Binary Search I #2

167. Two Sum II - Input Array Is Sorted

Given a **1-indexed** array of integers `numbers` that is already ***sorted in non-decreasing order***, find two numbers such that they add up to a specific `target` number. Let these two numbers be `numbers[index 1]` and `numbers[index 2]` where `1 <= index 1 < index 2 <= numbers.length`.

Return *the indices of the two numbers,* `index 1` *and* `index 2`*,* ***added by one*** *as an integer array* `[index 1, index 2]` *of length 2.*

The tests are generated such that there is **exactly one solution**. You **may not** use the same element twice.

Your solution must use only constant extra space.

Example 1:

```
Input: numbers = [2,7,11,15], target = 9
Output: [1,2]
Explanation: The sum of 2 and 7 is 9. Therefore, index1 = 1, index2 = 2. We return [1, 2].
```

Example 2:

```
Input: numbers = [2,3,4], target = 6
Output: [1,3]
Explanation: The sum of 2 and 4 is 6. Therefore index1 = 1, index2 = 3. We return [1, 3].
```

Example 3:

```
Input: numbers = [-1,0], target = -1
Output: [1,2]
Explanation: The sum of -1 and 0 is -1. Therefore index1 = 1, index2 = 2. We return [1, 2].
```

Constraints:

- `2 <= numbers.length <= 3 * 10 4`
- `1000 <= numbers[i] <= 1000`
- `numbers` is sorted in **non-decreasing order**.
- `1000 <= target <= 1000`
- The tests are generated such that there is **exactly one solution**.

1608. Special Array With X Elements Greater Than or Equal X

You are given an array `nums` of non-negative integers. `nums` is considered **special** if there exists a number `x` such that there are **exactly** `x` numbers in `nums` that are **greater than or equal to** `x`.

Notice that `x` **does not** have to be an element in `nums`.

Return `x` *if the array is* ***special****, otherwise, return* `-1`. It can be proven that if `nums` is special, the value for `x` is **unique**.

Example 1:

```
Input: nums = [3,5]
Output: 2
Explanation: There are 2 values (3 and 5) that are greater than or equal to 2.
```

Example 2:

```
Input: nums = [0,0]
Output: -1
Explanation: No numbers fit the criteria for x.
If x = 0, there should be 0 numbers >= x, but there are 2.
If x = 1, there should be 1 number >= x, but there are 0.
If x = 2, there should be 2 numbers >= x, but there are 0.
x cannot be greater since there are only 2 numbers in nums.
```

Example 3:

```
Input: nums = [0,4,3,0,4]
Output: 3
Explanation: There are 3 values that are greater than or equal to 3.
```

Constraints:

- `1 <= nums.length <= 100`
- `0 <= nums[i] <= 1000`

1351. Count Negative Numbers in a Sorted Matrix

Given a `m x n` matrix `grid` which is sorted in non-increasing order both row-wise and column-wise, return *the number of **negative** numbers in* `grid`.

Example 1:

```
Input: grid = [[4,3,2,-1],[3,2,1,-1],[1,1,-1,-2],[-1,-1,-2,-3]]
Output: 8
Explanation: There are 8 negatives number in the matrix.
```

Example 2:

```
Input: grid = [[3,2],[1,0]]
Output: 0
```

Constraints:

- `m == grid.length`
- `n == grid[i].length`
- `1 <= m, n <= 100`
- `100 <= grid[i][j] <= 100`

Follow up:

Could you find an

```
O(n + m)
```

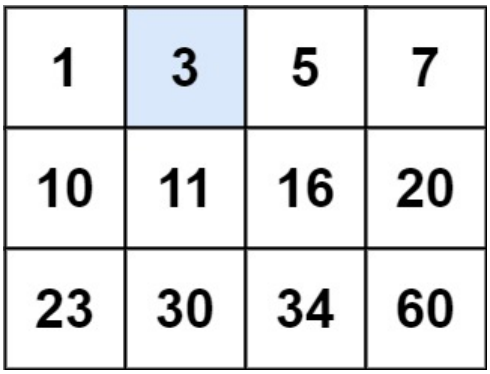
solution?

74. Search a 2D Matrix

Write an efficient algorithm that searches for a value `target` in an `m x n` integer matrix `matrix`. This matrix has the following properties:

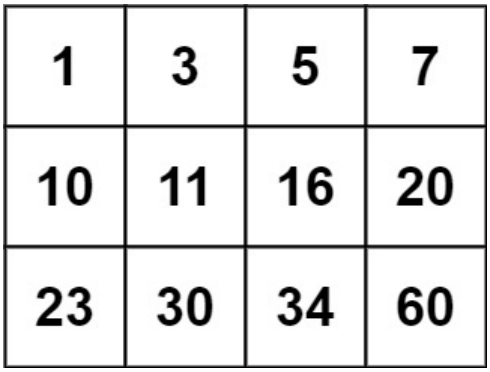
- Integers in each row are sorted from left to right.
- The first integer of each row is greater than the last integer of the previous row.

