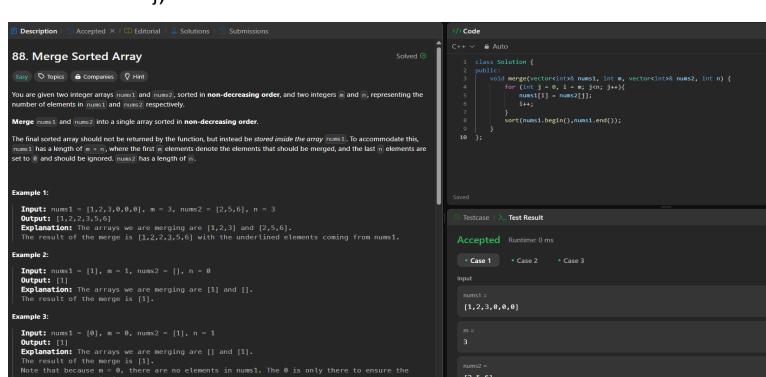
ASSIGNMENT 5

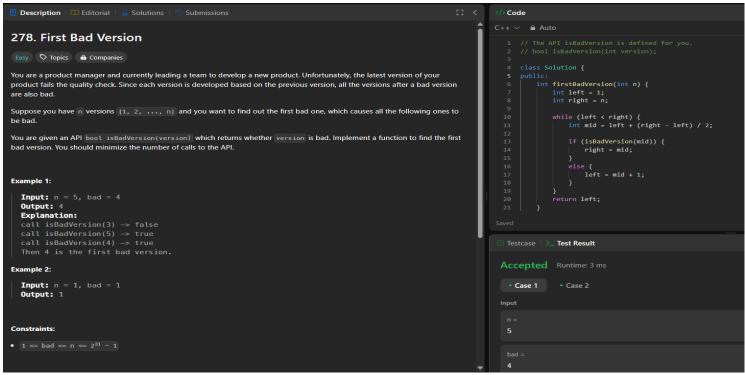
Name - Dhruv Bhatia UID - 22BCS12196 Section - IOT-608/B

Merge Sorted Array

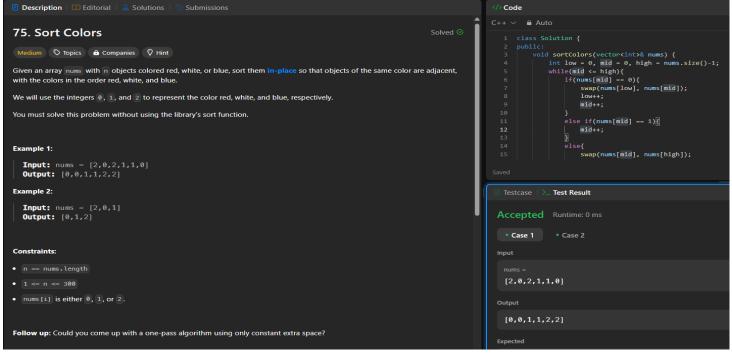
```
class Solution {
public:
    void merge(vector<int>& nums1, int m,
vector<int>& nums2, int n) {
    for (int j = 0, i = m; j<n; j++){
        nums1[i] = nums2[j];
        i++;
    }
    sort(nums1.begin(),nums1.end());
}
</pre>
```



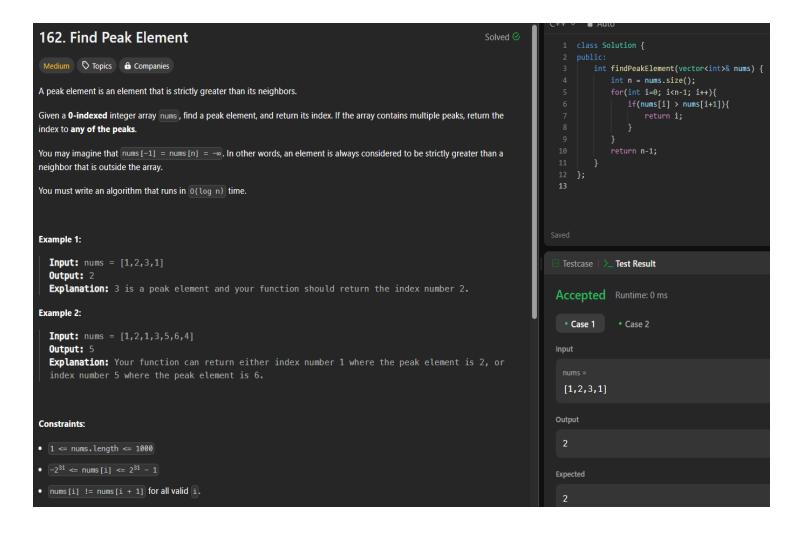
```
class Solution {
public:
  int firstBadVersion(int n) {
     int left = 1;
     int right = n;
     while (left < right) {
       int mid = left + (right - left) / 2;
       if (isBadVersion(mid)) {
          right = mid;
       else {
          left = mid + 1;
     return left;
```



```
class Solution {
public:
  void sortColors(vector<int>& nums) {
    int low = 0, mid = 0, high = nums.size()-1;
    while(mid <= high){
      if(nums[mid] == 0){
         swap(nums[low], nums[mid]);
         low++;
         mid++;
      else if(nums[mid] == 1){
         mid++;
      else{
         swap(nums[mid], nums[high]);
         high--;
```



```
class Solution {
  public:
    int findPeakElement(vector<int>& nums) {
      int n = nums.size();
      for(int i=0; i<n-1; i++){
        if(nums[i] > nums[i+1]){
          return i;
      }
    }
    return n-1;
  }
};
```



```
class Solution {
public:
  double findMedianSortedArrays(vector<int>&
nums1, vector<int>& nums2) {
    vector<int>v;
    for(auto num:nums1)
      v.push_back(num);
    for(auto num:nums2)
      v.push_back(num);
    sort(v.begin(),v.end());
    int n=v.size();
    return n%2?v[n/2]:(v[n/2-1]+v[n/2])/2.0;
};
```

