Round D APAC Test 2017

A. Vote

B. Sitting

C. Codejamon Cipher

D. Stretch Rope

Questions asked

Submissions

Vote

5pt Not attempted **1360/2559 users** correct (53%)

8pt Not attempted **913/1257 users** correct (73%)

Sitting

9pt Not attempted **683/1467 users** correct (47%)

10pt Not attempted **305/472 users** correct (65%)

Codejamon Cipher

7pt Not attempted **653/819 users** correct (80%)

16pt Not attempted **348/624 users** correct (56%)

Stretch Rope

15pt Not attempted **477/655 users** correct (73%)

30pt Not attempted **36/146 users** correct (25%)

Top Scores

jinzhao	100
axp	100
wcwswswws	100
t3cmax	100
prabowo	100
ZJiaQ	100
BoyZhou	100
sgtlaugh	100
YeYifan	100
shyoshyo	100

Problem A. **Vote**

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 5 points	Solve A-small
Large input 8 points	Solve A-large

Problem

A and B are the only two candidates competing in a certain election. We know from polls that exactly **N** voters support A, and exactly **M** voters support B. We also know that **N** is greater than **M**, so A will win.

Voters will show up at the polling place one at a time, in an order chosen uniformly at random from all possible (**N** + **M**)! orders. After each voter casts their vote, the polling place worker will update the results and note which candidate (if any) is winning so far. (If the votes are tied, neither candidate is considered to be winning.)

What is the probability that A stays in the lead the entire time -- that is, that A will always be winning after every vote?

Input

The input starts with one line containing one integer **T**, which is the number of test cases. Each test case consists of one line with two integers **N** and **M**: the numbers of voters supporting A and B, respectively.

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 probability that A will always be winning after every vote.

y will be considered correct if y is within an absolute or relative error of 10^{-6} of the correct answer. See the FAQ for an explanation of what that means, and what formats of real numbers we accept.

Limits

1 ≤ **T** ≤ 100.

Small dataset

0 ≤ **M** < **N** ≤ 10.

Large dataset

0 ≤ **M** < **N** ≤ 2000.

Sample

Input	Output
2 2 1 1 0	Case #1: 0.33333333 Case #2: 1.00000000

In sample case #1, there are 3 voters. Two of them support A -- we will call them A1 and A2 -- and one of them supports B. They can come to vote in six possible orders: A1 A2 B, A2 A1 B, A1 B A2, A2 B A1, B A1 A2, B A2 A1. Only the first two of those orders guarantee that Candidate A is winning after every vote. (For example, if the order is A1 B A2, then Candidate A is winning after the first vote but tied after the second vote.) So the answer is 2/6 = 0.333333...

In sample case #2, there is only 1 voter, and that voter supports A. There is only one possible order of arrival, and A will be winning after the one and only vote.

Problem B. **Sitting**

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 9 points	Solve B-small
Large input 10 points	Solve B-large

Problem

The ***Codejamon*** game is on fire! Many players have gathered in an auditorium to fight for the World Championship. At the opening ceremony, players will sit in a grid of seats with **R** rows and **C** columns.

The competition will be intense, and the players are sensitive about sitting near too many of their future opponents! A player will feel too crowded if another player is seated directly to their left *and* another player is seated directly to their right. Also, a player will feel too crowded if one player is seated directly in front of them *and* another player is seated directly behind them.

What is the maximum number of players that can be seated such that no player feels too crowded?

Input

The first line of the input gives the number of test cases, **T**. **T** test cases follow. Each test case consists of one line with two integers **R** and **C**: the number of rows and columns of chairs in the auditorium.

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 maximum number of players that can be seated, as described in the problem statement.

Limits

1 ≤ **T** ≤ 100.

Small dataset

1 ≤ **R** ≤ 5.
1 ≤ **C** ≤ 5.

Large dataset

1 ≤ **R** ≤ 100.
1 ≤ **C** ≤ 100.

