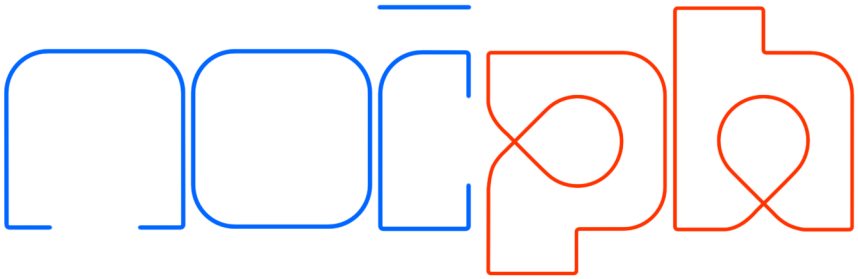


2024

National Olympiad in Informatics

Finals Round 2

Important! Read the following:

Hidden Test Cases. Your solution will be checked by running it against one or more (usually several) hidden test cases. You will not have access to these cases, but a correct solution is expected to handle them correctly.

Strict Output Format. The output checker is **strict**. Follow these guidelines strictly:

- It is **space sensitive**. Do not output extra leading or trailing spaces. Do not output extra blank lines unless explicitly stated.
- It is **case sensitive**. So, for example, if the problem asks for the output in lowercase, follow it.
- Do not print any tabs. (No tabs will be required in the output.)
- Do not output anything else aside from what's asked for in the Output section. So, do not print things like "`Please enter t`".

Not following the output format strictly and exactly will likely result in the verdict "*Output isn't correct*".

Use Standard I/O. Do not read from, or write to, a file. You must read from the standard input and write to the standard output.

Submit Code Only. Only include **one** file when submitting: the source code (.cpp, .py, etc.) and nothing else.

No Java Package. For Java submissions, do not include a `package` line.

No Weird Filenames. Only use letters, digits and underscores in your filename. Do not use spaces or other special symbols.

Use Fast I/O. Many problems have large input file sizes, so use fast I/O. For example:

- In C/C++, use `scanf` and `printf`.
- In Python, use `sys.stdin.readline()`

Flush On Interactive Problems. On interactive problems, make sure to **flush** your output stream after printing.

- In C++, use `fflush(stdout);` or `cout << endl;`
- In Python, use `sys.stdout.flush()` or `print` with `flush=True`
- For more details, including for other languages, ask a question/clarification through CMS.

Good luck and enjoy the contest! 😃

Contents

Notes

- Many problems have large input file sizes, so use fast I/O. For example:
  - In C/C++, use `scanf` and `printf`.
  - In Python, use `sys.stdin.readline()`
- On interactive problems, make sure to **flush** your output stream after printing.
  - In C++, use `fflush(stdout);` or `cout << endl;`
  - In Python, use `sys.stdout.flush()` or `print` with `flush=True`
  - For more details, including for other languages, ask a question/clarification through CMS.

Good luck and enjoy the problems!

Problem A
Biggest Brain Academy

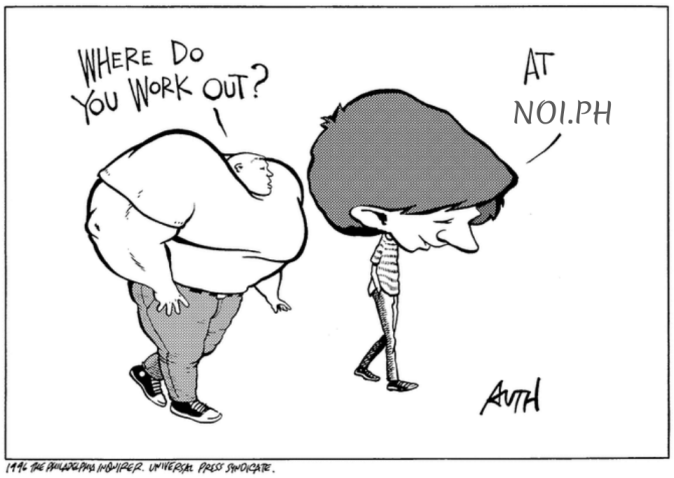
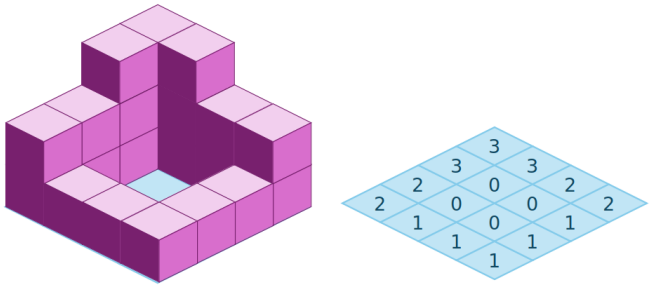


