|  |  |
| --- | --- |
| .EWU logo() | Categorized  UVA Problems |

Problem Type : Data Structure

Set : DATA-01

Source : UHUNT (1st Edition)

|  |  |
| --- | --- |
|  | Xplosive  Email: [nasif.002@gmail.com](mailto:nasif.002@gmail.com) |

| Data Structures With Built-in Libraries (4/20 = 20%) | | | |
| --- | --- | --- | --- |
| Static array, vector, bitset, Direct Addressing Table (1/3) | | | |
| [482](http://uva.onlinejudge.org/external/4/482.html) - [Permutation Arrays](http://uva.onlinejudge.org/index.php?option=com_onlinejudge&Itemid=8&category=24&page=show_problem&problem=423) | ★ [http://uhunt.felix-halim.net/images/udebug3.png](http://www.udebug.com/UVa/482) | [discuss](http://acm.uva.es/board/search.php?keywords=482) | Lev 2 | --- ? --- |
| [594](http://uva.onlinejudge.org/external/5/594.html) - [One Little, Two Little, Three Little Endians](http://uva.onlinejudge.org/index.php?option=com_onlinejudge&Itemid=8&category=24&page=show_problem&problem=535) | ★ [http://uhunt.felix-halim.net/images/udebug3.png](http://www.udebug.com/UVa/594) | [discuss](http://acm.uva.es/board/search.php?keywords=594) | Lev 2 | --- ? --- |
| [11340](http://uva.onlinejudge.org/external/113/11340.html) - [Newspaper](http://uva.onlinejudge.org/index.php?option=com_onlinejudge&Itemid=8&category=24&page=show_problem&problem=2315) | ★ [http://uhunt.felix-halim.net/images/udebug3.png](http://www.udebug.com/UVa/11340) | [discuss](http://acm.uva.es/board/search.php?keywords=11340) | Lev 2 | ✔ 0.575s/2343 |
| STL algorithm (0/3) | | | |
| [146](http://uva.onlinejudge.org/external/1/146.html) - [ID Codes](http://uva.onlinejudge.org/index.php?option=com_onlinejudge&Itemid=8&category=24&page=show_problem&problem=82) | ★ [http://uhunt.felix-halim.net/images/udebug3.png](http://www.udebug.com/UVa/146) | [discuss](http://acm.uva.es/board/search.php?keywords=146) | Lev 1 | --- ? --- |
| [10194](http://uva.onlinejudge.org/external/101/10194.html) - [Football (aka Soccer)](http://uva.onlinejudge.org/index.php?option=com_onlinejudge&Itemid=8&category=24&page=show_problem&problem=1135) | ★ [http://uhunt.felix-halim.net/images/udebug3.png](http://www.udebug.com/UVa/10194) | [discuss](http://acm.uva.es/board/search.php?keywords=10194) | Lev 3 | --- ? --- |
| [10258](http://uva.onlinejudge.org/external/102/10258.html) - [Contest Scoreboard](http://uva.onlinejudge.org/index.php?option=com_onlinejudge&Itemid=8&category=24&page=show_problem&problem=1199) | ★ [http://uhunt.felix-halim.net/images/udebug3.png](http://www.udebug.com/UVa/10258) | [discuss](http://acm.uva.es/board/search.php?keywords=10258) | Lev 2 | --- ? --- |
| Sorting-related problems (2/3) | | | |
| [299](http://uva.onlinejudge.org/external/2/299.html) - [Train Swapping](http://uva.onlinejudge.org/index.php?option=com_onlinejudge&Itemid=8&category=24&page=show_problem&problem=235) | ★ [http://uhunt.felix-halim.net/images/udebug3.png](http://www.udebug.com/UVa/299) | [discuss](http://acm.uva.es/board/search.php?keywords=299) | Lev 1 | ✔ 0.012s/>10K |
| [11462](http://uva.onlinejudge.org/external/114/11462.html) - [Age Sort](http://uva.onlinejudge.org/index.php?option=com_onlinejudge&Itemid=8&category=24&page=show_problem&problem=2457) | ★ [http://uhunt.felix-halim.net/images/udebug3.png](http://www.udebug.com/UVa/11462) | [discuss](http://acm.uva.es/board/search.php?keywords=11462) | Lev 2 | ✔ 0.586s/2322 |
| [11495](http://uva.onlinejudge.org/external/114/11495.html) - [Bubbles and Buckets](http://uva.onlinejudge.org/index.php?option=com_onlinejudge&Itemid=8&category=24&page=show_problem&problem=2490) | ★ [http://uhunt.felix-halim.net/images/udebug3.png](http://www.udebug.com/UVa/11495) | [discuss](http://acm.uva.es/board/search.php?keywords=11495) | Lev 3 | --- ? --- |
| STL stack (1/3) | | | |
| [514](http://uva.onlinejudge.org/external/5/514.html) - [Rails](http://uva.onlinejudge.org/index.php?option=com_onlinejudge&Itemid=8&category=24&page=show_problem&problem=455) | ★ [http://uhunt.felix-halim.net/images/udebug3.png](http://www.udebug.com/UVa/514) | [discuss](http://acm.uva.es/board/search.php?keywords=514) | Lev 2 | --- ? --- |
| [673](http://uva.onlinejudge.org/external/6/673.html) - [Parentheses Balance](http://uva.onlinejudge.org/index.php?option=com_onlinejudge&Itemid=8&category=24&page=show_problem&problem=614) | ★ [http://uhunt.felix-halim.net/images/udebug3.png](http://www.udebug.com/UVa/673) | [discuss](http://acm.uva.es/board/search.php?keywords=673) | Lev 1 | ✔ 0.248s/6509 |
| [727](http://uva.onlinejudge.org/external/7/727.html) - [Equation](http://uva.onlinejudge.org/index.php?option=com_onlinejudge&Itemid=8&category=24&page=show_problem&problem=668) | ★ [http://uhunt.felix-halim.net/images/udebug3.png](http://www.udebug.com/UVa/727) | [discuss](http://acm.uva.es/board/search.php?keywords=727) | Lev 3 | --- ? --- |
| STL queue (0/3) | | | |
| [336](http://uva.onlinejudge.org/external/3/336.html) - [A Node Too Far](http://uva.onlinejudge.org/index.php?option=com_onlinejudge&Itemid=8&category=24&page=show_problem&problem=272) | ★ [http://uhunt.felix-halim.net/images/udebug3.png](http://www.udebug.com/UVa/336) | [discuss](http://acm.uva.es/board/search.php?keywords=336) | Lev 2 | Tried (3) |
| [10901](http://uva.onlinejudge.org/external/109/10901.html) - [Ferry Loading III](http://uva.onlinejudge.org/index.php?option=com_onlinejudge&Itemid=8&category=24&page=show_problem&problem=1842) | ★ [http://uhunt.felix-halim.net/images/udebug3.png](http://www.udebug.com/UVa/10901) | [discuss](http://acm.uva.es/board/search.php?keywords=10901) | Lev 4 | --- ? --- |
| [11034](http://uva.onlinejudge.org/external/110/11034.html) - [Ferry Loading IV](http://uva.onlinejudge.org/index.php?option=com_onlinejudge&Itemid=8&category=24&page=show_problem&problem=1975) | ★ [http://uhunt.felix-halim.net/images/udebug3.png](http://www.udebug.com/UVa/11034) | [discuss](http://acm.uva.es/board/search.php?keywords=11034) | Lev 4 | --- ? --- |
| STL map/set (0/3) | | | |
| [10226](http://uva.onlinejudge.org/external/102/10226.html) - [Hardwood Species](http://uva.onlinejudge.org/index.php?option=com_onlinejudge&Itemid=8&category=24&page=show_problem&problem=1167) | ★ [http://uhunt.felix-halim.net/images/udebug3.png](http://www.udebug.com/UVa/10226) | [discuss](http://acm.uva.es/board/search.php?keywords=10226) | Lev 3 | --- ? --- |
| [11308](http://uva.onlinejudge.org/external/113/11308.html) - [Bankrupt Baker](http://uva.onlinejudge.org/index.php?option=com_onlinejudge&Itemid=8&category=24&page=show_problem&problem=2283) | ★ [http://uhunt.felix-halim.net/images/udebug3.png](http://www.udebug.com/UVa/11308) | [discuss](http://acm.uva.es/board/search.php?keywords=11308) | Lev 4 | --- ? --- |
| [11136](http://uva.onlinejudge.org/external/111/11136.html) - [Hoax or what](http://uva.onlinejudge.org/index.php?option=com_onlinejudge&Itemid=8&category=24&page=show_problem&problem=2077) | ★ [http://uhunt.felix-halim.net/images/udebug3.png](http://www.udebug.com/UVa/11136) | [discuss](http://acm.uva.es/board/search.php?keywords=11136) | Lev 4 | --- ? --- |
| STL priority\_queue  (0/2) | | | |
| [908](http://uva.onlinejudge.org/external/9/908.html) - [Re-connecting Computer Sites](http://uva.onlinejudge.org/index.php?option=com_onlinejudge&Itemid=8&category=24&page=show_problem&problem=849) | ★ [http://uhunt.felix-halim.net/images/udebug3.png](http://www.udebug.com/UVa/908) | [discuss](http://acm.uva.es/board/search.php?keywords=908) | Lev 3 | --- ? --- |
| [11492](http://uva.onlinejudge.org/external/114/11492.html) - [Babel](http://uva.onlinejudge.org/index.php?option=com_onlinejudge&Itemid=8&category=24&page=show_problem&problem=2487) | ★ [http://uhunt.felix-halim.net/images/udebug3.png](http://www.udebug.com/UVa/11492) | [discuss](http://acm.uva.es/board/search.php?keywords=11492) | Lev 4 | --- ? --- |

