1872. Stone Game VIII

Description

Alice and Bob take turns playing a game, with **Alice starting first**.

There are `n` stones arranged in a row. On each player's turn, while the number of stones is **more than one**, they will do the following:

1. Choose an integer `x > 1`, and **remove** the leftmost `x` stones from the row.
2. Add the **sum** of the **removed** stones' values to the player's score.
3. Place a **new stone**, whose value is equal to that sum, on the left side of the row.

The game stops when **only** **one** stone is left in the row.

The **score difference** between Alice and Bob is `(Alice's score - Bob's score)`. Alice's goal is to **maximize** the score difference, and Bob's goal is the **minimize** the score difference.

Given an integer array `stones` of length `n` where `stones[i]` represents the value of the `i th` stone **from the left**, return *the* ***score difference*** *between Alice and Bob if they both play* ***optimally***.

Example 1:

```
Input: stones = [-1,2,-3,4,-5]
Output: 5
Explanation:
- Alice removes the first 4 stones, adds (-1) + 2 + (-3) + 4 = 2 to her score, and places a stone of
  value 2 on the left. stones = [2,-5].
- Bob removes the first 2 stones, adds 2 + (-5) = -3 to his score, and places a stone of value -3 on
  the left. stones = [-3].
The difference between their scores is 2 - (-3) = 5.
```

Example 2:

```
Input: stones = [7,-6,5,10,5,-2,-6]
Output: 13
Explanation:
- Alice removes all stones, adds 7 + (-6) + 5 + 10 + 5 + (-2) + (-6) = 13 to her score, and places a
  stone of value 13 on the left. stones = [13].
The difference between their scores is 13 - 0 = 13.
```

Example 3:

```
Input: stones = [-10,-12]
Output: -22
Explanation:
- Alice can only make one move, which is to remove both stones. She adds (-10) + (-12) = -22 to her
  score and places a stone of value -22 on the left. stones = [-22].
The difference between their scores is (-22) - 0 = -22.
```

Constraints:

- `n == stones.length`
- `2 <= n <= 10 5`
- `-10 4 <= stones[i] <= 10 4`