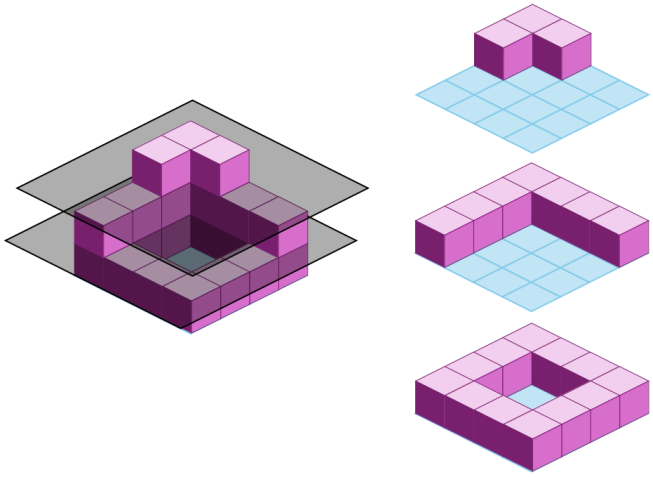
Figure 1: *Original by The Philadelphia Inquirer (1996), modified into a "funny meme".*

NOI.PH is proud to partner with Nine-Ten-Doh for their newest educational video game, "Biggest Brain Academy". Players complete a series of IQ-test-esque brain teasers, whose completions reward the player with "Brain Points". After every 1024 Brain Points, the game congratulates you with a shower of confetti, and a message appears that tells you how proud it is of you for being so smart and witty and attractive. Then your player avatar gets a cool hat.

Today, we're going to develop a new minigame for spatial reasoning! Consider a grid with r rows and c columns. A stack of cubes is placed on each square, with the number of cubes in each stack being an integer from 0 to H. We call this set-up a **stack diagram**. For example, with $r = c = 4$ and $H = 3$, we could have this:



The minigame then takes H cross-sectional slices of the diagram, with each cross section encoding the state of the stack diagram at some fixed elevation. Each cross-section is an $r \times c$ grid that encodes, for each square, whether a cube exists at that location at this cross-section's elevation. The following diagram illustrates how the $H = 3$ cross-sections are produced from the stack diagram shown earlier.



For the actual minigame, you will be presented with H grids of size $r \times c$. Is there a way to properly sequence these grids such that, if read in that order, they correspond to the cross-sections of some stack diagram, from *bottom to top*? If yes, is it unique? And if it is unique, please provide the order!

Input Format

The first line of input contains three space-separated integers r and c and H.

Then, descriptions of H grids follow. Each description begins with a single integer i, the index of this grid (which always counts up from 1 to H, in order). Then, this is followed by r lines that each contain a string of length c, representing the grid. The character `.` (dot) represents space without a cube, and the character `#` represents space with a cube.

Output Format

First, output a line containing the word `NO` or `MANY` or `UNIQUE`, depending on if there exists no solution, many solutions, or a unique solution, respectively.

If `UNIQUE`, also output another line containing H space-separated integers, some permutation of the integers from 1 to H, corresponding to the indices of the grids in the order that they should be arranged so that they correspond to the cross-sections of a stack diagram read from bottom-to-top.

Constraints

For all subtasks
$1 \leq r, c \leq 20$ $2 \leq H \leq 500$

Subtask	Points	Constraints
1	**28**	$r, c \leq 2$
2	**28**	$H \leq 5$
3	**28**	$r = 1$
4	**16**	No further constraints.

Sample I/O

Input 1	Output 1
4 4 3 1 #### #... #... #... 2 #### #..# #..# #### 3 ##.. #...	UNIQUE 2 1 3

Input 2	Output 2
2 3 2 1 .#. #.# 2 #.# .#.	NO

Problem B
Collecting a Construction Permit

Shynthia once constructed a city plan. She's a bit shy, but mostly shocked, about her somewhat shoddy city plan. Shynthia had a shower thought of shoring up her shifty city plan before she ships it. But after shampooing and shaking her shoulder-length hair, she shrewdly saw how shortsighted she was. "How sheepish of me," Shynthia said while shivering in the shower, "to shrink my showpiece, my shining city plan, out of shame!" She decided she should show some shareholders her city plan and get a permit so she can start construction.

