

```

1  #include<stdio.h>
2  #include<string.h>
3
4  #define MAX 112345
5  #define left(p) (p) << 1
6  #define right(p) ((p) << 1) + 1
7
8  typedef struct { int hom, ele, rat; }jogo;
9
10 int n, lazy[4 * MAX];
11 jogo st[4 * MAX];
12
13 void build(int p, int l, int r) {
14     int meio = (l + r) / 2;
15     if (l == r) { st[p].hom = 1; st[p].ele = st[p].rat = 0; return; }
16     build(left(p), l, meio);
17     build(right(p), meio + 1, r);
18     st[p].hom = st[left(p)].hom + st[right(p)].hom;
19     st[p].ele = st[left(p)].ele + st[right(p)].ele;
20     st[p].rat = st[left(p)].rat + st[right(p)].rat;
21 }
22
23 void range_update(int p, int l, int r, int i, int j) {
24     int aux, meio = (l + r) / 2, md;
25     if (lazy[p]) {
26         md = lazy[p] % 3;
27         if (md == 1) {
28             aux = st[p].ele; st[p].ele = st[p].hom;
29             st[p].hom = st[p].rat; st[p].rat = aux;
30         } else if (md == 2) {
31             aux = st[p].rat; st[p].rat = st[p].hom;
32             st[p].hom = st[p].ele; st[p].ele = aux;
33         }
34         if (l != r) {
35             lazy[left(p)] += lazy[p];
36             lazy[right(p)] += lazy[p];
37         }
38         lazy[p] = 0;
39     }
40     if (i > r || j < l) return;
41     if (i <= l && j >= r) {
42         aux = st[p].ele; st[p].ele = st[p].hom;
43         st[p].hom = st[p].rat; st[p].rat = aux;
44         if (l != r) {
45             lazy[left(p)] += 1;
46             lazy[right(p)] += 1;
47         }
48         return;
49     }
50     range_update(left(p), l, meio, i, j);
51     range_update(right(p), meio + 1, r, i, j);
52     st[p].hom = st[left(p)].hom + st[right(p)].hom;
53     st[p].ele = st[left(p)].ele + st[right(p)].ele;
54     st[p].rat = st[left(p)].rat + st[right(p)].rat;
55 }
56
57 jogo rmq(int p, int l, int r, int i, int j) {
58     int meio = (l + r) / 2, aux, md;
59     jogo p1, p2, ret;
60     if (i > r || j < l) { ret.hom = -1; return ret; }
61     if (lazy[p]) {
62         md = lazy[p] % 3;
63         if (md == 1) {
64             aux = st[p].ele; st[p].ele = st[p].hom;
65             st[p].hom = st[p].rat; st[p].rat = aux;
66         } else if (md == 2) {
67             aux = st[p].rat; st[p].rat = st[p].hom;
68             st[p].hom = st[p].ele; st[p].ele = aux;
69         }
70         if (l != r) {
71             lazy[left(p)] += lazy[p];
72             lazy[right(p)] += lazy[p];
73         }
74         lazy[p] = 0;

```

```
75     }
76     if (l >= i && r <= j) return st[p];
77     p1 = rmq(left(p), l, meio, i, j);
78     p2 = rmq(right(p), meio + 1, r, i, j);
79     if (p1.hom == -1) return p2;
80     if (p2.hom == -1) return p1;
81     ret.hom = p1.hom + p2.hom;
82     ret.ele = p1.ele + p2.ele;
83     ret.rat = p1.rat + p2.rat;
84     return ret;
85 }
86
87 int main(void) {
88     int m, a, b;
89     char c;
90     jogo resp;
91     while (scanf("%d %d", &n, &m) != EOF) {
92         memset(lazy, 0, sizeof(lazy)); build(1, 0, n - 1);
93         while (m--) {
94             scanf(" %c %d %d", &c, &a, &b); a--; b--;
95             if (c == 'M') range_update(1, 0, n - 1, a, b);
96             else {
97                 resp = rmq(1, 0, n - 1, a, b);
98                 printf("%d %d %d\n", resp.hom, resp.ele, resp.rat);
99             }
100         }
101     }
102     return 0;
103 }
```