| Data Structures With Our-Own Libraries (0/8 = 0%) | | | |
| --- | --- | --- | --- |
| Graph (simple ones) (0/2) | | | |
| [291](http://uva.onlinejudge.org/external/2/291.html) - [The House Of Santa Claus](http://uva.onlinejudge.org/index.php?option=com_onlinejudge&Itemid=8&category=24&page=show_problem&problem=227) | ★ [http://uhunt.felix-halim.net/images/udebug3.png](http://www.udebug.com/UVa/291) | [discuss](http://acm.uva.es/board/search.php?keywords=291) | Lev 2 | --- ? --- |
| [10928](http://uva.onlinejudge.org/external/109/10928.html) - [My Dear Neighbours](http://uva.onlinejudge.org/index.php?option=com_onlinejudge&Itemid=8&category=24&page=show_problem&problem=1869) | ★ [http://uhunt.felix-halim.net/images/udebug3.png](http://www.udebug.com/UVa/10928) | [discuss](http://acm.uva.es/board/search.php?keywords=10928) | Lev 3 | --- ? --- |
| Union-Find Disjoint Sets (0/3) | | | |
| [459](http://uva.onlinejudge.org/external/4/459.html) - [Graph Connectivity](http://uva.onlinejudge.org/index.php?option=com_onlinejudge&Itemid=8&category=24&page=show_problem&problem=400) | ★ [http://uhunt.felix-halim.net/images/udebug3.png](http://www.udebug.com/UVa/459) | [discuss](http://acm.uva.es/board/search.php?keywords=459) | Lev 2 | --- ? --- |
| [793](http://uva.onlinejudge.org/external/7/793.html) - [Network Connections](http://uva.onlinejudge.org/index.php?option=com_onlinejudge&Itemid=8&category=24&page=show_problem&problem=734) | ★ [http://uhunt.felix-halim.net/images/udebug3.png](http://www.udebug.com/UVa/793) | [discuss](http://acm.uva.es/board/search.php?keywords=793) | Lev 2 | --- ? --- |
| [11503](http://uva.onlinejudge.org/external/115/11503.html) - [Virtual Friends](http://uva.onlinejudge.org/index.php?option=com_onlinejudge&Itemid=8&category=24&page=show_problem&problem=2498) | ★ [http://uhunt.felix-halim.net/images/udebug3.png](http://www.udebug.com/UVa/11503) | [discuss](http://acm.uva.es/board/search.php?keywords=11503) | Lev 3 | --- ? --- |
| Segment Tree (0/3) | | | |
| [11235](http://uva.onlinejudge.org/external/112/11235.html) - [Frequent values](http://uva.onlinejudge.org/index.php?option=com_onlinejudge&Itemid=8&category=24&page=show_problem&problem=2176) | ★ [π](http://www.algorithmist.com/index.php/UVa_11235) |[http://uhunt.felix-halim.net/images/udebug3.png](http://www.udebug.com/UVa/11235) |[discuss](http://acm.uva.es/board/search.php?keywords=11235) | Lev 3 | --- ? --- |
| [11297](http://uva.onlinejudge.org/external/112/11297.html) - [Census](http://uva.onlinejudge.org/index.php?option=com_onlinejudge&Itemid=8&category=24&page=show_problem&problem=2272) | ★ [http://uhunt.felix-halim.net/images/udebug3.png](http://www.udebug.com/UVa/11297) | [discuss](http://acm.uva.es/board/search.php?keywords=11297) | Lev 4 | --- ? --- |
| [11402](http://uva.onlinejudge.org/external/114/11402.html) - [Ahoy, Pirates!](http://uva.onlinejudge.org/index.php?option=com_onlinejudge&Itemid=8&category=24&page=show_problem&problem=2397) | ★ [http://uhunt.felix-halim.net/images/udebug3.png](http://www.udebug.com/UVa/11402) | [discuss](http://acm.uva.es/board/search.php?keywords=11402) | Lev 4 | --- ? --- |

