apac test
powered by Google Code Jam

Questions asked

— Submissions

Seven-segment Display

| 8pt | Not attempted **159/2058 users** correct (8%) |
| 14pt | Not attempted **34/155 users** correct (22%) |

Super 2048

| 6pt | Not attempted **875/2084 users** correct (42%) |
| 13pt | Not attempted **667/858 users** correct (78%) |

Addition

| 11pt | Not attempted **29/689 users** correct (4%) |
| 19pt | Not attempted **11/26 users** correct (42%) |

Cut Tiles

| 13pt | Not attempted **30/576 users** correct (5%) |
| 16pt | Not attempted **22/29 users** correct (76%) |

— Top Scores

Prowindy	100
MRain	86
Dumbear2	86
Hao.Wu	84
Gyosh	71
LinKin	71
divanshu	70
Krooonal	70
dizem	59
LMH	57

Problem A. **Seven-segment Display**

This contest is open for practice. You can try every problem as many times as you like, though we won't keep track of which problems you solve. Read the Quick-Start Guide to get started.

| Small input 8 points | Solve A-small |
| Large input 14 points | Solve A-large |

Problem

Tom is a boy whose dream is to become a scientist, he invented a lot in his spare time. He came up with a great idea several days ago: to make a stopwatch by himself! So he bought a seven-segment display immediately.

The seven elements of the display are all light-emitting diodes (LEDs) and can be lit in different combinations to represent the arabic numerals like:


However, just when he finished the programs and tried to test the stopwatch, some of the LEDs turned out to be broken! Some of the segments can never be lit while others worked fine. So the display kept on producing some ambiguous states all the time...

Tom has recorded a continuous sequence of states which were produced by the display and is curious about whether it is possible to understand what this display was doing. He thinks the first step is to determine the state which the display will show **next**, could you help him?

Please note that the display works well despite those broken segments, which means that the display will keep on counting down **cyclically** starting from a certain number (can be any one of 0-9 since we don't know where this record starts from). 'Cyclically' here means that each time when the display reaches 0, it will keep on counting down starting from 9 again.

For convenience, we refer the seven segments of the display by the letters A to G as the picture below:
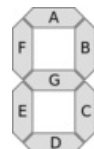
For example, if the record of states is like:

It's not that hard to figure out that ONLY segment B is broken and the sequence of states the display is trying to produce is simply "9 -> 8 -> 7 -> 6 -> 5". Then the next number should be 4, but considering of the brokenness of segment B, the next state should be:

Input

The first line of the input gives the number of test cases, **T**. Each test case is a line containing an integer **N** which is the number of states Tom recorded and a list of the **N** states separated by spaces. Each state is encoded into a 7-character string represent the display of segment A-G, from the left to the right. Characters in the string can either be '1' or '0', denoting the corresponding segment is on or off, respectively.

Output

For each test case, output one line containing "Case #x: y", where x is the test case number (starting from 1). If the input unambiguously determines the next state of the display, y should be that next state (in the same format as the input). Otherwise, y should be "ERROR!".

Limits

1 ≤ **T** ≤ 2000.

Small dataset

1 ≤ **N** ≤ 5.

Large dataset

1 ≤ **N** ≤ 100.

Sample

```
Input

4
1 1111111
2 0000000 0001010
3 0100000 0000111 0000011
5 1011011 1011111 1010000 1011111 1011011
```

```
Output

Case #1: 1110000
Case #2: ERROR!
Case #3: 0100011
Case #4: 0010011
```

Problem B. **Super 2048**

This contest is open for practice. You can try every problem as many times as you like, though we won't keep track of which problems you solve. Read the **Quick-Start Guide** to get started.

| Small input 6 points | Solve B-small |
| Large input 13 points | Solve B-large |

Problem

2048 is a famous single-player game in which the objective is to slide tiles on a grid to combine them and create a tile with the number 2048.

