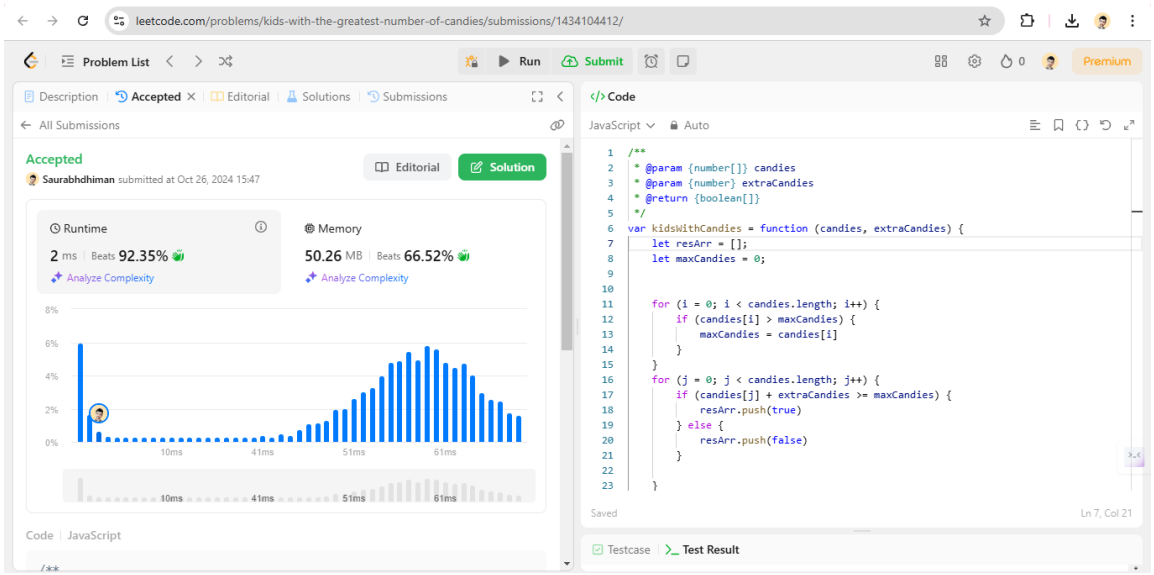
My leetcode profile link - https://leetcode.com/u/Saurabhdhiman/

QnO.- 1431

Link - https://leetcode.com/problems/kids-with-the-greatest-number-of-candies/description/
solution link- https://leetcode.com/problems/kids-with-the-greatest-number-of-candies/submissions/1434112605/



Description -

Time Complexicity-

For first loop = O(n)

For second loop = O(n)

so, O(n) + O(n) = O(2n)

Time complexicity = **O(n)** because we dont consider 2.

Space Complexicity -

resArr will take n number of boolen values -- O(n)

i,j, maxCandies will take -- O(1)

So Space Complexity = **O(n).**

Code Explanation -

1. Initialized a empty resArr which will store boolean values.

2. maxCandies initialized to 0 so that it will store max cndies any kid have.

3. First for loop will iterate thrugh candies array to find max no of candies and it will update maxCandies.

4. Second for loop will check all candies and add extra candies to that and compare with maxCandies.

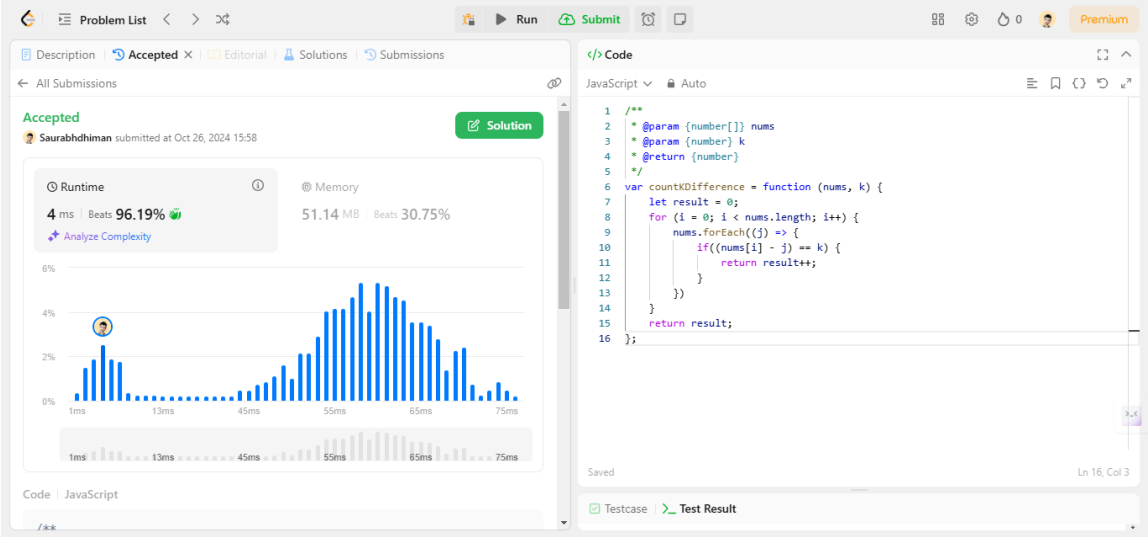
If extra candies + current candies a kid have greater than or equal max candies push true to result array else push false to result aray

5. Return result array containing boolean values.

QNO.- 2006

Link - https://leetcode.com/problems/count-number-of-pairs-with-absolute-difference-k/description/

Solution Link - https://leetcode.com/problems/count-number-of-pairs-with-absolute-difference-k/submissions/1434111053/



Description -

Time Complexicity - As we are using two loops and second loop is present inside of first loop so both outer and inner loop will itterate n times.