**482 - Permutation Arrays**

Time limit: 3.000 seconds

|  |
| --- |
| **Permutation Arrays** |

In many computer problems, it is necessary to permute data arrays. That is, the data in an array must be re-arranged in some specified order. One way to permute arbitrary data arrays is to specify the permutations with an index array to point out the position of the elements in the new array. Let *x* be an array that is to be permuted and let *x'* be the permuted array. Then, we have the relationship between *x* and *x'* that *x'*pi = *x*i.

**Input**

The input begins with a single positive integer on a line by itself indicating the number of the cases following, each of them as described below. This line is followed by a blank line, and there is also a blank line between two consecutive inputs.

Each input set will contain two lines of numbers. The first line will be an index array *p* containing the integers 1...*n*, where *n* is the number of integers in the list. The numbers in the first line will have been permuted in some fashion. The second line will contain a list numbers in floating point format.

**Output**

For each test case, the output must follow the description below. The outputs of two consecutive cases will be separated by a blank line.

The output for this program will be the list of floating point numbers from the input set, ordered according to the permutation array from the input file. The output numbers must be printed one per line in the same format in which they each appeared in the input file.

**Sample Input**

1

3 1 2

32.0 54.7 -2

**Sample Output**

54.7

-2

32.0

*Miguel Revilla 2001-05-21*

**594 - One Little, Two Little, Three Little Endians**

Time limit: 3.000 seconds

|  |
| --- |
| **One Little, Two Little, Three Little Endians** |

Writing programs that are completely portable across different operating systems, operating system versions and hardware platforms is a challenging task. One of the difficulties encountered is a result of decisions made by hardware manufacturers about how they will store integer data in memory. Because these representations can differ from machine to machine, sharing binary data often cannot be done without modifying the way in which the data is stored or the way in which it is handled by one or more of the platforms.

Fortunately there is near-universal agreement among hardware manufacturers that addressable memory be ordered into 8-bit bytes. For integer data values that require more than 8-bits, such as the typical 2- byte, 4-byte, and 8-byte integer types available on most modern hardware, there is no such agreement and two incompatible storage schemes exist. The first stores integers as groups of consecutive 8-bit bytes with the least significant byte occupying the lowest memory location within the group and the most significant byte occupying the highest memory location. The second is just the reverse; the least significant byte is stored in the highest memory location within the group, and the most significant byte is stored in the lowest memory location. The computing industry has dubbed these schemes Little Endian and Big Endian, respectively. There is also near-universal agreement that signed integers are stored using "two's complement" representation, and you may assume that this is the case.

When binary integer data is shared between a Little Endian and Big Endian machine, a data conversion must be performed which involves reversing the bytes of the data. Once the bytes have been reversed the integer is then correctly interpreted by the hardware as the original value from the opposite-endian machine. The object of this problem is to write a program that will read a list of integers and report the integers that the binary representations of the input integers would represent on an opposite-endian machine.

## Input

The input will consist of a list integers. The end of the input file marks the end of the list. All input integers can be represented as a 32-bit signed integer value. That is, the input integers will be in the range -2147483648 to 2147483647.

## Output

For each input integer a single line should be printed to the output file. The line should contain the input integer followed by the phrase ``converts to" followed by one space followed the other-endian value.

## Sample Input

123456789

-123456789

1

16777216

20034556

## Sample Output

123456789 converts to 365779719

-123456789 converts to -349002504

1 converts to 16777216

16777216 converts to 1

20034556 converts to -55365375

**10194 - Football (aka Soccer)**

Time limit: 3.000 seconds

|  |
| --- |
| **Problem A: Football (aka Soccer)** |

## The Problem

Football the most popular sport in the world (americans insist to call it "Soccer", but we will call it "Football"). As everyone knows, Brasil is the country that have most World Cup titles (four of them: 1958, 1962, 1970 and 1994). As our national tournament have many teams (and even regional tournaments have many teams also) it's a very hard task to keep track of standings with so many teams and games played!

So, your task is quite simple: write a program that receives the tournament name, team names and games played and outputs the tournament standings so far.

A team wins a game if it scores more goals than its oponent. Obviously, a team loses a game if it scores less goals. When both teams score the same number of goals, we call it a tie. A team earns 3 points for each win, 1 point for each tie and 0 point for each loss.

Teams are ranked according to these rules (in this order):

1. Most points earned.
2. Most wins.
3. Most goal difference (i.e. goals scored - goals against)
4. Most goals scored.
5. Less games played.
6. Lexicographic order.

## The Input

The first line of input will be an integer N in a line alone (0 < N < 1000). Then, will follow N tournament descriptions. Each one begins with the tournament name, on a single line. Tournament names can have any letter, digits, spaces etc. Tournament names will have length of at most 100. Then, in the next line, there will be a number T (1 < T <= 30), which stands for the number of teams participating on this tournament. Then will follow T lines, each one containing one team name. Team names may have any character that have ASCII code greater than or equal to 32 (space), except for '#' and '@' characters, which will never appear in team names. No team name will have more than 30 characters.

Following to team names, there will be a non-negative integer G on a single line which stands for the number of games already played on this tournament. G will be no greater than 1000. Then, G lines will follow with the results of games played. They will follow this format:

team\_name\_1#goals1@goals2#team\_name\_2

For instance, the following line:

Team A#3@1#Team B

Means that in a game between Team A and Team B, Team A scored 3 goals and Team B scored 1. All goals will be non-negative integers less than 20. You may assume that there will not be inexistent team names (i.e. all team names that appear on game results will have apperead on the team names section) and that no team will play against itself.

