资源限制

内存限制：$256.0MB$

$C/C++$ 时间限制：$1.0s$

$Java$ 时间限制：$3.0s$

$Python$ 时间限制：$5.0s$

问题描述

南将军手下有 N 个士兵，分别编号 1 到 N，这些士兵的杀敌数都是已知的。

小工是南将军手下的军师，南将军经常想知道第 m 号到第 n 号士兵的总杀敌数，请你帮助小工来回答南将军吧。

南将军的某次询问之后士兵 i 可能又杀敌 q 人，之后南将军再询问的时候，需要考虑到新增的杀敌数。

输入格式

多组测试数据，以EOF结尾；

每组第一行是两个整数 N, M 其中 N 表示士兵的个数 $(1 < N < 1000000)$, M 表示指令的条数。$(1 < M < 100000)$

随后的一行是 N 个整数，a_i 表示第 i 号士兵杀敌数目。$(0 <= ai <= 100)$

随后的M行每行是一条指令，这条指令包含了一个字符串和两个整数，首先是一个字符串，如果是字符串 QUERY 则表示南将军进行了查询操作，后面的两个整数 m, n 表示查询的起始与终止士兵编号；如果是字符串 ADD 则后面跟的两个整数 I , $A(1 <= I <= N, 1 <= A <= 100)$ 表示第 I 个士兵新增杀敌数为 A .

输出格式

对于每次查询，输出一个整数 R 表示第 m 号士兵到第 n 号士兵的总杀敌数，每组输出占一行

样例输入

```
5 6
1 2 3 4 5
QUERY 1 3
ADD 1 2
QUERY 1 3
ADD 2 3
QUERY 1 2
QUERY 1 5
```

样例输出

```
6
8
8
20
```

解题思路

线段树模板，区间查询，单点修改，无需懒标记。

代码如下：

```
import java.util.*;
import java.io.*;

public class Main{
    static final int N = (int)1e6+10;
    static class Node{
        int l, r, sum;
    }
    static int n, m;
    static int[] a = new int[N];
    static Node[] tr = new Node[N*4];

    static void build(int u, int l, int r) {
        tr[u] = new Node();
        tr[u].l = l;
        tr[u].r = r;
        if (l == r) {
            tr[u].sum = a[l];
            return;
        }
        int mid = l + r >> 1;
        build(u<<1, l, mid);
        build(u<<1|1, mid+1, r);
        pushUp(u);
    }

    static void pushUp(int u) {
        tr[u].sum = tr[u<<1].sum + tr[u<<1|1].sum;
    }

    static int query(int u, int l, int r) {
        if (l <= tr[u].l && tr[u].r <= r) {
            return tr[u].sum;
        }
        int mid = tr[u].l + tr[u].r >> 1;
        int res = 0;
        if (l <= mid) {
            res += query(u << 1, l, r);
        }
        if (r > mid) {
```

```
            res += query(u << 1 | 1, l, r);
        }
        return res;
    }

    static void update(int u, int x, int val) {
        if (tr[u].l == x && tr[u].r == x) {
            tr[u].sum += val;
            return ;
        }
        int mid = tr[u].l + tr[u].r >> 1;
        if (x <= mid) update(u<<1, x, val);
        else update(u<<1|1, x, val);
        pushUp(u);
    }

    public static void main(String[] args) throws Exception {
        BufferedReader bf = new BufferedReader(new InputStreamReader(System.in));
        PrintWriter pw = new PrintWriter(System.out);
        String[] s;
        String str = null;
        while ((str = bf.readLine()) != null) {
            s = str.split(" ");
            n = Integer.valueOf(s[0]);
            m = Integer.valueOf(s[1]);
            s = bf.readLine().split(" ");
            for (int i = 1; i <= n; i ++ ) {
                a[i] = Integer.valueOf(s[i-1]);
            }
            build(1, 1, n);
            while (m -- > 0) {
                s = bf.readLine().split(" ");
                String ch = s[0];
                int l = Integer.valueOf(s[1]);
                int r = Integer.valueOf(s[2]);
                if (ch.equals("QUERY")) {
                    pw.println(query(1, l, r));
                } else if (ch.equals("ADD")) {
                    update(1, l, r);
                }
            }
        }
        pw.close();
    }
}
```