Example 1:




```
Input: matrix = [[1,3,5,7],[10,11,16,20],[23,30,34,60]], target = 3
Output: true
```

Example 2:




```
Input: matrix = [[1,3,5,7],[10,11,16,20],[23,30,34,60]], target = 13
Output: false
```

Constraints:

- `m == matrix.length`
- `n == matrix[i].length`
- `1 <= m, n <= 100`
- `10 4 <= matrix[i][j], target <= 10 4`

1337. The K Weakest Rows in a Matrix

You are given an `m x n` binary matrix `mat` of `1`'s (representing soldiers) and `0`'s (representing civilians). The soldiers are positioned **in front** of the civilians. That is, all the `1`'s will appear to the **left** of all the `0`'s in each row.

A row `i` is **weaker** than a row `j` if one of the following is true:

- The number of soldiers in row `i` is less than the number of soldiers in row `j`.
- Both rows have the same number of soldiers and `i < j`.

Return *the indices of the* `k` ***weakest*** *rows in the matrix ordered from weakest to strongest*.

Example 1:

```
Input: mat =
[[1,1,0,0,0],
 [1,1,1,1,0],
 [1,0,0,0,0],
 [1,1,0,0,0],
 [1,1,1,1,1]],
k = 3
Output: [2,0,3]
Explanation:
The number of soldiers in each row is:
- Row 0: 2
- Row 1: 4
- Row 2: 1
- Row 3: 2
- Row 4: 5
The rows ordered from weakest to strongest are [2,0,3,1,4].
```

Example 2:

```
Input: mat =
[[1,0,0,0],
 [1,1,1,1],
 [1,0,0,0],
 [1,0,0,0]],
k = 2
Output: [0,2]
Explanation:
The number of soldiers in each row is:
- Row 0: 1
- Row 1: 4
- Row 2: 1
- Row 3: 1
The rows ordered from weakest to strongest are [0,2,3,1].
```

Constraints:

- `m == mat.length`
- `n == mat[i].length`
- `2 <= n, m <= 100`
- `1 <= k <= m`
- `matrix[i][j]` is either 0 or 1.

1346. Check If N and Its Double Exist

Given an array `arr` of integers, check if there exists two integers `N` and `M` such that `N` is the double of `M` (i.e. `N = 2 * M`).

More formally check if there exists two indices `i` and `j` such that :

- `i != j`
- `0 <= i, j < arr.length`
- `arr[i] == 2 * arr[j]`

Example 1:

```
Input: arr = [10,2,5,3]
Output: true
Explanation: N = 10 is the double of M = 5,that is,10 = 2 * 5.
```

Example 2:

```
Input: arr = [7,1,14,11]
Output: true
Explanation: N = 14 is the double of M = 7,that is,14 = 2 * 7.
```

Example 3:

```
Input: arr = [3,1,7,11]
Output: false
Explanation: In this case does not exist N and M, such that N = 2 * M.
```

Constraints:

- `2 <= arr.length <= 500`
- `10^3 <= arr[i] <= 10^3`

350. Intersection of Two Arrays II

Given two integer arrays `nums1` and `nums2`, return *an array of their intersection*. Each element in the result must appear as many times as it shows in both arrays and you may return the result in **any order**.

Example 1:

```
Input: nums1 = [1,2,2,1], nums2 = [2,2]
Output: [2,2]
```

Example 2:

```
Input: nums1 = [4,9,5], nums2 = [9,4,9,8,4]
Output: [4,9]
Explanation: [9,4] is also accepted.
```

Constraints:

- `1 <= nums1.length, nums2.length <= 1000`
- `0 <= nums1[i], nums2[i] <= 1000`

Follow up:

- What if the given array is already sorted? How would you optimize your algorithm?
- What if `nums1`'s size is small compared to `nums2`'s size? Which algorithm is better?

- What if elements of `nums2` are stored on disk, and the memory is limited such that you cannot load all elements into the memory at once?

633. Sum of Square Numbers

Given a non-negative integer `c`, decide whether there're two integers `a` and `b` such that `a2 + b2 = c`.

Example 1:

```
Input: c = 5
Output: true
Explanation: 1 * 1 + 2 * 2 = 5
```

Example 2:

```
Input: c = 3
Output: false
```

Constraints:

- `0 <= c <= 231 - 1`