## The Output

For each tournament, you must output the tournament name in a single line. In the next T lines you must output the standings, according to the rules above. Notice that should the tie-breaker be the lexographic order, it must be done case insenstive. The output format for each line is shown bellow:

[a]) Team\_name [b]p, [c]g ([d]-[e]-[f]), [g]gd ([h]-[i])

Where:

* [a] = team rank
* [b] = total points earned
* [c] = games played
* [d] = wins
* [e] = ties
* [f] = losses
* [g] = goal difference
* [h] = goals scored
* [i] = goals against

There must be a single blank space between fields and a single blank line between output sets. See the sample output for examples.

|  |  |
| --- | --- |
| Sample Input 2  World Cup 1998 - Group A  4  Brazil  Norway  Morocco  Scotland  6  Brazil#2@1#Scotland  Norway#2@2#Morocco  Scotland#1@1#Norway  Brazil#3@0#Morocco  Morocco#3@0#Scotland  Brazil#1@2#Norway  Some strange tournament  5  Team A  Team B  Team C  Team D  Team E  5  Team A#1@1#Team B  Team A#2@2#Team C  Team A#0@0#Team D  Team E#2@1#Team C  Team E#1@2#Team D | Sample Output World Cup 1998 - Group A  1) Brazil 6p, 3g (2-0-1), 3gd (6-3)  2) Norway 5p, 3g (1-2-0), 1gd (5-4)  3) Morocco 4p, 3g (1-1-1), 0gd (5-5)  4) Scotland 1p, 3g (0-1-2), -4gd (2-6)  Some strange tournament  1) Team D 4p, 2g (1-1-0), 1gd (2-1)  2) Team E 3p, 2g (1-0-1), 0gd (3-3)  3) Team A 3p, 3g (0-3-0), 0gd (3-3)  4) Team B 1p, 1g (0-1-0), 0gd (1-1)  5) Team C 1p, 2g (0-1-1), -1gd (3-4) |

### 10258 - Contest Scoreboard

Time limit: 3.000 seconds

Think the contest score boards are wrong? Here's your chance to come up with the right rankings.

Contestants are ranked first by the number of problems solved (the more the better), then by decreasing amounts of penalty time. If two or more contestants are tied in both problems solved and penalty time, they are displayed in order of increasing team numbers.

A problem is considered solved by a contestant if any of the submissions for that problem was judged correct. Penalty time is computed as the number of minutes it took for the first correct submission for a problem to be received plus 20 minutes for each incorrect submission received prior to the correct solution. Unsolved problems incur no time penalties.

## Input:

**The input begins with a single positive integer on a line by itself indicating the number of the cases following, each of them as described below. This line is followed by a blank line, and there is also a blank line between two consecutive inputs.**

Input consists of a snapshot of the judging queue, containing entries from some or all of contestants 1 through 100 solving problems 1 through 9. Each line of input will consist of three numbers and a letter in the format

contestant problem time L

where L can be C, I, R, U or E. These stand for Correct, Incorrect, clarification Request, Unjudged and Erroneous submission. The last three cases do not affect scoring.

Lines of input are in the order in which submissions were received.

## Output:

**For each test case, the output must follow the description below. The outputs of two consecutive cases will be separated by a blank line.**

Output will consist of a scoreboard sorted as previously described. Each line of output will contain a contestant number, the number of problems solved by the contestant and the time penalty accumulated by the contestant. Since not all of contestants 1-100 are actually participating, display only the contestants that have made a submission.

## Sample Input:

1

1 2 10 I

3 1 11 C

1 2 19 R

1 2 21 C

1 1 25 C

## Sample Output:

1 2 66

3 1 11

### 11495 - Bubbles and Buckets

Time limit: 3.000 seconds

## The Problem

Andrea, Carlos and Marcelo are close friends and spend their weekends by the swimming pool. While Andrea gets a suntan, both friends play*Bubbles*. Andrea, a very smart computer scientist, has already told them that she does not understand why they spend so much time playing a game so simple.

Using her laptop, Carlos and Marcelo generate a random integer *N* and a sequence, also random, which is a permutation from *1, 2, ..., N*.

The game then begins. The players play by turns, and at each turn a player makes a move. Marcelo is always the first to play.

A move consists of choosing one pair of consecutive elements that are out of order in the sequence, and swapping both elements. For example, given the sequence *1, 5, 3, 4, 2*, a player may swap *3* and *5* or *4* and *2*, but cannot swap *3* and *4* nor *5* and *2*. Continuing with the example, if the player decides to swap *5* and *3*, the new sequence will be *1, 3, 5, 4, 2*.

Sooner or later, the sequence will be sorted. The player that cannot make a move loses. Andrea, with disdain, always says that it would be simpler to play Odd or Even, to the same effect. Your mission, in case you decide to accept it, is to determine who wins the game, given the initial permutation *P*.

## The Input

The input contains several test cases. Each test case is composed of a single line, in which all integers are separated by one space. Each line contains an integer *N* (*2 ≤ N ≤ 105*), followed by the initial sequence *P* = (*X1, X2, ...,XN*) of *N* distinct integers, with *1 ≤ Xi* ≤ N for *1 ≤ i ≤ N*.

The end of input is indicated by a line containing only one zero.

## The Output

For each test case in the input, your program must print a single line, containing the name of the winner, equal to Carlos or Marcelo.

## Sample Input

5 1 5 3 4 2

5 5 1 3 4 2

5 1 2 3 4 5

6 3 5 2 1 4 6

5 5 4 3 2 1

6 6 5 4 3 2 1

0

## Sample Output

Marcelo

Carlos

Carlos

Carlos

Carlos

Marcelo

### 514 - Rails

Time limit: 3.000 seconds

There is a famous railway station in PopPush City. Country there is incredibly hilly. The station was built in last century. Unfortunately, funds were extremely limited that time. It was possible to establish only a surface track. Moreover, it turned out that the station could be only a dead-end one (see picture) and due to lack of available space it could have only one track.

\begin{picture}(6774,3429)(0,-10)
\put(1789.500,1357.500){\arc{3645.278}{4.7247}...
...tFigFont{14}{16.8}{\rmdefault}{\mddefault}{\updefault}Station}}}}}
\end{picture}

The local tradition is that every train arriving from the direction A continues in the direction B with coaches reorganized in some way. Assume that the train arriving from the direction A has $N \leŸ 1000$coaches numbered in increasing order $1, 2, \dots, N$. The chief for train reorganizations must know whether it is possible to marshal coaches continuing in the direction B so that their order will be $a_1. a_2, \dots, a_N$. Help him and write a program that decides whether it is possible to get the required order of coaches. You can assume that single coaches can be disconnected from the train before they enter the station and that they can move themselves until they are on the track in the direction B. You can also suppose that at any time there can be located as many coaches as necessary in the station. But once a coach has entered the station it cannot return to the track in the direction A and also once it has left the station in the direction B it cannot return back to the station.