$O(n) \times O(n) = O(n^2)$

Time complexicity = **$O(n^2)$**

Space complexicity -

result,i,j will take only one space = O(1).

so Space Complexicity = **O(1).**

Code Explanation -

1. Initialized a variable result with value 0 which will count and store our value.

2. In for loop i am iterating through nums array to get all the values of aray.

3. applying forEach loop on nums array so that all all eleemsnts will be itteated in j value.

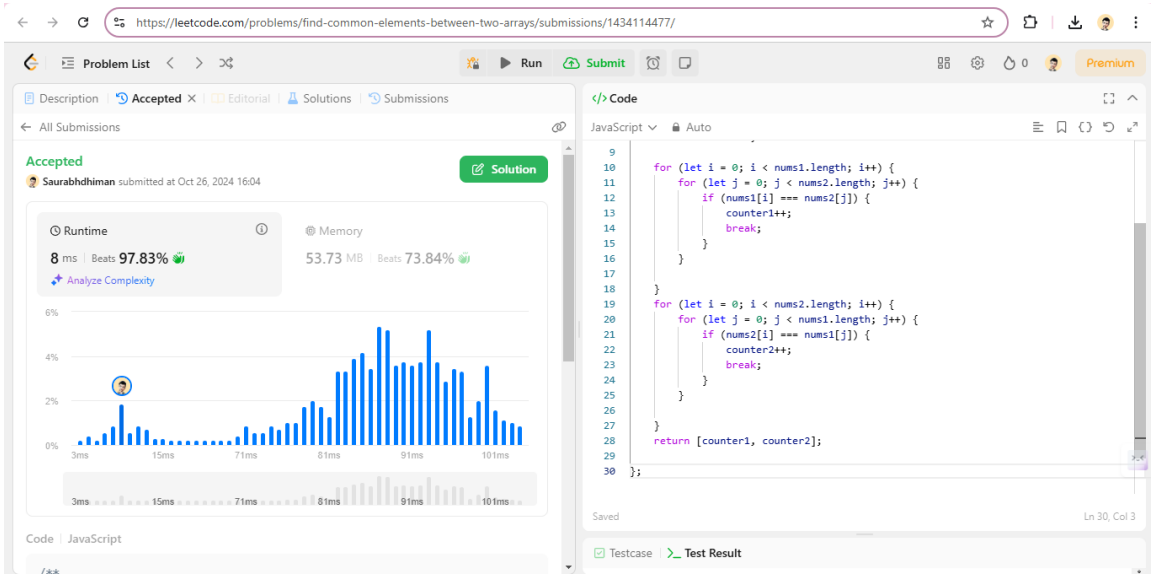
4. appliying codindition to check difference between nums array and j and i it is equal to k increase our counter i.e result.

5. Return result with counted values.

QNO.- 2956

Link - https://leetcode.com/problems/find-common-elements-between-two-arrays/description/

solution link - https://leetcode.com/problems/find-common-elements-between-two-arrays/submissions/1434114477/



Description -

Time Complexicity-

we are using two nested for loop so,

for first nested for loop - $O(n^2)$

for second nested for loop - $O(n^2)$

Time complexicity = $O(n^2) + O(n^2) =$ **$O(n^2)$**

Space complexicity-

counter1, counter2, i, j will take only one space i.e O(1)

so space complexicity = **O(1)**

Code Explanation -

1. First taken two variables namely counter1 and counter2 that will act as counter for increasing count.

2.now used a nested for loop 1st loop will iterates nums1 array and second loop will iterate nums2 array.

3. if elements of nums 1 array = to nums2 array increase the counter.

