtime: 0 ms

• Case 1 • Case 2

Input

boxes =

"110"

But, could be better using prefix & surfix.

```python
class Solution:
    def minOperations(self, boxes: str) -> List[int]:
        answer = []
        pref = p = s = 0
        for i, el in enumerate(boxes):
            if el == '1':
                pref += i
                p += 1
        for el in boxes:
            answer.append(pref)
            if el == '1':
                p -= 1
                s += 1
            pref = pref - p + s
        return answer
```

Eg. boxes = "001011"

→ after this for loop:

pref = 2 + 4 + 5 = 11

p = 3