## Input

The input file consists of blocks of lines. Each block except the last describes one train and possibly more requirements for its reorganization. In the first line of the block there is the integer *N* described above. In each of the next lines of the block there is a permutation of $1, 2, \dots, N$ The last line of the block contains just 0. The last block consists of just one line containing 0.

## Output

The output file contains the lines corresponding to the lines with permutations in the input file. A line of the output file contains Yes if it is possible to marshal the coaches in the order required on the corresponding line of the input file. Otherwise it contains No. In addition, there is one empty line after the lines corresponding to one block of the input file. There is no line in the output file corresponding to the last ``null'' block of the input file.

|  |  |
| --- | --- |
| Sample Input 5  1 2 3 4 5  5 4 1 2 3  0  6  6 5 4 3 2 1  0  0 | Sample Output Yes  No  Yes |

### 727 - Equation

Time limit: 3.000 seconds

## Input and Output

Write a program that changes an infix expression to a postfix expression according to the following specifications.

1. The infix expression to be converted is in the input file in the format of one character per line, with a maximum of 50 lines in the file. For example, (3+2)\*5 would be in the form:

(

3

+

2

)

\*

5

1. The input starts with an integer on a line by itself indicating the number of test cases. Several infix expressions follows, preceded by a blank line.
2. The program will only be designed to handle the binary operators +,-, \*, /.
3. The operands will be one digit numerals.
4. The operators \* and / have the highest priority. The operators + and- have the lowest priority. Operators at the same precedence level associate from left to right. Parentheses act as grouping symbols that over-ride the operator priorities.
5. The output file will have each postfix expression all on one line. Print a blank line between different expressions.
6. Each testcase will be an expression with valid syntax.

## Sample Input

1

(

3

+

2

)

\*

5

## Sample Output

32+5\*

Miguel Revilla   
2000-08-31

### 10901 - Ferry Loading III

Time limit: 3.000 seconds

Before bridges were common, ferries were used to transport cars across rivers. River ferries, unlike their larger cousins, run on a guide line and are powered by the river's current. Cars drive onto the ferry from one end, the ferry crosses the river, and the cars exit from the other end of the ferry.

There is a ferry across the river that can take n cars across the river in tminutes and return in t minutes. A car may arrive at either river bank to be transported by the ferry to the opposite bank. The ferry travels continuously back and forth between the banks so long it is carrying a car or there is at least one car waiting at either bank. Whenever the ferry arrives at one of the banks, it unloads its cargo and loads up to n cars that are waiting to cross. If there are more than n, those that have been waiting the longest are loaded. If there are no cars waiting on either bank, the ferry waits until one arrives, loads it (if it arrives on the same bank of the ferry), and crosses the river. At what time does each car reach the other side of the river?

The first line of input contains *c*, the number of test cases. Each test case begins with *n, t, m*. *m* lines follow, each giving the arrival time for a car (in minutes since the beginning of the day), and the bank at which the car arrives ("left" or "right"). For each test case, output one line per car, in the same order as the input, giving the time at which that car is unloaded at the opposite bank. Output an empty line between cases.

You may assume that 0 < n, t, m ≤ 10000. The arrival times for each test case are strictly increasing. The ferry is initially on the left bank. Loading and unloading time may be considered to be 0.

|  |  |
| --- | --- |
| Sample input 2  2 10 10  0 left  10 left  20 left  30 left  40 left  50 left  60 left  70 left  80 left  90 left  2 10 3  10 right  25 left  40 left | Output for sample input 10  30  30  50  50  70  70  90  90  110  30  40  60 |

*Gordon V. Cormack*

### 11034 - Ferry Loading IV

Time limit: 3.000 seconds

## http://uva.onlinejudge.org/external/110/p11034.jpgProblem A: Ferry Loading IV

Before bridges were common, ferries were used to transport cars across rivers. River ferries, unlike their larger cousins, run on a guide line and are powered by the river's current. Cars drive onto the ferry from one end, the ferry crosses the river, and the cars exit from the other end of the ferry.

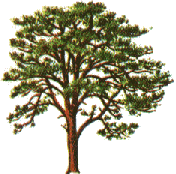
There is an *l*-meter-long ferry that crosses the river. A car may arrive at either river bank to be transported by the ferry to the opposite bank. The ferry travels continuously back and forth between the banks so long as it is carrying a car or there is at least one car waiting at either bank. Whenever the ferry arrives at one of the banks, it unloads its cargo and loads up cars that are waiting to cross as long as they fit on its deck. The cars are loaded in the order of their arrival; ferry's deck accommodates only one lane of cars. The ferry is initially on the left bank where it broke and it took quite some time to fix it. In the meantime, lines of cars formed on both banks that await to cross the river.

The first line of input contains *c*, the number of test cases. Each test case begins with *l, m*. *m* lines follow describing the cars that arrive in this order to be transported. Each line gives the length of a car (in centimeters), and the bank at which the car arrives ("left" or "right").

For each test case, output one line giving the number of times the ferry has to cross the river in order to serve all waiting cars.

|  |  |
| --- | --- |
| Sample input 4  20 4  380 left  720 left  1340 right  1040 left  15 4  380 left  720 left  1340 right  1040 left  15 4  380 left  720 left  1340 left  1040 left  15 4  380 right  720 right  1340 right  1040 right | Output for sample input 3  3  5  6 |

### 10226 - Hardwood Species

Time limit: 3.000 seconds

Hardwoods are the botanical group of trees that have broad leaves, produce a fruit or nut, and generally go dormant in the winter.

America's temperate climates produce forests with hundreds of hardwood species -- trees that share certain biological characteristics. Although oak, maple and cherry all are types of hardwood trees, for example, they are different species. Together, all the hardwood species represent 40 percent of the trees in the United States.

On the other hand, softwoods, or conifers, from the Latin word meaning "cone-bearing," have needles. Widely available US softwoods include cedar, fir, hemlock, pine, redwood, spruce and cypress. In a home, the softwoods are used primarily as structural lumber such as 2x4s and 2x6s, with some limited decorative applications.

Using satellite imaging technology, the Department of Natural Resources has compiled an inventory of every tree standing on a particular day. You are to compute the total fraction of the tree population represented by each species.

The first line is the number of test cases, followed by a blank line. Each test case of your program consists of a list of the species of every tree observed by the satellite; one tree per line. No species name exceeds 30 characters. There are no more than 10,000 species and no more than 1,000,000 trees. There is a blank line between each consecutive test case.