4. Used break so that number will count once only even it repeat in num2 array.

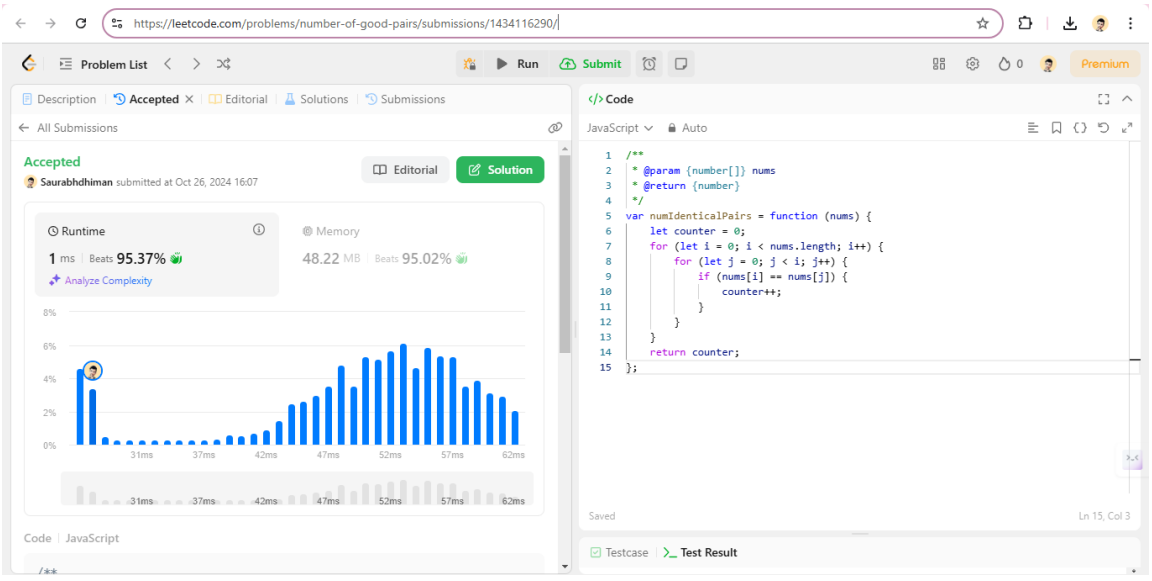
5. Now used again a nested for loop and this time changed such that nums2values = nums1 value incerase counter and a break statement for same reason.

6. Returned the counter1 and counter 2 in array.

QNO.- 1512

Link - https://leetcode.com/problems/number-of-good-pairs/description/

solution link- https://leetcode.com/problems/number-of-good-pairs/submissions/1434116290/



Description -

Time Complexicity-

As we are using two nested for loop so **O(n^2)**

Space Complexicity -

Counter,i,j - O(1)

so space complexicity will be **O(1).**

Code Explanation -

1. Taking a counter variable which will count the pair.

2. using nested for loop such that 1st loop iterate through nums arra and second loop also iterates through nums array .

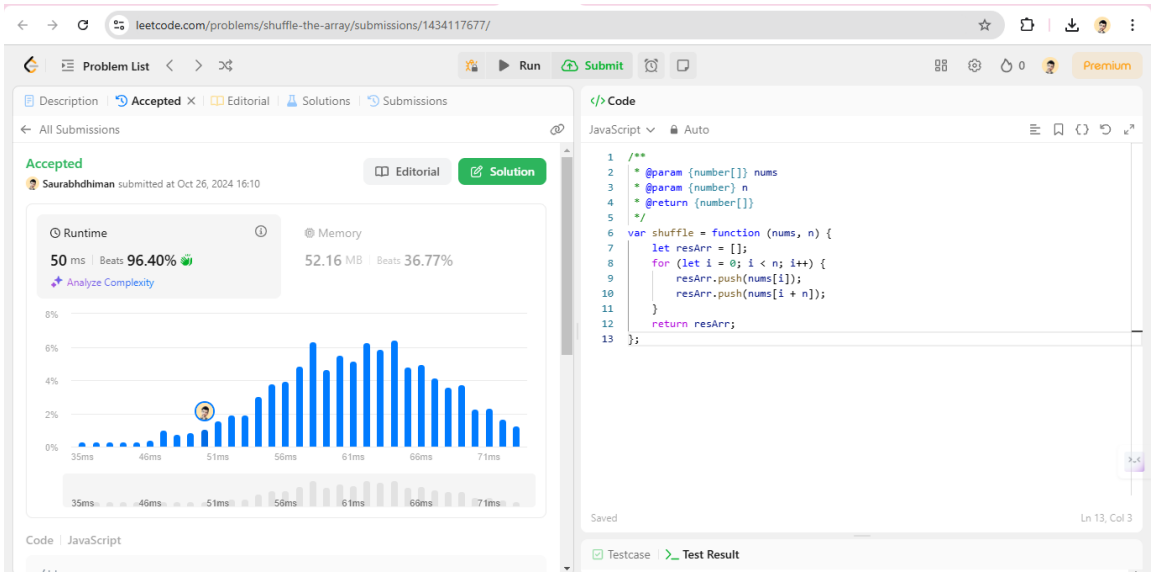
3. Adding if condition such that value of nums at index i == value of inex j so increase the counter.

4. Return the counter.

QnO.- 1470

Link - https://leetcode.com/problems/shuffle-the-array/description/

solution link - https://leetcode.com/problems/shuffle-the-array/submissions/1434117677/



Description -

Time Complexicity-

As we are using one for loop so **O(n).**

Space Complexicity -

USing empty arr which will take size of array so

O(n).

Code Explanation-

1. Defined a empty array which will take result of suffeled array.

2. For loop which will iterates upto n (as we need to store values such that we have to iterate only hal of aray.

3. Now when loop run for first time at 0 index it will push 0 hen 0+n at 1inde and so on upto n.

4. Return result array.