

Arrays, Strings & Linked Lists Lecture 2

Sunday, 21 July 2024 3:02 PM

2-Sum Problem

$$a = [2, 7, 11, 15] \quad k = 9$$

↓ ↓ ↓

$$a = [-1, 3, -2, -6, 10, 5, -3] \quad k = 3$$

0, 1, 2, 3, 4, 5, 6

$[-2, 5] \leftarrow \text{values}$
 $[2, 5] \leftarrow \text{index}$

sorted $O(n \log n)$

$$a = [-6, -3, -2, -1, 3, 5, 10]$$

$O(n) \rightarrow$

↑ ↑ ↑ ↑ ↑

i p l i j

$i \rightarrow j$

Sum
move left ↑
move right ↓

$$-6 + 10 = 4$$

$$-6 + 5 = -1$$

$$-3 + 5 = 2$$

$$-2 + 5 = \boxed{3}$$

left (i) smallest → largest
right (j) largest → smallest

<https://leetcode.com/problems/two-sum/>

```
class Solution:
    def twoSum(self, nums: List[int], target: int) -> List[int]:
        for i in range(len(nums)):
            nums[i] = (nums[i], i)
        nums.sort()
        i = 0
        j = len(nums)-1
        while i<j:
            if nums[i][0] + nums[j][0] > target:
                j -= 1
            elif nums[i][0] + nums[j][0] < target:
                i += 1
            else:
                return [nums[i][1], nums[j][1]]
        return []
```

```

class Solution {
public:
    vector<int> twoSum(vector<int>& nums, int target) {
        vector<pair<int, int>> vals;
        for(int i=0; i<nums.size(); i++) {
            vals.push_back({nums[i], i});
        }
        sort(vals.begin(), vals.end());
        int i=0, j=vals.size()-1;
        while(i<j) {
            if(vals[i].first + vals[j].first < target) i++;
            else if(vals[i].first + vals[j].first > target) j--;
            else return vector<int> {vals[i].second, vals[j].second};
        }
        return vector<int>{};
    }
};

```

$O(n \log n + n)$

==x==

3-Sum
 4-Sum

$O(n \log n + n^2)$
 $O(n \log n + n^3)$

==x==

Dutch National Flag Algorithm

Problem:- Sort an array containing 3 distinct values (which can repeat).

eg. AAABACABCC
{AAAAABBBCC}

0, 0, 1, 2, 0, 1, 1, 2, 0, 1.
↳ {0, 0, 0, 0, 1, 1, 1, 1, 2, 2}

Sort $\rightarrow n \log n$

Map-frequency (count of 0, 1, 2) :- $T = O(n)$, $S = O(1)$

Optimal :-

{ 2, 1, 0, 0, 1, 1, 0, 2, 0, 1, 2, 1 }

{ 0, 0, 0, 0, 1, 1, 1, 1, 1, 2, 2, 2 }

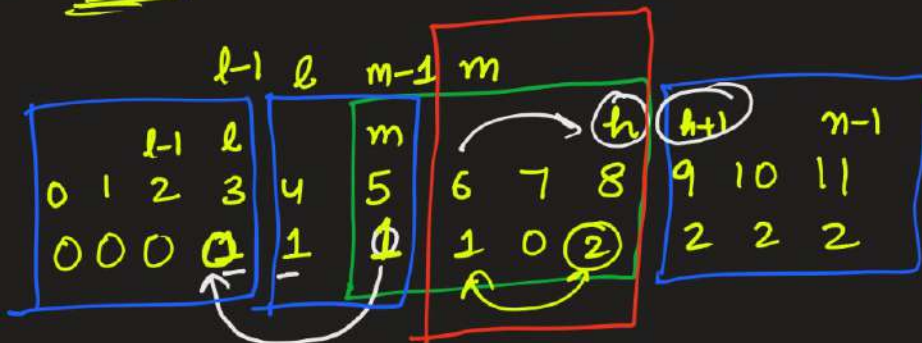
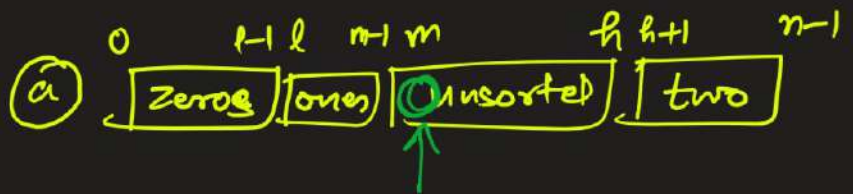
← zeros → ↑ ↑ ← ones → ↑ ↑ ← two's →

 l m m h

2, 1, 0, 0, 1, 1, 0, 2, 0, 1, 2, 1

l, m, h :-

- * 0 → l-1 all zeros
- * l → m-1 all ones
- * m → h unsorted array
- * h+1 → n-1 all twos



n=12

$l = 0, m = 0, h = a.size() - 1$

while($m \leq h$) {

if $a[m] == 0$:- swap($a[l]$, $a[m]$);
 $l++$, $m++$;

if $a[m] == 1$:- $m++$

if $a[m] == 2$:- swap($a[m]$, $a[h]$);
 $h--$;