For each test case print the name of each species represented in the population, in alphabetical order, followed by the percentage of the population it represents, to 4 decimal places. Print a blank line between 2 consecutive data sets.

|  |  |
| --- | --- |
| Sample Input 1  Red Alder  Ash  Aspen  Basswood  Ash  Beech  Yellow Birch  Ash  Cherry  Cottonwood  Ash  Cypress  Red Elm  Gum  Hackberry  White Oak  Hickory  Pecan  Hard Maple  White Oak  Soft Maple  Red Oak  Red Oak  White Oak  Poplan  Sassafras  Sycamore  Black Walnut  Willow | Output for Sample Input Ash 13.7931  Aspen 3.4483  Basswood 3.4483  Beech 3.4483  Black Walnut 3.4483  Cherry 3.4483  Cottonwood 3.4483  Cypress 3.4483  Gum 3.4483  Hackberry 3.4483  Hard Maple 3.4483  Hickory 3.4483  Pecan 3.4483  Poplan 3.4483  Red Alder 3.4483  Red Elm 3.4483  Red Oak 6.8966  Sassafras 3.4483  Soft Maple 3.4483  Sycamore 3.4483  White Oak 10.3448  Willow 3.4483  Yellow Birch 3.4483 |

### 11308 - Bankrupt Baker

Time limit: 3.000 seconds

Wolfgang Puck has an extensive collection of cake recipes. They are separated into different binders depending on the type of cake. Although Wolfgang has restaurant franchises all over the world, he is in a period of hard times and is struggling to afford ingredients for his cakes. What cakes can he create with his small budget?

## Input

On the first line you are given t (1 ≤ t ≤ 100), the number of binders. Each binder begins with title, the name of the binder, then on the next linem n b (1 ≤ m, n ≤ 100, 1 ≤ b ≤ 106) where b is Wolfgang's budget in dollars. The next m lines are given as "ingredient c" (see sample input) where c (0 ≤ c ≤ 5000) is the price in dollars for one unit of ingredient.

Then follow n recipes. Each recipe begins with name on a line of its own, then on the very next line k (1 ≤ k ≤ 100). The following k lines are of the form "requirement x" (see sample input) where x is the number of units of the ingredient requirement used in the recipe name.

## Output

For each binder, output the name of the binder in uppercase letters then on separate lines a list of recipes within Wolfgang's budget in increasing order of cost. If no such recipe exists, print "Too expensive!". If recipes have the same cost print them in lexicographical order. Print a blank line after each binder.

|  |  |
| --- | --- |
| Sample Input 2  My Favourite Cheesecake  8 3 100  sugar 4  water 0  lemonjuice 3  creamcheese 20  vanilla 5  egg 5  cream 10  strawberry 5  Strawberry Whipped Cream  2  cream 5  strawberry 3  Scrumptious Caramel Topping  3  sugar 6  water 3  lemonjuice 1  Secret Cheesecake Base  5  creamcheese 3  sugar 5  vanilla 1  egg 6  cream 1  Million Dollar Cakes  3 1 999999  costlyflour 500  gold 4500  diamond 5000  Display Cake - Do Not Eat!  3  costlyflour 100  gold 100  diamond 100 | Output for the Sample Input MY FAVOURITE CHEESECAKE  Scrumptious Caramel Topping  Strawberry Whipped Cream  MILLION DOLLAR CAKES  Too expensive! |

### 11136 - Hoax or what

Time limit: 3.000 seconds

## Problem H: Hoax or what

Each Mal-Wart supermarket has prepared a promotion scheme run by the following rules:

* A client who wants to participate in the promotion (aka a sucker) must write down their phone number on the bill of their purchase and put the bill into a special urn.
* Two bills are selected from the urn at the end of each day: first the highest bill is selected and then the lowest bill is selected. The client who paid the largest bill receives a monetary prize equal to the difference between his bill and the lowest bill of the day.
* Both selected bills are not returned to the urn while all the remaining ones are kept in the urn for the next day.
* Mal-Wart has many clients such that at the end of each day there are at least two bills in the urn.
* It is quite obvious why Mal-Wart is doing this: they sell crappy products which break quickly and irreparably. They give a short-term warranty on their products but in order to obtain a warranty replacement you need the bill of sale. So if you are gullible enough to participate in the promotion you will regret it.

Your task is to write a program which takes information about the bills put into the urn and computes Mal-Wart's cost of the promotion.

The input contains a number of cases. The first line in each case contains an integer *n*, 1<=*n*<=5000, the number of days of the promotion. Each of the subsequent *n* lines contains a sequence of non-negative integers separated by whitespace. The numbers in the *(i+1)*-st line of a case give the data for the *i*-th day. The first number in each of these lines, *k*, 0≤*k*≤105, is the number of bills and the subsequent *k* numbers are positive integers of the bill amounts. No bill is bigger than 106. The total number of all bills is no bigger than 106. The case when *n* = 0 terminates the input and should not be processed.

For each case of input print one number: the total amount paid to the clients by Mal-Wart as the result of the promotion.

### Sample input

5

3 1 2 3

2 1 1

4 10 5 5 1

0

1 2

2

2 1 2

2 1 2

0

### Output for sample input

19

2

## Warning: Time limit is too tight to get accepted using STL. The input file size is around 16 MB

### 11492 - Babel

Time limit: 5.000 seconds

John and Mary are brothers, and are enthusiastic about their courses on foreign languages. Each of the brothers is taking several language courses. When they get home they comment on grammar, vocabulary, culture of the different countries and so on. In one of those conversations they realized some words are common to more than one language, even though the words may have different meanings in the languages. For example, the word "amigo" exists in Portuguese and Spanish and has the same meaning, while "date" is a word that exists in English and French and may have different meanings, since "date" is also a fruit, besides meaning a calendar date. On the other hand, "red" in Spanish is a network, while in English it is a color.

Thrilled by these findings, the brothers decided to write in a notepad all words in common they could think of, associating each word to a pair of languages. Observant and smart, John proposed a challenge to Mary: given one language to start and one language to finish, write down a sequence of words such that the first word is included in the vocabulary of the start language, and the last word is included in the vocabulary of the finish language. Two adjacent words in the sequence must be in the vocabulary of the same language. For example, if the start language is Portuguese and the finish language is French, Mary could write the sequence "amigo actual date" (Portuguese/Spanish, Spanish/English, English/French).

To John's surprise, Mary solved the problem rather easily. Annoyed by his sister's success, he decided to make the problem more difficult: Mary must find a solution in which the sequence has the smallest number of letters in total (not counting spaces between words), and, besides, two consecutive words must not have the same initial letter.

Note that the previous solution is now invalid, as "amigo" and "actual" share the same initial letter. It is possible, however, to find another solution, "amigo red date", with a total length equal to 12.

