Prefix arrays and STL custom comparators

Prefix Array

Q.) You are given an array of n numbers, find the sum between index “l” to “r”.

Arr : {-1, 3, 0, 2, -1, 2, 1, -2}
idx : 0 1 2 3 4 5 6 7
l……..r

```
int sum=0;
for(int i=l;i<=r;i++) sum+=Arr[i];
cout<<"Sum between l to r is :" <<sum<<endl;
```

Time complexity -> O(n);

Q.) You have to solve the same problem but now there will be “q” number of queries.

Qi -> l,r sum between l to r.

q-> queries

n-> number of elements

```
for(int query=1;query<=q;query++){
    int l,r;
    cin>>l>>r;
    int sum=0;
    for(int i=l;i<=r;i++) sum+=Arr[i];
    cout<<sum<<endl;
}
```

Time Complexity -> **O(q*n)**

When q<=10^5 and n<=10^5 [**TLE]**

Prefix sum->

Arr : {-1, 3, 0, 2, -1, 2, 1, -2}

idx : 0 1 2 3 4 5 6 7

int Prefix[n];

Prefix[i]-> sum of all elements from start till index i.

Prefix[3] = 4

Prefix[6] = 6

Prefix[2] = 2

Prefix[7] = 4

Sum[l….r] -> Prefix[r]-Prefix[l-1];

Arr : {-1, 3, 0, 2, -1, 2, 1, -2}

idx : 0 1 2 3 4 5 6 7

l……..r

Prefix[r] -> {0,1,2,3…l-1,l,l+1…..r}

Prefix[l-1]-> {0,1,2,3…l-1}

Prefix[r]-Prefix[l-1] -> **{l,l+1….r}**

Prefix[r] -> sum[0….r];

Prefix[l] -> sum[0…..l];

Prefix[l-1] -> sum[0…..l-1];

Sum[l….r] = Prefix[r]-Prefix[l-1]

-> How to calculate prefix array?

```
int Prefix[n],suffix[n];
    Prefix[0]=arr[0];
    suffix[n-1]=arr[n-1];
    for(int i=1;i<n;i++)
Prefix[i]=Prefix[i-1]+Arr[i];
    for(int i=n-2;i>=0;i--)
suffix[i] = suffix[i+1]+arr[i];
```

Prefix[i-1]-> sum from [0…..i-1];
Prefix[i] -> sum from [0……i]?
= sum[0……i-1]+Arr[i];
= Prefix[i-1]+Arr[i];

Time Complexity-> O(n);

Solution->

Prefix Array has already been calculated. O(n);

```
for(int query=1;query<=q;query++){  // O(q);
        int l,r;
        cin>>l>>r;
        int sum = 0;
       if(l!=0)
          sum=Prefix[r]-Prefix[l-1]; // O(1);
        else sum=Prefix[r];  // O(1);
        cout<<sum<<endl;
}
```

Time Complexity-> O(max(q,n))=O(n)

Q.) You are given an array of n numbers and q number of queries, in each query you will be given 3 integers “l”,”r”,”x” and do an operation of adding the number x to each element between index “l” to “r”.
After all the q operations what would be the final array?

A[10] = {0,0,0,0,0,0,0,0,0,0}
Q queries -> l r x
l=3,r=7,x=5
1<=n<=10^5 , 1<=q<=10^5
l=2 r= 4 ,x=3
A[10] = {0,0,0,5,0,0,0,0,-5,0}
= {0,0,3,5,0,-3,0,0,-5,0}
= {0,0,3,8,8,5,5,5,0,0}

```
int n;
cin>>n;
int a[n+1]={0};
```

```
int q;
cin>>q;
while(q--){   //->for(int i=1;i<=q;i++)
    int l,r,x;
    cin>>l>>r>>x;
    a[l]+=x;    // l=0,r=n-1;
    a[r+1]-=x;
}
for(int i=1;i<n;i++){
a[i]=a[i]+a[i-1];
}
```

https://codeforces.com/contest/1473/problem/D

Comparator function in sort:-

```
bool compare(int p1,int p2)
{
    if(p1>=p2){
        return 1;
    }
    return 0;
}
```

```
int main(){
int n;
cin>>n;
int a[n];
for(int i=0;i<n;i++){
    cin>>a[i];
}
sort(a,a+n,compare);
}
```

Home-Work Problem

Q.) When we sort a vector of pairs then by default it sort according to the first element of the pair but using comparator function could you sort according to the second element?

(1,2),(2,5),(4,3),(2,2)

By Default(By first element of pair) ->
(1,2) (2,2) (2,5) (4,3) // **According to first element**
Sorting by second (By second element of pair) ->
(1,2) (2,2) (4,3) (2,5) // **According to second element**

Problem -> **Define your own comparator function for doing this process. (Home Work)?**

Note regarding last discussed problem-

```
bool com(pair<int,int> p1,pair<int,int> p2)
{
    if(2*p1.first+p1.second>=2*p2.first+p2.second){
        return 1;
    }
    return 0;
}
```

pair<int,int> p[n]; //{x,y}

pair<int,pair<int,int> > p;
p.first,p.second.first,p.second.second

vector<pair<int,int> > v

int x;
string s=to_string(x);
int y=stoi(s);