}

$T = O(n)$
 $S = O(1)$

<https://leetcode.com/problems/sort-colors>

```
class Solution:
    def sortColors(self, nums: List[int]) -> None:
        """
        Do not return anything, modify nums in-place instead.
        """
        n = len(nums)
        l = m = 0
        h = n-1
        while(m<=h):
            if nums[m] == 0:
                nums[m], nums[l] = nums[l], nums[m]
                l += 1
                m += 1
            elif nums[m] == 1:
                m += 1
            elif nums[m] == 2:
                nums[m], nums[h] = nums[h], nums[m]
                h -= 1
```

==x==

==x==

Moore's Voting Algorithm

Problem:- Find the majority element in array
↳ ($> n/2$ times)

eg. 3, 3, 3, 2, 2, 1, 1 there is no majority element.

eg. 3, 3, 1, 3, 2, 1, 3, 3, 3 is a majority element.

① Brute :-

```
for (i : arr) {  
    c = 0
```

```
    for (j : arr) {  
        if (arr[j] == i) c++;
```

```
    }  
    if (c > n/2) return i;
```

```
}
```

$T = O(n^2)$

$S = O(1)$

② un_map <int, int> freq;

```

for (i : arr) {
    freq[i]++;
}
for (i : freq) {
    if (freq[i] > n/2) {
        return i;
    }
}

```

$T = O(n)$
 $S = O(n)$

③ optimal

~~3~~, ~~3~~, ~~1~~, ~~3~~, ~~7~~, ~~1~~, 3

If all other elements try to the cancel out the majority element, still the majority element will be left.

~~3~~, ~~3~~, ~~3~~, ~~3~~, ~~7~~, ~~1~~, 1

~~1~~, ~~3~~, ~~2~~, ~~3~~, 3, ~~3~~, ~~2~~, 3

$c = \cancel{1} \cancel{0} \cancel{1} \cancel{0} \cancel{1} \cancel{2} \cancel{1} \textcircled{2}$
 $\underline{m} = \cancel{1} \cancel{2} \textcircled{3}$

if a majority element exists,
 it has to be m

<https://leetcode.com/problems/majority-element/>

```
class Solution {
public:
    int majorityElement(vector<int>& nums) {
        int c=0, m;
        for(auto n: nums) {
            if(c==0) {
                m=n;
                c=1;
            }
            else if(n == m) c++;
            else c--;
        }
        c = 0;
        for(auto n: nums) if(n==m) c++;
        if(c>nums.size()/2) return m;
        return INT_MIN;
    }
};
```

Maximum Subarray Sum

[Kadane's Algorithm]

Problem:- Find the subarray with maximum sum.

eg: $[-2, 1, -3, 4, -1, 2, 1, -5, 4]$
sum = 6
(maximum)

(i) Brute force :-

check every subarray.

Find the sum

$maxSum = \max(ms, sum)$

```
msum = 0;
for (i : 0 → n-1) {
    for (j : i → n-1) {
        sum = 0;
        for (l : i → j) {
            sum += arr[l];
        }
        msum = max(msum, sum);
    }
}
return msum;
```

1, 1, 2, 1, 3, 2, 1

T =

$O(n^3)$

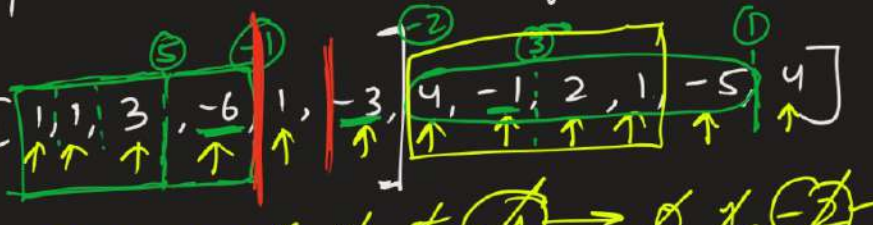
S = $O(1)$

<https://leetcode.com/problems/maximum-subarray/>

```
class Solution {
public:
    int maxSubArray(vector<int>& nums) {
        int ms = INT_MIN;
        for(int i=0; i<nums.size(); i++) {
            int s=0;
            for(int j=i; j<nums.size(); j++) {
                s+=nums[j];
                ms=max(ms, s);
            }
        }
        return ms;
    }
};
```

