

Cut the Rope

Limits: 5s, 512 MB

Bokkor is a rope seller. He has a special tool for cutting rope. But recently this tool is behaving weirdly. When he tries to cut the rope, it puts the cut in a random position of the rope from the starting position. He doesn't have enough money to repair the machine. As his friend, you need to find the k^{th} smallest rope segment after cutting the rope few times with the machine.

Input

There will be several test cases. First line of the input contains the number $T < 40$ (Number of test cases).

Every test case start with a value N (Length of the rope) and Q (number of query). Next Q lines contain a character ('C' or 'F') and an integer 'X'. If the query starts with 'C', then the machine cut the rope at position X . If it starts with 'F' then find the Length of the X^{th} rope segment, after sorting the ropes by length. Queries will always be valid.

Constraints:

$$1 < N < 10^{15}$$

$$1 \leq Q \leq 10^5$$

$$1 \leq X \leq 10^{15}$$

Output

For every test case, the first line contains "**Case X:**". X is the case number.

For every query starts with **F**. you need to print the length of the desired rope in a single line.

Samples

Input	Output
1	
10 5	
C 4	
F 1	
F 2	
C 9	
F 1	

```

1  /// https://toph.co/p/cut-the-rope
2  #include<bits/stdc++.h>
3  #include<ext/pb_ds/assoc_container.hpp>
4  #include<ext/pb_ds/tree_policy.hpp>
5  using namespace std;
6  using namespace __gnu_pbds;
7  using namespace __gnu_cxx;
8
9  #define ll long long
10 #define pll pair<ll,ll>
11
12 typedef tree<
13     pll,
14     null_type,
15     less<pll>,
16     rb_tree_tag,
17     tree_order_statistics_node_update>
18     ordered_set;
19
20 ordered_set os;
21 set<pll>ss;
22
23 int main(){
24     int tt; scanf("%d",&tt);
25     for(int ks=1; ks<=tt; ks++){
26         printf("Case %d:\n",ks);
27         ll n,q; scanf("%lld %lld",&n,&q);
28
29         os.clear();
30         ss.clear();
31
32         int id = 0;
33         os.insert({0,0});
34         os.insert({n,++id});
35
36         ss.insert({n,1});
37
38         while(q--){
39             char ch[2]; ll x;
40             scanf("%s %lld",&ch,&x);
41             if(ch[0]=='C'){
42                 auto it = ss.upper_bound({x,-1});
43                 pll cur = (*it);
44                 ll st = cur.second, ed = cur.first;
45                 ll len = ed-st+1;
46
47                 int cnt = os.order_of_key({len,-1});
48                 auto jt = os.find_by_order(cnt);
49
50                 os.erase(*jt);
51                 os.insert({x-st+1, ++id});
52                 os.insert({ed-x, ++id});
53
54                 ss.erase(it);
55                 ss.insert({x,st});
56                 ss.insert({ed,x+1});
57
58             }else{
59                 ll ans = (*os.find_by_order(x)).first;
60                 printf("%lld\n",ans);
61             }
62         }
63     }
64     return 0;
65 }

```