```cpp
#include<bits/stdc++.h>
using namespace std;
#define MAXN 100005
int n,a[MAXN];
struct Node{
    Node *left, *right;
    int val;
    int n;

    Node(int v=0, Node* _l=NULL, Node* _r=NULL) :
        val(v), left(_l), right(_r) {} /// Constructor

    void build(int b, int e){
        if(b==e){
            this -> val = a[b];
            return;
        }
        left  = new Node();
        right = new Node();
        int m = (b+e)/2;
        left  -> build(b, m);
        right -> build(m+1, e);
        this -> val = left -> val + right -> val;
    }

    Node *update(int b,int e,int x,int v){
        if(x<b || x>e)return this;
        if(b==e){
            Node *ret = new Node(val, left, right);
            ret -> val += v;
            return ret;
        }
        int m = (b+e)/2;
        Node *ret = new Node(val);
        ret -> left  = left -> update(b, m, x, v);
        ret -> right = right -> update(m+1, e, x, v);
        ret -> val = (ret -> left -> val) + (ret -> right -> val);
        return ret;
    }

    int query(int b,int e,int x,int y){
        if(b>y || e<x) return 0;
        if(b>=x && e<=y) return this -> val;
        int m = (b+e)/2;
        return  left -> query(b, m, x, y) + right -> query(m+1, e, x, y);
    }
};
Node *tree[MAXN];

int main(){
    scanf("%d",&n);
    for(int i=1; i<=n; i++)scanf("%d",&a[i]);

    tree[0] = new Node();
    tree[0] -> build(1,n);

    int q; scanf("%d",&q);
    int kth = 0;
    while(q--){
        int choice; scanf("%d",&choice);
        if(choice==1){
            int idx,x,v;
            scanf("%d%d%d",&idx,&x,&v);
            tree[++kth] = tree[idx] -> update(1, n, x, v);
        }else{
            int idx,x,y;
            scanf("%d%d%d",&idx,&x,&y);
            int ans = tree[idx] -> query(1, n, x, y);
            printf("%d\n",ans);
        }
    }
}
```