SegmentTrees.cpp

```cpp
//querys and build takes O(log n)
//example with segment sum
#include<bits/stdc++.h>

using namespace std;

int *p;

struct SegmentTree{
    SegmentTree *L, *R;
    int sum = 0;
    int l, r;

    int query(int a, int b){
        if(a == l && b == r) return sum;
        if(b <= L->r) return L->query(a,b);
        if(a >= R->l) return R->query(a,b);
        return (L->query(a,L->r) + R->query(R->l, b));
    }

    void update(int a, int val){
        if(l == r){
            sum += val;
            return;
        }
        int mid = (l + r)/2;
        if(l <= a && a<= mid)
            L->update(a, val);
        else
            R->update(a, val);
        sum = L->sum + R->sum;
    }

    void updateRange(int a, int b, int val){
        if(b < l or a > r)
            return;
        if(l == r){
            sum += val;
            return;
        }
        L->updateRange(a, b, val);
        R->updateRange(a,b,val);
        sum = L->sum + R->sum;
    }

    SegmentTree(int a, int b): l(a), r(b){
        if(a == b){
            sum = p[a];
            //L = R = null;
        }
        else{
            L = new SegmentTree ( a, (a+b)/2 );
            R = new SegmentTree ( (a+b)/2 + 1, b );
            sum = L->sum + R->sum;
        }
    }
};



int main(){
    int T;
    cin >> T;

    int l[T];
    for(int i = 0; i < T; i++)
        cin >> l[i];
    p = l;
    SegmentTree *stree = new SegmentTree(0, T);
    int N;
    cin >> N;
    int I;
    int J;
    while(N > 0){
        N -= 1;
        cin >> I;
        cin >> J;
        int aux = stree->query(I-1,J-1);
        cout << aux << "\n";
    }
}
```