John did an extensive research on the Internet and compiled an enormous list of words, and challenged Mary to solve the problem. As there may be more than one solution, he asked her to answer if there is a solution, and in that case to answer the number of letters in the best solution. Can you help Mary?

## The Input

The input contains several test cases. The first line of a test case contains one integer *M* (*1 ≤ M ≤ 2000*), representing the total number of words compiled by John. The second line contains two distinct strings *O* and *D*, separated by one space, indicating respectively the start language and the finish language. Each of the next *M* lines contains three strings *I1, I2*and *P*, separated by one space, representing respectively two languages and one word in common between both languages (*I1* and *I2* are always distinct). All strings will have length at least 1 and at most 50, and will be composed of lower case letters only. The same pair of languages may have several words associated to it, but a word *P* will be never repeated in a test case. The end of input is indicated by a line containing only one zero.

## The Output

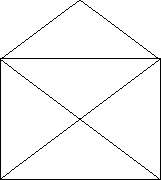
For each test case in the input, your program must print a line with a single integer, the length of the shortest sequence that satisfies John's restrictions, or the word "impossivel" (lowercase, meaning "impossible" in Portuguese) in case it is not possible.

|  |  |
| --- | --- |
| Sample Input 4  portugues frances  ingles espanhol red  espanhol portugues amigo  frances ingles date  espanhol ingles actual  4  portugues alemao  ingles espanhol red  espanhol portugues amigo  frances ingles date  espanhol ingles actual  6  portugues frances  ingles espanhol red  espanhol portugues amigo  frances ingles date  frances espanhol la  portugues ingles a  espanhol ingles actual  0 | Sample Output 12  impossivel  5 |

### 291 - The House Of Santa Claus

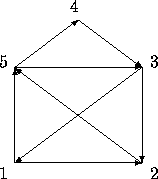
Time limit: 3.000 seconds

In your childhood you most likely had to solve the riddle of the house of Santa Claus. Do you remember that the importance was on drawing the house in a stretch without lifting the pencil and not drawing a line twice? As a reminder it has to look like shown in Figure 1.



**Figure:** The House of Santa Claus

Well, a couple of years later, like now, you have to ``draw'' the house again but on the computer. As one possibility is not enough, we require *all*the possibilities when starting in the lower left corner. Follow the example in Figure 2 while defining your stretch.



**Figure:** This Sequence would give the Outputline 153125432

All the possibilities have to be listed in the outputfile by increasing order, meaning that 1234... is listed before 1235... .

## Output

So, an outputfile could look like this:

12435123

13245123

...

15123421

### 10928 - My Dear Neighbours

Time limit: 3.000 seconds

### Time Limit: 1 second

Manuel is a Portuguese that is vegetarian and does not like his neighbourhood very much because all his neighbours are always doing barbecues and the smoke goes into his house.

Now Manuel has decided to move and he wants to go to a new place with as few neighbours as possible. So he asked your help to write a program to determine the best place where he could live in peace without the barbecue smoke.

## Input

The first line of the input is **N ≤ 30** that indicates the number of test cases. Each test case consists of a number **P**, where **2 ≤ P ≤ 1000**, that indicates the number of places where Manuel can live, each place is numbered from **1** to **P**. Then there will be **P** lines indicating the neighbours of each place, each neighbour is separated by exactly one space. Each place has at least **1** neighbour and at most **P - 1** neighbours, because Manuel can not be a neighbour of himself. For this problem if **P1**has **P2** as his neighbour does not mean that **P2** has **P1** as his neighbour.

Each test case is separated by a blank line.

## Output

For each test case you should print the place that has the minimum number of neighbours. If there are more than one you should print all places separeted by one space and ordered by the indices, the lower indices should come first.

## Sample Input

2

3

2

1 3

2 1

4

2

3

1 4 2

2 1 3

## Sample Output

1

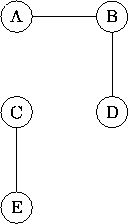
1 2

**Problem setter: S�rgio Queiroz de Medeiros**

### 459 - Graph Connectivity

Time limit: 3.000 seconds

Consider a graph *G* formed from a large number of nodes connected by edges. *G* is said to be connected if a path can be found in 0 or more steps between any pair of nodes in *G*. For example, the graph below is not connected because there is no path from *A* to *C*.



This graph contains, however, a number of subgraphs that are connected, one for each of the following sets of nodes: {*A*}, {*B*}, {*C*}, {*D*}, {*E*}, {*A*,*B*}, {*B*,*D*}, {*C*,*E*}, {*A*,*B*,*D*}

A connected subgraph is maximal if there are no nodes and edges in the original graph that could be added to the subgraph and still leave it connected. There are two maximal connected subgraphs above, one associated with the nodes {*A*, *B*, *D*} and the other with the nodes {*C*, *E*}.

Write a program to determine the number of maximal connected subgraphs of a given graph.

## Input and Output

The input begins with a single positive integer on a line by itself indicating the number of the cases following, each of them as described below. This line is followed by a blank line, and there is also a blank line between two consecutive inputs.

The first line of each input set contains a single uppercase alphabetic character. This character represents the largest node name in the graph. Each successive line contains a pair of uppercase alphabetic characters denoting an edge in the graph. The sample input section contains a possible input set for the graph pictured above.

Input is terminated by a blank line.

For each test case, the output the number of maximal connected subgraphs. The outputs of two consecutive cases will be separated by a blank line.

## Sample Input

1

E

AB

CE

DB

EC

## Sample Output

2

### 11503 - Virtual Friends

Time limit: 10.000 seconds

## Problem C: Virtual Friends

These days, you can do all sorts of things online. For example, you can use various websites to make virtual friends. For some people, growing their social network (their friends, their friends' friends, their friends' friends' friends, and so on), has become an addictive hobby. Just as some people collect stamps, other people collect virtual friends.

Your task is to observe the interactions on such a website and keep track of the size of each person's network.

Assume that every friendship is mutual. If Fred is Barney's friend, then Barney is also Fred's friend.

### Input Specification

The first line of input contains one integer specifying the number of test cases to follow. Each test case begins with a line containing an integer *F*, the number of friendships formed, which is no more than 100 000. Each of the following *F* lines contains the names of two people who have just become friends, separated by a space. A name is a string of 1 to 20 letters (uppercase or lowercase).

### Sample Input

1

3

Fred Barney

Barney Betty

Betty Wilma

### Output Specification

Whenever a friendship is formed, print a line containing one integer, the number of people in the social network of the two people who have just become friends.

