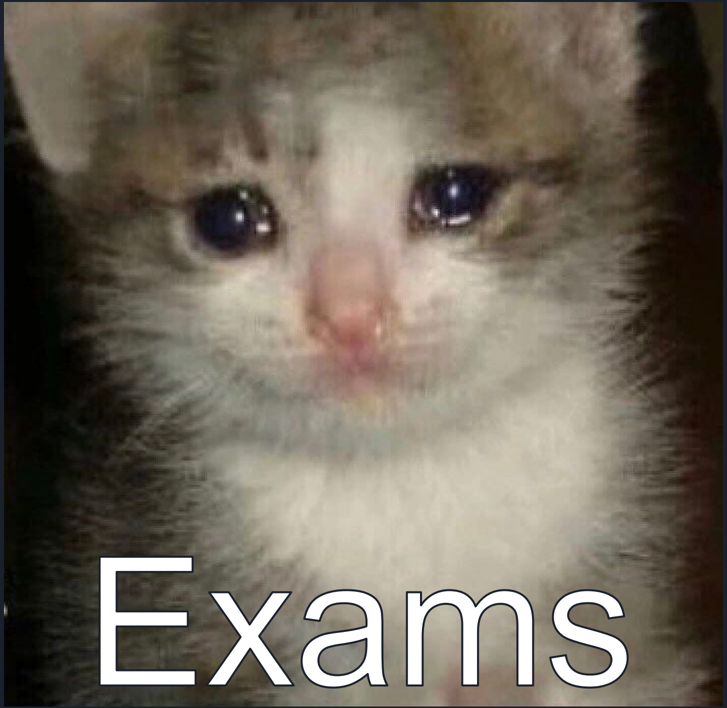
CS 314 Discussion

Problems

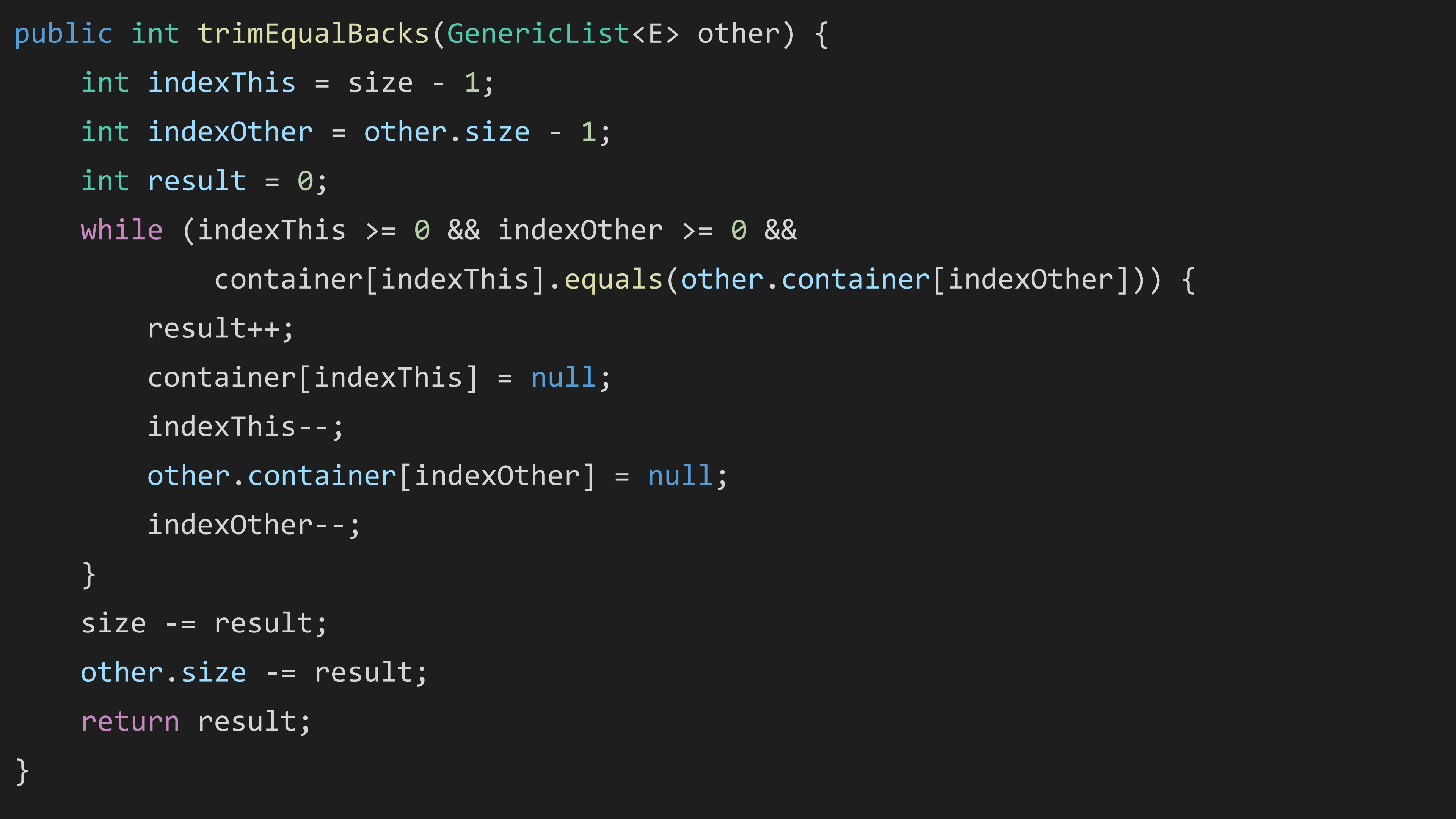
- Main Problem
  - GenericList
- Extra Problem
  - Extra Data Structure