2048 is played on a simple 4 x 4 grid with tiles that slide smoothly when a player moves them. For each movement, the player can choose to move all tiles in 4 directions, left, right, up, and down, as far as possible at the same time. If two tiles of the same number collide while moving, they will merge into a tile with the total value of the two tiles that collided. **In one movement, one newly created tile can not be merged again and always is merged with the tile next to it along the moving direction first.** E.g. if the three "2" are in a row "2 2 2" and the player choose to move left, it will become "4 2 0", the most left 2 "2" are merged.
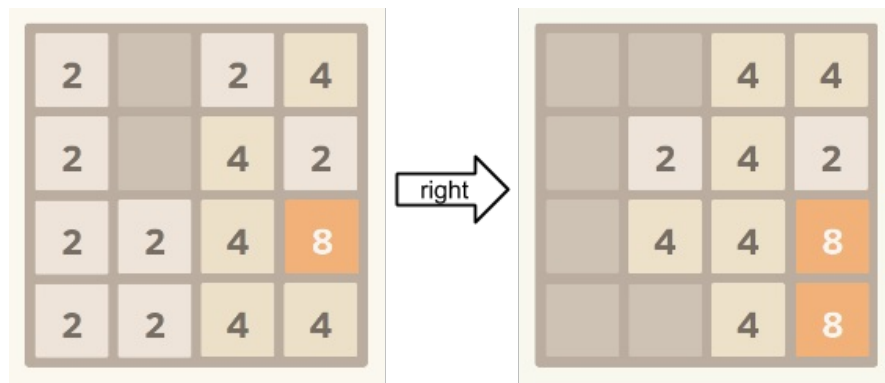
The above figure shows how 4 x 4 grid varies when player moves all tiles 'right'.

Alice and Bob accidentally find this game and love the feel when two tiles are merged. After a few round, they start to be bored about the size of the board and decide to extend the size of board to **N** x **N**, which they called the game "Super 2048".

The big board then makes them dazzled (no zuo no die -_-|). They ask you to write a program to help them figure out what the board will be looked like after all tiles move to one specific direction on a given board.

Input

The first line of the input gives the number of test cases, **T**. **T** test cases follow. The first line of each test case gives the side length of the board, **N**, and the direction the tiles will move to, **DIR**. **N** and **DIR** are separated by a single space. **DIR** will be one of four strings: "left", "right", "up", or "down".

The next **N** lines each contain **N** space-separated integers describing the original state of the board. Each line represents a row of the board (from top to bottom); each integer represents the value of a tile (or 0 if there is no number at that position).

Output

For each test case, output one line containing "Case #x:", where x is the test case number (starting from 1). Then output **N** more lines, each containing **N** space-separated integers which describe the board after the move in the same format as the input.

Limits

Each number in the grid is either 0 or a power of two between 2 and 1024, inclusive.

Small dataset

1 ≤ **T** ≤ 20
1 ≤ **N** ≤ 4

Large dataset

$1 \le T \le 100$
$1 \le N \le 20$

Sample

```
Input                    Output

3                        Case #1:
4 right                  0 0 4 4
2 0 2 4                  0 2 4 2
2 0 4 2                  0 4 4 8
2 2 4 8                  0 0 4 8
2 2 4 4                  Case #2:
10 up                    4 0 0 0 0 0 0 0 0 0
2 0 0 0 0 0 0 0 0 0      4 0 0 0 0 0 0 0 0 0
2 0 0 0 0 0 0 0 0 0      4 0 0 0 0 0 0 0 0 0
2 0 0 0 0 0 0 0 0 0      4 0 0 0 0 0 0 0 0 0
2 0 0 0 0 0 0 0 0 0      4 0 0 0 0 0 0 0 0 0
2 0 0 0 0 0 0 0 0 0      0 0 0 0 0 0 0 0 0 0
2 0 0 0 0 0 0 0 0 0      0 0 0 0 0 0 0 0 0 0
2 0 0 0 0 0 0 0 0 0      0 0 0 0 0 0 0 0 0 0
2 0 0 0 0 0 0 0 0 0      0 0 0 0 0 0 0 0 0 0
2 0 0 0 0 0 0 0 0 0      0 0 0 0 0 0 0 0 0 0
2 0 0 0 0 0 0 0 0 0      Case #3:
3 right                  0 2 4
2 2 2                    0 4 8
4 4 4                    0 8 16
8 8 8
```

Questions asked

Problem C. Addition

This contest is open for practice. You can try every problem as many times as you like, though we won't keep track of which problems you solve. Read the Quick-Start Guide to get started.

| Small input 11 points | [Solve C-small] |
| Large input 19 points | [Solve C-large] |

Problem

Six years ago, a robot, Bob, with infant's intelligence has been invented by an evil scientist, Alice.