### Output for Sample Input

2

3

4

*Ondřej Lhoták*

### 11235 - Frequent values

Time limit: 3.000 seconds

2007/2008 ACM International Collegiate Programming Contest   
University of Ulm Local Contest

# Problem F: Frequent values

You are given a sequence of **n** integers **a1 , a2 , ... , an** in non-decreasing order. In addition to that, you are given several queries consisting of indices **i** and **j** (*1 ≤ i ≤ j ≤ n*). For each query, determine the most frequent value among the integers **ai , ... , aj**.

#### Input Specification

The input consists of several test cases. Each test case starts with a line containing two integers **n** and **q** (*1 ≤ n, q ≤ 100000*). The next line contains **n** integers **a1 , ... , an** (*-100000 ≤ ai ≤ 100000*, for each *i ∈ {1, ..., n}*) separated by spaces. You can assume that for each *i ∈ {1, ..., n-1}: ai ≤ ai+1*. The following **q** lines contain one query each, consisting of two integers **i** and **j** (*1 ≤ i ≤ j ≤ n*), which indicate the boundary indices for the query.

The last test case is followed by a line containing a single *0*.

#### Output Specification

For each query, print one line with one integer: The number of occurrences of the most frequent value within the given range.

#### Sample Input

10 3

-1 -1 1 1 1 1 3 10 10 10

2 3

1 10

5 10

0

#### Sample Output

1

4

3

*A naive algorithm may not run in time!*

### 11297 - Census

Time limit: 8.000 seconds

## Description

This year, there have been many problems with population calculations, since in some cities, there are many emigrants, or the population growth is very high. Every year the ACM (for Association for Counting Members) conducts a census in each region. The country is divided into N^2 regions, consisting of an N x N grid of regions. Your task is to find the least, and the greatest population in some set of regions. Since in a single year there is no significant change in the populations, the ACM modifies the population counts by some number of inhabitants.

## The Input

In the first line you will find N (0 <= N <= 500), in following the N lines you will be given N numbers, which represent, the initial population of city C [i, j]. In the following line is the number Q (Q <= 40000), followed by Q lines with queries:   
  
There are two possible queries:   
  
- "x1 y1 x2 y2" which represent the coordinates of the upper left and lower right of where you must calculate the maximum and minimum change in population.   
  
- "x y v" indicating a change of the population of city C [x, y] by value v.

## The Output

For each query, "x1 y1 x2 y2" print in a single line the greatest and least amount of current population. Separated each output by a space.   
  
**Notice:** There is only a single test case.

|  |  |
| --- | --- |
| **Sample Input** | **Sample Output** |
| 5 5 1 2 3 4 5 0 9 2 1 3 0 2 3 4 1 0 1 2 4 5 8 5 3 1 4 4 q 1 1 2 3 c 2 3 10 q 1 1 5 5 q 1 2 2 2 | 9 0 10 0 9 2 |

Problemsetter: Rodrigo Burgos Domínguez

### 11402 - Ahoy, Pirates!

Time limit: 5.000 seconds

**Problem H**  
**Ahoy, Pirates!**

**Input:**Standard Input

**Output:**Standard Output

In the ancient pirate ages, the Pirate Land was divided into two teams of pirates, namely, the Buccaneer and the Barbary pirates. Each pirate’s team was not fixed, sometimes the opponent pirates attacked and he was taken away to the other pirate team. All on a sudden a magician appeared in the Pirate Land, where he was making transition of pirates from their team to other team at his own will. Of course, handy spells were used. The process of changing team was known as mutating.

There were N pirates and all of the pirates have a unique id from 0 to N-1. The great magician could mutate a bunch of pirates with consecutive id’s to another one.

Suppose there were 100 pirates in the pirate land and all of them were Barbarypirates, then the magician could cast a spell to change pirates with id’s from 10 to 33 to Buccaneer pirates. Then the whole pirate land would have 24 Buccaneer and 76 Barbary pirates.

The magician was very fast casting the spell. Once, God started to dislike this. God had favor for the Buccaneer pirates and God asked the magician, “Tell me, how many of the pirates of index from 2 to 30 are Buccaneer pirates?”. Now the magician was puzzled as he was only efficient in casting spells, not in countingJ

Being clever enough, the magician captured a clever man from the Earth Land. And unfortunately it’s you! Now you have to answer the God’s questions.

**Input**

The first line of input will contain number of test cases T.

For each test case:

The first part of the description will be of the pirate land. There could be up to N (1<=N<=1024000) pirates. Each pirate is either assigned to Buccaneer or Barbary Pirate. Buccaneer pirates are described by ‘1’ (ONE) and Barbarypirates are described by ‘0’ (ZERO). You have to build a string of the pirates’ description. Each case starts with an integer M (M<=100), where M pair lines follow. In each pair of lines (we call it a set), first has an integer **T**(T <= 200) and next one has a nonempty string **Pirates**(consisting of 0 and 1, 0 for Barbary, 1 for Buccaneer, has maximum length of 50). For each pair concatenate the string**Pirates**, **T** times. Concatenate all the resulting M sets of strings to build the pirate description. The final concatenated string describes the pirates from index 0 to end (N-1 for N pirates).

Now the next part of the input will contain queries. First line of next part has an integer Q describing number of queries. Each subsequence Q (1<=Q<=1000) lines describe each query. Each query has a string F or E or I or S and two integers, a and b denoting indexes. The meaning of the query string are follows:

F a b, means, mutate the pirates from index a to b to Buccaneer Pirates.

E a b, means, mutate the pirates from index a to b to Barbary Pirates.

I a b, means, mutate the pirates from index a to b to inverse pirates.

S a b, means, “God’s query” God is asking a question: “Tell me, how many Buccaneer pirates are there from index a to b?”

(a <= b, 0 <= a < n, 0 <= b < n, index range are inclusive)

**Output**

For each test print the case number as the sample output suggests. Then for each of “God’s query”, output the query number, colon (:) and a space and the answer to the query as the sample suggest.

**Sample Input                                                  Output for Sample Input**

|  |  |
| --- | --- |
| 2  2  5  10  2  1000  5  F 0 17  I 0 5  S 1 10  E 4 9  S 2 10  3  3  1  4  0  2  0  2  I 0 2  S 0 8 | Case 1:  Q1: 5  Q2: 1  Case 2:  Q1: 0 |

Explanation:

Case1:

The pirate land is as follows (N = 18)

101010101010001000

Before God’s first query it was as follows

000000111111111111

Case 2:

The pirate land is as follows (N=9)

111000000

Problemsetter: Istiaque Ahmed

Special Thanks to: Mohammad Mahmudur Rahman