Sample

Input	Output
3 2 2 2 3 4 1	Case #1: 4 Case #2: 4 Case #3: 3

In sample case #1, we can fill all seats, and no player will feel too crowded.

In sample case #2, each row has three seats. We can't put three players in a row, since that would make the middle player feel too crowded. One optimal solution is to fill each of the first two columns, for a total of four players.

In sample case #3, one optimal solution is to fill the first two rows and the last row, for a total of three players.

Round D APAC Test 2017

A. Vote

B. Sitting

C. Codejamon Cipher

D. Stretch Rope

Questions asked

Submissions

Vote

5pt Not attempted **1360/2559 users** correct (53%)

8pt Not attempted **913/1257 users** correct (73%)

Sitting

9pt Not attempted **683/1467 users** correct (47%)

10pt Not attempted **305/472 users** correct (65%)

Codejamon Cipher

7pt Not attempted **653/819 users** correct (80%)

16pt Not attempted **348/624 users** correct (56%)

Stretch Rope

15pt Not attempted **477/655 users** correct (73%)

30pt Not attempted **36/146 users** correct (25%)

Top Scores

jinzhao	100
axp	100
wcwswswws	100
t3cmax	100
prabowo	100
ZJiaQ	100
BoyZhou	100
sgtlaugh	100
YeYifan	100
shyoshyo	100

Problem C. Codejamon Cipher

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 7 points	Solve C-small
Large input 16 points	Solve C-large

Problem

The ***Codejamon*** monsters talk in enciphered messages. Here is how it works:

Each kind of monster has its own unique *vocabulary*: a list of **V** different words consisting only of lowercase English letters. When a monster speaks, it first forms a sentence of words in its vocabulary; the same word may appear multiple times in a sentence. Then, it turns the sentence into an enciphered string, as follows:

1. Randomly shuffle each word in the sentence.
2. Remove all spaces.

Understanding the monsters can bring you huge advantages, so you are building a tool to do that. As the first step, you want to be able to take an enciphered string and determine how many possible original sentences could have generated that enciphered string. For example, if a monster's vocabulary is *["this", "is", "a", "monster", "retsnom"]*, and it speaks the enciphered string *"ishtsiarestmon"*, there are four possible original sentences:

- "is this a monster"
- "is this a retsnom"
- "this is a monster"
- "this is a retsnom"

You have **S** enciphered strings from the same monster. For each one, can you figure out the number of possible original sentences?

IMPORTANT: Since the output can be a really big number, we only ask you to output the remainder of dividing the result by the prime $10^9 + 7$ (1000000007).

Input

The first line of the input gives the number of test cases, **T**. **T** test cases follow. Each test case consists of one line with two integers **V** and **S**, the size of the monster's vocabulary and the number of enciphered strings. Then, **V** lines follow; each contains a single string of lowercase English letters, representing a word in the monster's vocabulary. Finally, **S** lines follow. Each contains a string consisting only of lowercase English letters, representing an enciphered sentence. It is guaranteed that all enciphered sentences are valid; that is, each one has at least one possible original sentence.

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 a space separated list of of **S** integers: the answers (modulo $10^9 + 7$) for each enciphered sentence, in the order given in the input, as described in the problem statement.

Limits

$1 \le \mathbf{T} \le 100$.
$1 \le \mathbf{S} \le 5$.

Small dataset

1 ≤ the length of each word in the monster's vocabulary ≤ 5.
1 ≤ the length of the enciphered string ≤ 50.
$5 \le \mathbf{V} \le 10$.

Large dataset

1 ≤ the length of each word in the monster's vocabulary ≤ 20.
2000 ≤ the length of the enciphered string ≤ 4000.
$200 \le \mathbf{V} \le 400$.