fr this time though

Exam Stuff

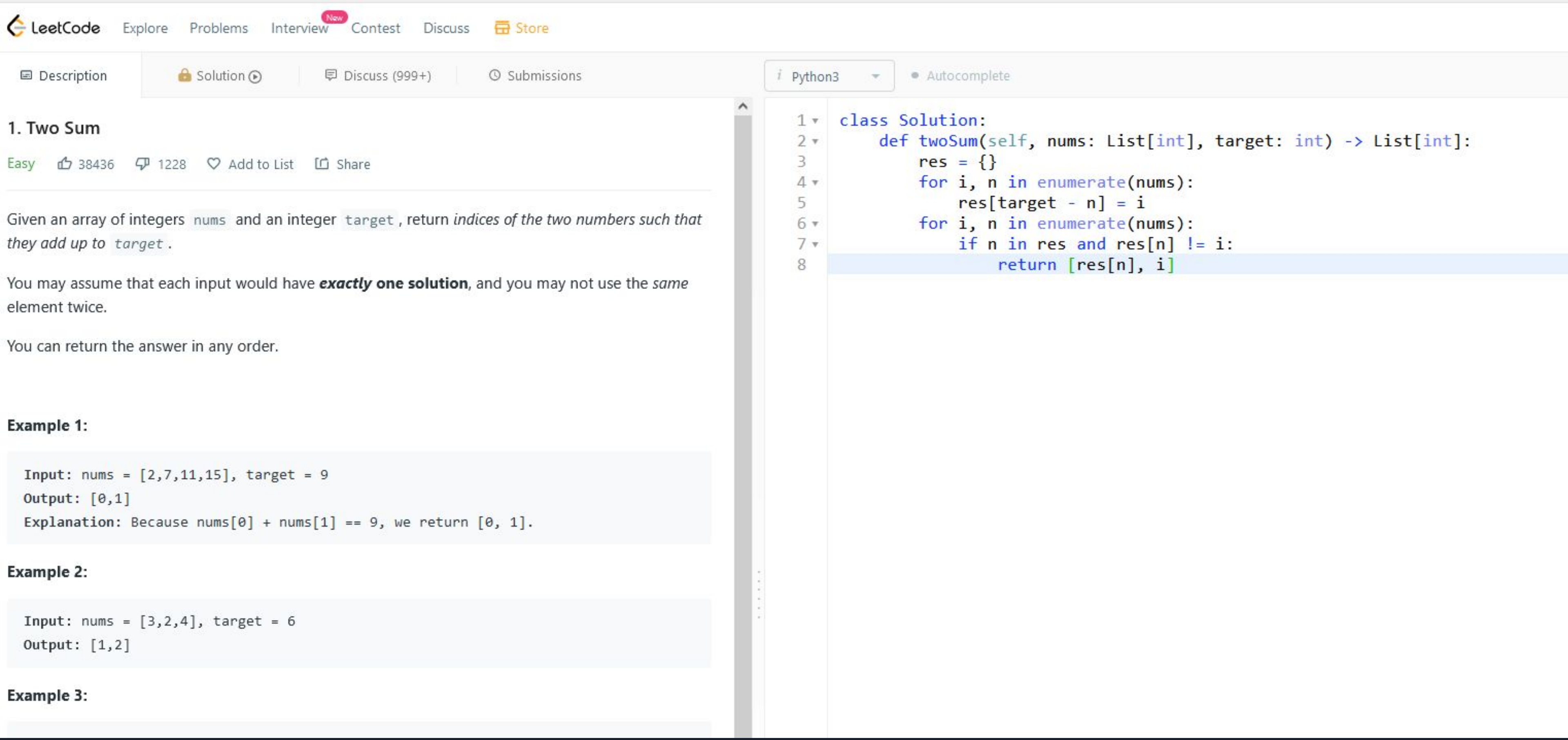
- Exam Structure
  - 25 Short Answer
  - 3 Coding Questions
- No questions during exams. Write down your assumptions if confused
  - Doesn’t mean we’ll grade based on your assumption
- Keep note of time. A fair # of students don’t attempt some coding questions



```java
public int trimEqualBacks(GenericList<E> other) {
    int indexThis = size - 1;
    int indexOther = other.size - 1;
    int result = 0;
    while (indexThis >= 0 && indexOther >= 0 &&
            container[indexThis].equals(other.container[indexOther])) {
        result++;
        container[indexThis] = null;
        indexThis--;
        other.container[indexOther] = null;
        indexOther--;
    }
    size -= result;
    other.size -= result;
    return result;
}
```

```java
public int trimEqualBacks(GenericList<E> other) {
    int oldSize = size;
    while (size > 0 && other.size > 0 &&
            container[size - 1].equals(other.container[other.size - 1])) {
        container[indexThis] = null;
        other.container[indexOther] = null;
        size--;
        other.size--;
    }
    return oldSize - size;
}
```

```java
public boolean removeSingleOccurrence(Object tgt) {
    for (int i = 0; i < numberOfElements; i++) {
        if (tgt.equals(elements[i]) {
            numberOfElements--;
            elements[i] = elements[numberOfElements];
            elements[numberOfElements] = null;
            return true;
        }
    }
    return false;
}
```



1. Two Sum

Given an array of integers `nums` and an integer `target`, return *indices of the two numbers such that they add up to* `target`.

You may assume that each input would have ***exactly* one solution**, and you may not use the *same* element twice.

You can return the answer in any order.

Example 1:

