Problem on Arrays – 3

Assignment Solutions

PW SKILLS

Q1 - Given an integer array and two integers L and R. Find the sum of elements between the index L and index R.
Note: Both L and R inclusive. (Medium)

Input: [1,2,3,4,5] L=1 R=3
Output: 9
Explanation: 2+3+4=9

Input: [1,2,3,4,5] L=2 R=2
Output: 3
Explanation: 3

```
for(int i=1;i<n;i++)
  {
    arr[i]+=arr[i-1];
  }
  int ans=arr[R]-arr[L-1];
  return ans;
```

Explanation: The prefix sum will store the sum till Rth index and if we subtract the sum till L-1th index then we get the sum between L and R.

Q2 - There is a man going on a trek. The trek consists of n + 1 points at different altitudes. The man starts his trek on point 0 with altitude equal 0. You are given an integer array height of length n where height[i] is the net height in altitude between points i and i + 1 for all (0 <= i < n). Return the highest altitude of a point. (Easy)

Input: height=[-4,1,6,0,-8]
Output: 3
Explanation: The man starts at 0 and since then the altitudes covered will be [0,-4,-3,3,3,-5] so the greatest altitude will be 3

Input: height=[-5,-3,-2]
Output: 0
Explanation: The man starts at 0 and since then the altitudes will be[0,-5,-3,-2] so the greatest altitude will be 0.

```
for(int i=1;i<n;i++)
  {
    height[i]+=height[i-1];
  }
  int ans=0; //initially the man is at height=0
  for(int i=0;i<n;i++)
  {
    ans=max(ans,height[i]);
  }
return ans;
```

Explanation: We just need to find the prefix sum of all heights and then find the max height which is required as per the question.

Q3 - Given an integer array arr consisting of 0's and 1's only, return the max length of sequence which contains equal numbers of 0 and 1. (Medium)

Input: arr=[0,1,0,1]
Output: 4
Explanation: The longest sequence is 0,1,0,1

Input: arr=[0,1,1,0,1,1,1]
Output: 4
Explanation: The longest sequence is 0,1,1,0

```
int ans = 0;
int height= 0;
for (int i=0;i<gain.size();i++) {
  height = height + gain[i];
  if(ans>height)
  {
    ans=height;
  }
}
return ans;
```

Q4 - Given an integer array arr, return the number of consecutive sequences(subarrays) with odd sum. (Hard)

Input: [1,3,5]
Output: 4
Explanation: These sequences are [1],[3],[5] and [1,3,5]

Input: [0,2,4]
Output: 0

```
for(int i=1;i<n;i++)
{
  arr[i]+=arr[i-1];
}
int count=0;
for(int i=0;i<n;i++)
{
    for(int j=i+1;j<n;j++)
    {
      if(i==0 && arr[j]%2!=0)
      {
        count++;
      }
      else if((arr[j]-arr[i-1] && i!=0)%2!=0)
      {
        count++;
      }
    }
}
return count;
```

Explanation:Just find prefix sum and then check all the subarrays if the sum is odd

Q5 - Given an integer array arr, return an array ans such that ans[i] is equal to the product of all the elements of arr except arr[i]. (Hard)

Input: arr=[1,3,5,7]
Output: [105,35,21,15]
Explanation: ans=[3*5*7,1*5*7,1*3*7,1*3*5]

Input: [-5,-4,0,4,5]
Output: [0,0,400,0,0]

```
vector<int> ans(nums.size());
int pre[nums.size()];
pre[0]=nums[0];
int pos[nums.size()];
pos[nums.size()-1]=nums[nums.size()-1];
for(int i=1;i<nums.size();i++)
{
  pre[i]=nums[i]*pre[i-1];
}
cout<<pos[nums.size()-1];
for(int i=nums.size()-2;i>=0;i--)
{
  pos[i]=pos[i+1]*nums[i];
}
for(int i=0;i<nums.size()-1;i++)
{
 if(i==0)
 {
    ans[0]=pos[1];
 }else{
    ans[i]=pos[i+1]*pre[i-1];
 }
}
ans[nums.size()-1]=pre[nums.size()-2];
return ans;
```

Explanation: Here we need to find the prefix product of our array from the beginning as well as the end and store it in two different vectors/arrays. We are doing this because for each index i the answer will be product of begin[i-1]*end[i+1] i.e. the product from 0th to i-1th index and from i+1th index to end the array.

Q6 - Given an array of size 'n' (initially zero) and 'Q' updates (increase the value of all index from l to r with value x) and in the end print all the numbers of the array. (Medium)

Input: q = 3, x = 5, n = 5
[l,r] = {{0,3}, {4,4}, {1,2}
Output: [5 10 10 5 5]

```
void solve() {
int n, q, x;
cin >> n >> q >> x;

vector<int> a(n, 0);

while (q--) {
  int l, r;
  cin >> l >> r;
  if (r + 1 <= n - 1) a[r + 1] -= x;
  a[l] += x;
}
// taking prefix sums
for (int i = 1; i < n; i++) {
   a[i] += a[i - 1];
}
for (auto &i : a) cout << i << " ";
}
```