

```

1 #include<bits/stdc++.h>
2 using namespace std;
3 #define MAXN 1024005
4 string s;
5 struct data{
6     int one, lazy;
7 }tree[4*MAXN];
8 void relaxation(int nd,int b,int e){
9     if(tree[nd].lazy==2){
10         tree[nd].one = (e-b+1) - tree[nd].one;
11     }else{
12         tree[nd].one = (e-b+1) * tree[nd].lazy;
13     }
14 }
15 void pushDown(int nd, int b,int e){
16     if(tree[nd].lazy!=-1) {
17         relaxation(nd,b,e);
18         if(b!=e){
19             int l=2*nd, r=2*nd+1, m=(b+e)/2;
20             if(tree[nd].lazy==2) {
21                 if(tree[l].lazy == 0) tree[l].lazy = 1;
22                 else if(tree[l].lazy == 1) tree[l].lazy = 0;
23                 else if(tree[l].lazy == 2) tree[l].lazy = -1;
24                 else if(tree[l].lazy ==-1) tree[l].lazy = 2;
25
26                 if(tree[r].lazy == 0) tree[r].lazy = 1;
27                 else if(tree[r].lazy == 1) tree[r].lazy = 0;
28                 else if(tree[r].lazy == 2) tree[r].lazy = -1;
29                 else if(tree[r].lazy ==-1) tree[r].lazy = 2;
30             }else {
31                 tree[l].lazy = tree[nd].lazy;
32                 tree[r].lazy = tree[nd].lazy;
33             }
34         }
35         tree[nd].lazy = -1;
36     }
37 }
38 void build(int nd, int b, int e){
39     if(b==e){
40         tree[nd].one = s[b]-'0';
41         tree[nd].lazy = -1;
42         return;
43     }
44     int l=2*nd, r=2*nd+1, m = (b+e)/2;
45     build(l,b,m);
46     build(r,m+1,e);
47     tree[nd].one = tree[l].one + tree[r].one;
48     tree[nd].lazy = -1;
49 }
50 void update(int nd,int b,int e,int x,int y,int c){
51     pushDown(nd,b,e);
52     int l=2*nd, r=2*nd+1, m = (b+e)/2;
53     if(b>y || e<x) return;
54     if(b>=x && e<=y){
55         if(c==2) tree[nd].one = (e-b+1) - tree[nd].one;
56         else tree[nd].one = (e-b+1) * c;
57
58         if(c==2) {
59             if(tree[l].lazy == 0) tree[l].lazy = 1;
60             else if(tree[l].lazy == 1) tree[l].lazy = 0;
61             else if(tree[l].lazy == 2) tree[l].lazy = -1;
62             else if(tree[l].lazy ==-1) tree[l].lazy = 2;
63
64             if(tree[r].lazy == 0) tree[r].lazy = 1;
65             else if(tree[r].lazy == 1) tree[r].lazy = 0;
66             else if(tree[r].lazy == 2) tree[r].lazy = -1;
67             else if(tree[r].lazy ==-1) tree[r].lazy = 2;
68         }else {
69             tree[l].lazy = c;
70             tree[r].lazy = c;
71         }
72         tree[nd].lazy = -1;
73         return;
74     }
75
76     update(l,b,m,x,y,c);
77     update(r,m+1,e,x,y,c);
78     tree[nd].one = tree[l].one + tree[r].one;
79 }
```

```
80 int query(int nd,int b,int e,int x,int y){
81     pushDown(nd,b,e);
82     if(b>y || e<x) return 0;
83     if(b>=x && e<=y) return tree[nd].one;
84     int l=2*nd, r=2*nd+1, m = (b+e)/2;
85     return query(l,b,m,x,y) + query(r,m+1,e,x,y);
86 }
87 int main(){
88     ios::sync_with_stdio(false); cin.tie(NULL); cout.tie(NULL);
89     int tt; cin>>tt;
90     for(int ks=1; ks<=tt; ks++){
91         cout<<"Case "<<ks<<":"<<endl;
92         s="";
93         int m; cin>>m;
94         while(m--){
95             int t; cin>>t;
96             string h; cin>>h;
97             while(t--) s += h;
98         }
99         int n = s.size();
100        build(1,0,n-1);
101
102        int q,k=0; cin>>q;
103        while(q--){
104            char c; int x,y;
105            cin>>c>>x>>y;
106            if(c=='F'){
107                update(1,0,n-1,x,y,1);
108            }else if(c=='E'){
109                update(1,0,n-1,x,y,0);
110            }else if(c=='I'){
111                update(1,0,n-1,x,y,2);
112            }else{
113                int ans = query(1,0,n-1,x,y);
114                cout<<"Q"<<++k<<": "<<ans<<endl;
115            }
116        }
117    }
118 }
119 return 0;
120 */
121 2
122 2
123 5
124 10
125 2
126 1000
127 5
128 F 0 17
129 I 0 5
130 S 1 10
131 E 4 9
132 S 2 10
133 3
134 3
135 1
136 4
137 0
138 2
139 0
140 2
141 I 0 2
142 S 0 8
143 */
144
```