| I | **Secret Problemetters' Union** |
|---|---|
| | **Input:** Standard Input |
| | **Output:** Standard Output |

Solving problem is tough? Solving problem is easy? Rather confusing. But no doubt, problem solving is fun. Again, there's no doubt that problem setting is hard. Let alone the art of finding new ideas & all those statement, solution, data stuffs. The toughest part is to live with the fact that, if ever there's the slightest of errors, you are doomed. Hence, setting quality problems requires time. But once upon a time in a far far(!) away country, contest organizers forgot this fact completely. So they went on organizing one after another prestigious contests with very short notice to the problem setters. However, the contests had grand success & everyone forgot about what happened to the poor problem setters thereafter. Hence, the problem setters united & decided to have a useful plan for the future.

The problem setters decided that everyone of them will create problems throughout the year & maintain his own problem bank. Whenever someone creates a new problem, he adds the problem to his problem bank & the problem setting director gives it a rating. Rating is an integer between 1 and 100007. Whenever a problem setter is requested to arrange a contest, best **K** problems (**K** problems with maximum rating) are picked from his bank. The problems are used & removed from the bank. There can be at most 100 problems in a contest.

Once a year, before the tide of contest season begins, the problem setting director collects all the problems from the available banks & merges into a central problem bank. Any subsequent request for problem is served from the central problem bank.

You are to write a program that manages the problem bank.

## Input

Input file start with an integer **C (1<=C<=15),** The number of test cases. Every test case starts with an integer **P (1<=P<=50)** indicating the number of problems setters for that test case. Every problem setter has his own problem bank. The bank of problem setter **X** is denoted by the integer **X (1<=X<=P)**. At the beginning of a test case, all problem banks are empty. Another important thing to notice is that, at a time there will be at most **5000000** problems in the merged problem bank & at most **100000** problems in any other bank. Rest of the lines describe the operations on a problem bank using some instructions. There can be 4 kinds of instructions as described below:

An **I** instruction stands for an insert operation. The format of a single character **I** followed by 4 integers **B (1<=B<=P)**, **N(0<=N<=100000)**, **$A_0$ (0<=$A_0$<=1000000)** & **X(0<=X<=10000000)**, means insert **N** problems in the problem bank with id **B**. The ratings of these **N** problems can be calculated using **$A_0$** & **X**. The rating of a particular problem **$A_n$ = ($A_{n-1}$ + n * X) % 100007** where **n = 1 to N**.

An **U** instruction stands for a use & remove operation. The **U** instruction consists of a single **U** character followed by two integers **B (1<=B<=P)** & **K(0<=K<=number of problems in bank B)** meaning that pick & remove the best **K** problems from problem bank **B**.

A **M** instruction means a merge operation. There'll be at most one merge operation in a test case. A **M** instruction merges all the existing problem banks in a single problem bank. This merged bank is denoted as bank #1 thereafter.

A **Q** instruction indicates the end of a test case.

## Output

For every U instruction you receive, print 2 integers in a line showing the highest & lowest rating of the problems selected with that particular instruction. If you don't pick any problem print a couple of 0. Print a blank line between test cases.

### Sample Input

```
1
3
I 1 3 0 1
I 2 5 0 3
I 3 4 2 3
I 1 2 5 3
U 1 4
U 2 2
M
U 1 3
I 1 0 1 1
Q
```

### Output for Sample Input

```
14 3
45 30
32 18
```

Problemsetter: Mohammad Mahmudur Rahman
Special Thanks: Manzurur Rahman Khan