

```
#include<bits/stdc++.h>
using namespace std;
const int MAX_N = (1 << 21) - 1;
const int inf = numeric_limits<int>::max();

int tree[MAX_N];

// call init(0,0,n) to init
// node k => [l,r)
void init(int k,int l,int r){
    if(r - l == 1){
        tree[k] = 0; // init leaf
    }else{
        int chl = k*2+1;
        int chr = k*2+2;
        tree[k] = 0; // init internal-node
        init(chl, l, (l+r)/2);
        init(chr, (l+r)/2, r);
    }
}

// call add(i,x,0,0,n) to add x to i
// node k => [l,r)
void add(int i,int x,int k,int l, int r){
    if(l <= i && i < r){
        if(r - l == 1){
            tree[k] += x;
        }else{
            int chl = k*2+1;
            int chr = k*2+2;
            add(i,x,chl,l, (l+r)/2);
            add(i,x,chr, (l+r)/2,r);
            tree[k] = tree[chl] + tree[chr];
        }
    }
}

// call sum(a,b,0,0,n) to calc sum [a,b)
// node k => [l,r)
int sum(int a,int b,int k,int l,int r){
    // not cross
    if(r <= a || b <= l) return 0;
    // [a,b) contain [l,r)
    else if(a <= l && r <= b) return tree[k];
    // otherwise
    else{
        int chl = k*2+1;
        int chr = k*2+2;
        int vl = sum(a,b,chl,l, (l+r)/2);
        int vr = sum(a,b,chr, (l+r)/2,r);
        return vl + vr;
    }
}

int main(void){
    int n,q;
    cin >> n >> q;
    init(0,0,n);
    for(int i = 0;i < q;++i){
        int c,x,y;
        cin >> c >> x >> y;
        if(c == 0){
            add(x-1,y,0,0,n);
        }else{
            cout << sum(x-1,y,0,0,n) << endl;
        }
    }
    return 0;
}
```