Now the robot is six years old and studies in primary school. Addition is the first operation he learned in math. Due to his strong reasoning ability, he could now conclude a+b=12 from a=2 and b=10.

Alice wanted to test Bob's addition skills. Some equations were given to Bob in form of a=2, b=10, c=4, and Bob has to find out the answers of questions like a+b, a+c, etc.

Alice checked Bob's answers one by one in the test papers, and no mistake has been found so far, but Alice lost the given equations after a cup of coffee poured on them. However she has some of Bob's correct answers, e.g. a+b=12, a+c=6, c+d=5. She wants to continue with the checkable equations, e.g. b+d=11 could be concluded by a+b=12, a+c=6, c+d=5, and thus the question b+d is checkable.

To prevent the artificial intelligence technology from being under the control of Alice, you disguised yourself as her assistant. Now Alice wants you to figure out which of the rest of questions are checkable and their answers.

Input

The first line of the input gives the number of test cases, **T**. **T** test cases follow.

The first line of each test case contains a single integer **N**: the number of correctly answered questions. Each of the next **N** lines contain one correctly answered question in the form "**x+y=z**", where **x** and **y** are names of variables and **z** is a decimal integer.

The next line contains a single integer **Q**: the number of remaining questions. Each of the next **Q** lines contain one question in the form "**x+y**", where **x** and **y** are names of variables.

Output

For each test case, the first line of output contains "Case #**x**:", where **x** is the test case number (starting from 1). For each question in the input that was checkable, output a single line with the answer in the form "**x+y=z**", where **x** and **y** are names of variables and **z** is a decimal integer. Questions should be listed in the same order as they were given in the input. Please do **NOT** ignore duplicated questions, since Alice would fire you if you pointed any mistake of hers.

Limits

Names of variables are strings of lowercase English letters. Each name contains at most 10 characters.

$-200000 \le z \le 200000$

There is no contradiction in the answered questions and if the answer is checkable, the result is an integer.

Small dataset

$T \le 10$

$N \le 10$

$Q \le 10$

Large dataset

$T \le 3$

$N \le 5000$

$Q \le 5000$

Sample

```
Input                          Output

2                              Case #1:
2                              apple+banana=10
apple+banana=10                apple+banana=10
coconut+coconut=12             banana+apple=10
5                              Case #2:
apple+banana                   a+d=3
apple+banana                   b+c=3
apple+apple
banana+apple
peach+apple
3
a+b=3
b+c=3
c+d=3
4
a+c
a+d
b+c
b+d
```

---

apac test
powered by Google Code Jam

Problem D. **Cut Tiles**

This contest is open for practice. You can try every problem as many times as you like, though we won't keep track of which problems you solve. Read the **Quick-Start Guide** to get started.

Small input
13 points

[Solve D-small]

Large input
16 points

[Solve D-large]

Problem

Enzo is doing renovation for his new house. The most difficult part is to buy exactly the right number of tiles. He wants **N** tiles of different sizes. Of course they have to be cut from the tiles he bought. All the required tiles are square. The lengths of side of the tiles are 2^{S_1}, 2^{S_2}, ..., 2^{S_N}. He can only buy a lot of tiles sized **M*****M**, and he decides to only cut tiles parallel to their sides for convenience. How many tiles does he need to buy?

Input

The first line of the input gives the number of test cases: **T**. **T** lines follow. Each line start with the number **N** and **M**, indicating the number of required tiles and the size of the big tiles Enzo can buy. **N** numbers follow: S_1, S_2, ... S_N, showing the sizes of the required tiles.

Output

For each test case, output one line containing "Case #x: y", where x is the test case number (starting from 1) and y is the number of the big tiles Enzo need to buy.

Limits

$1 \le 2^{S_k} \le M \le 2\text{^}31\text{-}1$.

Small dataset

$1 \le T \le 100$.
$1 \le N \le 20$.

Large dataset

$1 \le T \le 1000$.
$1 \le N \le 500$.

Sample

```
Input                    Output

4                        Case #1: 1
1 6 2                    Case #2: 2
2 6 2 2                  Case #3: 1
3 6 2 1 1                Case #4: 2
7 277 3 8 2 6 1 3 6
```

Powered by

Google Cloud Platform