Sample

Input	Output

Input	Output
`` 2 5 1 this is a good day sithsiaodogyad 5 3 pt ybsb xnydt qtpb kw xnydttbpqtpqb yxdtntpbsby ptptxytdnsbybpt ```	Case #1: 2 Case #2: 1 1 1

Submissions

Vote

5pt Not attempted **1360/2559 users** correct (53%)

8pt Not attempted **913/1257 users** correct (73%)

Sitting

9pt Not attempted **683/1467 users** correct (47%)

10pt Not attempted **305/472 users** correct (65%)

Codejamon Cipher

7pt Not attempted **653/819 users** correct (80%)

16pt Not attempted **348/624 users** correct (56%)

Stretch Rope

15pt Not attempted **477/655 users** correct (73%)

30pt Not attempted **36/146 users** correct (25%)

Top Scores

jinzhao	100
axp	100
wcwswswws	100
t3cmax	100
prabowo	100
ZJiaQ	100
BoyZhou	100
sgtlaugh	100
YeYifan	100
shyoshyo	100

Problem D. Stretch Rope

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 15 points	Solve D-small
Large input 30 points	Solve D-large

Problem

Mary likes playing with rubber bands. It's her birthday today, and you have gone to the rubber band shop to buy her a gift.

There are **N** rubber bands available in the shop. The i-th of these bands can be stretched to have any length in the range $[\mathbf{A_i}, \mathbf{B_i}]$, inclusive. Two rubber bands of range [a, b] and [c, d] can be connected to form one rubber band that can have any length in the range [a+c, b+d]. These new rubber bands can themselves be connected to other rubber bands, and so on.

You want to give Mary a rubber band that can be stretched to a length of *exactly* **L**. This can be either a single rubber band or a combination of rubber bands. You have **M** dollars available. What is the smallest amount you can spend? If it is impossible to accomplish your goal, output `IMPOSSIBLE` instead.

Input

The first line of the input gives the number of test cases, **T**. **T** test cases follow. Each test case starts with 3 integers **N**, **M**, **L**, the number of rubber bands available in the shop, the number of dollars you have and the desired rubber band length. Then **N** lines follow. Each line represents one rubber band and consists of 3 integers, $\mathbf{A_i}$, $\mathbf{B_i}$, and $\mathbf{P_i}$. $[\mathbf{A_i}, \mathbf{B_i}]$ is the inclusive range of lengths that the i-th rubber band can stretch to, and $\mathbf{P_i}$ is the price of the i-th rubber band in dollars.

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 `IMPOSSIBLE` if you cannot buy rubber bands to satisfy the goal described above, or otherwise an integer: the minimum price you can pay.

Limits

$1 \le \mathbf{T} \le 100$.
$1 \le \mathbf{P_i} \le \mathbf{M}$.
$1 \le \mathbf{L} \le 10000$.
$1 \le \mathbf{A_i} \le \mathbf{B_i} \le 10000$.

Small dataset

$1 \le \mathbf{N} \le 10$.
$1 \le \mathbf{M} \le 100$.

Large dataset

$1 \le \mathbf{N} \le 1000$.
$1 \le \mathbf{M} \le 1000000000$.

Sample

Input	Output
2 3 8 6 3 5 2 4 4 3 1 2 5 3 11 14 1 3 4 5 5 3 2 6 5	Case #1: 7 Case #2: IMPOSSIBLE

In sample case #1, none of the rubber bands in the shop are long enough on their own. It will not work to buy the two cheapest rubber bands and stick them together, because the new band would have a stretch range of [7, 9], which does not include 6. (Remember, the rubber band must be able to stretch to a length of *exactly* **L**.) The optimal solution is to buy the rubber bands costing 2 and 5 and stick them together; the new band has a stretch range of [4, 7], which does include 6. You have 8 dollars, so you can afford the total cost of 7 dollars.

In sample case #2, you need to buy all of the rubber bands to be able to stretch to length 14. That would cost 12 dollars, but you only have 11, so this case is `IMPOSSIBLE`.