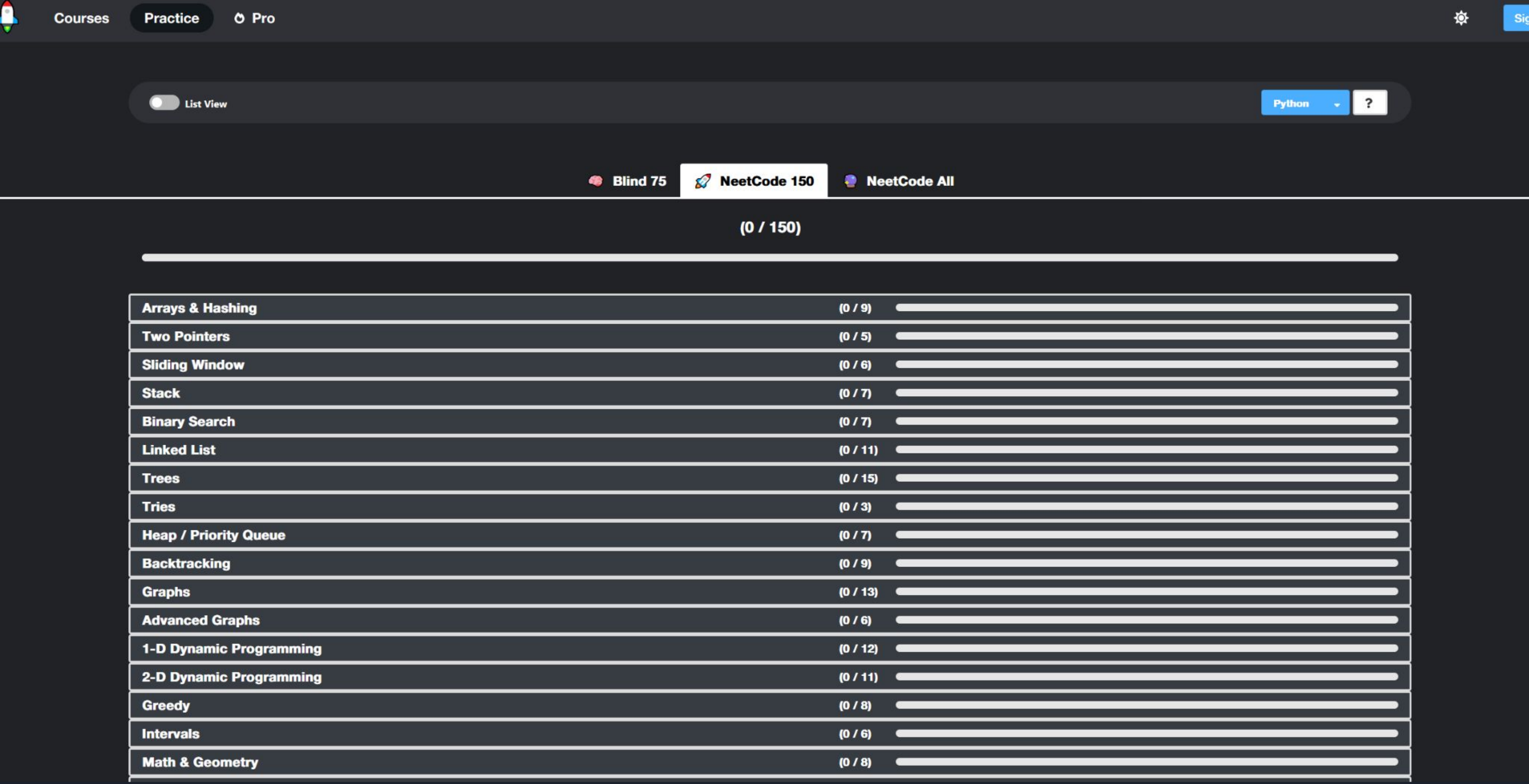
```
Input: nums = [2,7,11,15], target = 9
Output: [0,1]
Explanation: Because nums[0] + nums[1] == 9, we return [0, 1].
```

Example 2:

```
Input: nums = [3,2,4], target = 6
Output: [1,2]
```

Example 3:

```python
class Solution:
    def twoSum(self, nums: List[int], target: int) -> List[int]:
        res = {}
        for i, n in enumerate(nums):
            res[target - n] = i
        for i, n in enumerate(nums):
            if n in res and res[n] != i:
                return [res[n], i]
```

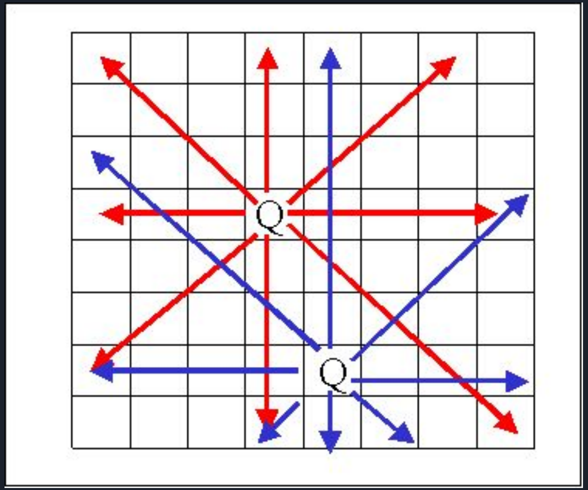


(0 / 150)

