Algorithm Design Laboratory with Applications

Prof. Stefano Leucci

Problem: *Catch the robber.*

Its 5:59AM in Algoland. A criminal has just robbed a bank and is now on the run. The police suspects that he will try to escape using highway 42. Highway 42 is n kilometers long, and it is connected to the rest of the road network every kilometer through a toll booth. Currently all toll booths are closed for the night (people in Algoland only travel during daytime), but they will soon start to open. The police owns an helicopter which can be tasked with monitoring all traffic between any two consecutive toll booths, i.e., a segment $[k, k+1]$ from kilometer k to kilometer $k+1$, for some $k \in \{0, \dots, n-1\}$, and can possibly be ordered to relocate to a different segment. The police knows that the amount t_k of traffic in the segment $[k, k+1]$ is proportional to $\eta(0,k) \cdot \eta(k+1,n)$, where $\eta(i,j)$ denotes the number of toll booths open in $[i,j]$. Consequently, it is convenient to place the helicopter in the segment $[k^*, k^*+1]$ where $k^* = \arg\max_k t_k = \arg\max_k \eta(0,k) \cdot \eta(k+1,n)$.

You are in contact with both the highway company and the police: the highway company will notify you whenever a toll booth opens for the day, while the police will periodically ask you for the best interval $[k^*, k^*+1]$ to monitor with their helicopter.

Your task is to design an algorithm that is able to quickly compute the value k^* whenever requested.

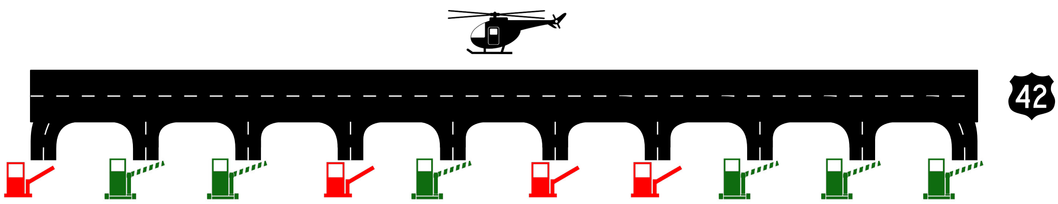
Input. The input consists of a set of instances, or *test-cases*, of the previous problem. The first line of the input contains the number T of test-cases. The first line of each test-case contains n and the number e of "events". Each of the following e lines describes an event. In particular, each of these lines has one of the following two formats:

Toll Booth Event: The line contains the character `T`, followed by a space, followed by a number $x \in \{0, \dots, n\}$. This means that the toll both at kilometer x is now open.

Police Event: The line contains only the character `P`. This means that you should report the integer $k^* \in \{0, \dots n\}$ that maximizes t_{k^*} to the police. In case of ties, the smallest value of k^* is preferred.

Output. The output consists of T lines, each corresponding to a test-case. The i-th of the lines contains m integers $k_1^*, \dots, k_m^*$, where m is the number of police events in the i-th test case and k_j^* is the answer to the j-th police event.

Example.



Km	0	1	2	3	4	5	6	7	8	9
t_k	0	5	8	8	9	9	9	8	5	0

Input (the last police event corresponds to the situation depicted above):

```
1
9 10
T 2
T 7
T 1
P
T 8
P
T 4
T 9
P
T 5
```

Output:

```
1 2 4
```

Assumptions. $1 \leq n < 2^{31}$; $1 \leq e \leq 2^{20}$. You can assume that the first police events happens after at least 2 toll booth events.

Requirements. Your algorithm must have an asymptotic time complexity of $O(e \log e)$.

Notes. A reasonable implementation should not require more than 1 second for each input file.