$T = O(n^2)$
 $S = O(1)$

— x —
 (iii) optimal (Kadane's Algorithm)

eg: 
 $\left\{ \begin{array}{l} \text{Sum Till Now} = 1, 2, 5, -1, 0, -3, 1, 0, 2, 3, -2, 1 \\ \text{max Sum} = 5 \end{array} \right.$

```

class Solution {
public:
    int maxSubArray(vector<int>& nums) {
        int s=0, ms=INT_MIN;
        for(auto n: nums) {
            s+=n;
            ms=max(ms, s);
            if(s<0) s=0; (*)
        }
        return ms;
    }
};

```

$$T = O(n)$$

$$S = O(1)$$

↓ ↓ ↓
-3, -2, -5

ms = ~~LAST-MIN~~ -3 -2

s = ~~-3~~ ~~0~~ ~~-2~~ ~~0~~ -5

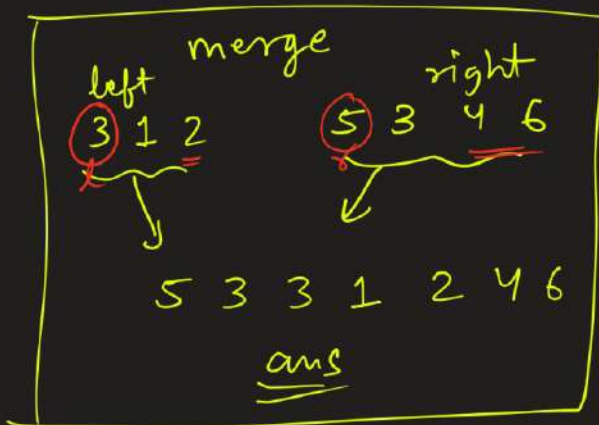
=x=

Sorting

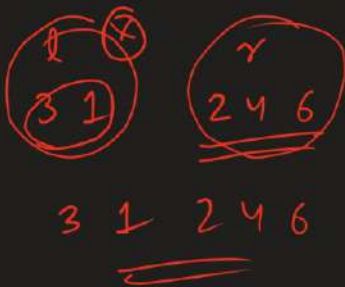
- Bubble
- Selection
- Insertion
- Merge ← $O(n \log n)$
- Quick ✓
- Heap ✓

vector<int> a;
{ sort(a.begin(), a.end())

{ a ← List.
a.sort()



- ✓ { both odd
insert bigger element
- ✓ { one even one odd
insert odd element
- ✓ { both even
insert smaller element
- = { insert all the remaining elements.




```

//{ Driver Code Starts
#include <bits/stdc++.h>
using namespace std;

// } Driver Code Ends
class Solution
{
public:
    vector<int> meo(vector<int> left, vector<int> right) {
        int l=0, r=0;
        vector<int> ans;
        while(l<left.size() && r<right.size()) {
            if(left[l]%2==0 && right[r]%2!=0)
                ans.push_back(right[r++]);
            else if(left[l]%2!=0 && right[r]%2==0)
                ans.push_back(left[l++]);
            else if(left[l]%2!=0) {
                if(left[l] > right[r])
                    ans.push_back(left[l++]);
                else
                    ans.push_back(right[r++]);
            }
            else {
                if(left[l] < right[r])
                    ans.push_back(left[l++]);
                else
                    ans.push_back(right[r++]);
            }
        }
        while(l<left.size()) ans.push_back(left[l++]);
        while(r<right.size()) ans.push_back(right[r++]);
        return ans;
    }
}

```

$$\left[\begin{array}{l} T = O(n \log n) \\ S = \underline{O(n)} \end{array} \right] ..$$

```

vector<int> mseo(vector<int> &nums, int l, int r) {
    if(l==r) return vector<int> {nums[l]};
    int m = (l+r)/2;
    vector<int> left = mseo(nums, l, m);
    vector<int> right = mseo(nums, m+1, r);
    return meo(left, right);
}

void sortIt(long long arr[], long long n) {
    vector<int> nums(arr, arr+n);
    vector<int> ans = mseo(nums, 0, nums.size()-1);
    int c=0;
    for(auto n: ans) arr[c++] = n;
}

//{ Driver Code Starts.
int main() {
    long long t;
    cin >> t;
    while (t--) {
        long long n;
        cin >> n;
        long long arr[n];

        for (int i = 0; i < n; i++)
            cin >> arr[i];

        Solution ob;
        ob.sortIt(arr, n);

        for (int i = 0; i < n; i++)
            cout << arr[i] << " ";
        cout << endl;
    }
    return 0;
}
// } Driver Code Ends

```