Arrays & Hashing	(0 / 9)
Two Pointers	(0 / 5)
Sliding Window	(0 / 6)
Stack	(0 / 7)
Binary Search	(0 / 7)
Linked List	(0 / 11)
Trees	(0 / 15)
Tries	(0 / 3)
Heap / Priority Queue	(0 / 7)
Backtracking	(0 / 9)
Graphs	(0 / 13)
Advanced Graphs	(0 / 6)
1-D Dynamic Programming	(0 / 12)
2-D Dynamic Programming	(0 / 11)
Greedy	(0 / 8)
Intervals	(0 / 6)
Math & Geometry	(0 / 8)

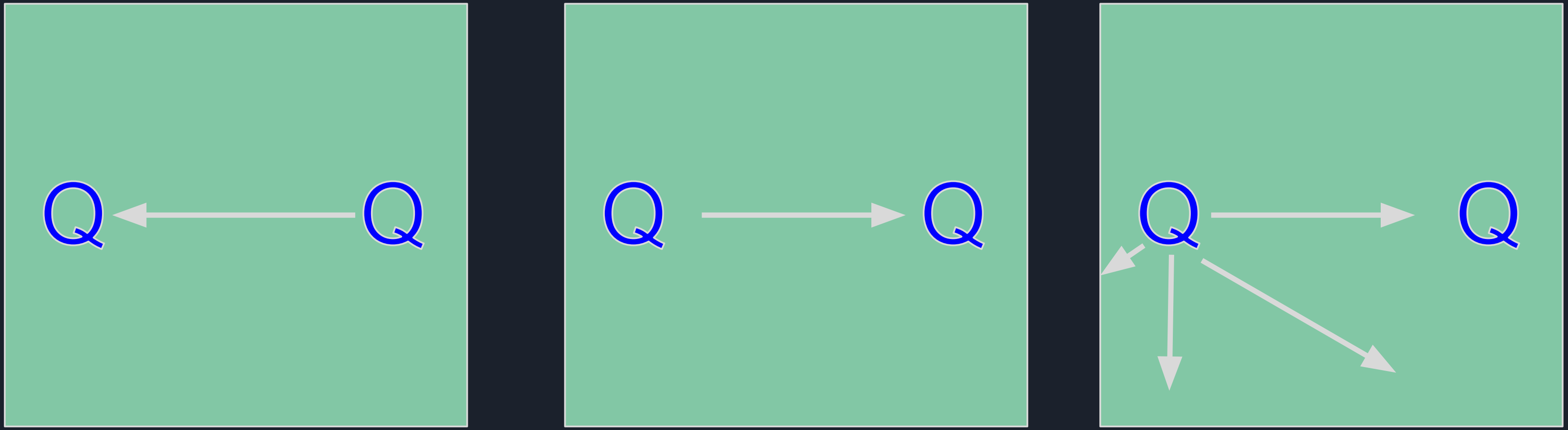
QueensAreSafe

- Relevant Directions
- Parameterized Row/Col Solutions
- Slope Method



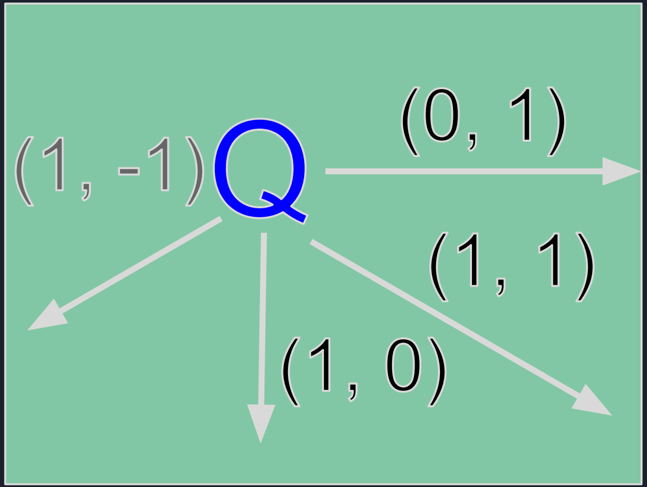
Directions

- Only need to check 4 directions



Directions

- You can use an array to store the different changes in rows and columns



rows = {0, 1, 1, 1}
cols = {1, 1, 0, -1}