To get a construction permit, she needs to send an application to an office of the Metropolitan Alliance to Develop and Fix All Kantos and Sidewalks. There are n offices she can send an application to, but not all of the offices are the same. The ith office has a *corruption value* c_i, a real number between 0 and 1 (inclusive), and an integer *bureaucracy value* b_i. When Shynthia sends her application to the ith office, it takes b_i days before she gets a response. Her application is rejected with probability c_i, and she gets a construction permit with probability $1 - c_i$.

Once Shynthia is rejected by an office, she can't apply to the same office again. She also can't apply to two offices at the same time; she must wait for the result before reapplying at another office. It takes no time between getting a rejection and reapplying.

Shynthia is smart, so whenever she has a choice of which office to apply to, she makes the choice that will minimize the expected number of days before her application gets approved. What's the expected number of days before Shynthia gets a construction permit?

It is guaranteed that one of the offices has a corruption value of 0, thus, her application will surely get approved at some point.

Suppose that Shynthia performs a random experiment involving simulating the process of her applying for a permit, and then records the number of days that it took for her to get the permit. She performs this experiment more and more times, all the while keeping track of the running average of the number of days required in each simulation. As she performs more and more experiments, this running average converges to a value, which is the expected number of days.[1]

Input Format

The first line of input contains a single integer t, the number of test cases. Then, the descriptions of the t test cases follow.

The first line of each test case contains a single integer n, the number of offices.

[1] More precisely, it can be shown that there's a value e (the expected value) such that the running average converges to e with probability 1.

The next n lines describe the offices. The ith line contains a real number c_i, the corruption value, followed by a space, followed by the integer b_i, the bureaucracy value.

Output Format

For each test case, output a single line containing a single decimal value, the expected number of days before Shynthia gets a construction permit.

Your answer will be accepted if it has an absolute or relative error of at most 10^{-6} from the judge's answer. In symbols, let ans_{you} be your answer, and let ans_{judge} be the judge's answer. Your answer will be accepted if

$$\frac{|ans_{\text{you}} - ans_{\text{judge}}|}{\max(1, |ans_{\text{you}}|, |ans_{\text{judge}}|)} \leq 10^{-6}$$

In short, **make sure you print your answer to sufficiently many decimal places**. For example,

- In C++, `#include` the `<iomanip>` library, and use

  ```
  cout << fixed << setprecision(12) << ans << endl;
  ```

  to output `ans` to exactly 12 decimal places.
- In Python, use `print(f'{ans:.12f}')` to output `ans` to exactly 12 decimal places.

Constraints

For all subtasks
$1 \leq t \leq 100\,000$ $1 \leq n \leq 300\,000$ $0 \leq c_i \leq 1$ $0 \leq b_i \leq 10^9$ For each test case, $c_i = 0$ for some $1 \leq i \leq n$.

Subtask	Points	Constraints
1	**19**	The sum of n over all test cases is at most 6.
2	**12**	The sum of n over all test cases is at most 50.
3	**11**	The sum of n over all test cases is at most 3 000.
4	**58**	The sum of n over all test cases is at most 300 000.

Sample I/O

Input	Output
2 1 0.0 100 2 0.0 100 0.8 1	100.00000000000000000000 81.00000000000000000000

Explanation

We explain the second test case. Shynthia has two options:

- Shynthia applies to the first city first. After 100 days, her application gets accepted. The expected number of days in this case is 100.
- Shynthia applies to the second city first. Then 20% of the time, her application is accepted, and it took her 1 day to get the permit. The other 80% of the time, she gets rejected. She then applies to the first city, and 100 days later, gets accepted; in this case, it took her 101 days to get the permit. The expected number of days in this case is $20\% \times 1 + 80\% \times 101 = 81$.

In the option that minimizes the expected number of days, the expected number of days before getting a permit is 81, which is the answer.

Problem C
Like the Molave

In 2011, the USA was astonished to learn that Captain America—war hero and legend—was revived by S.H.I.E.L.D. after his frozen body was discovered in the Arctic. Similarly, the Philippines was also wonderstruck when the body of Jose Rizal, presumed executed at Bagumbayan, was actually found perfectly preserved within two split halves of a molave tree.

Jose Rizal would finally publish Makamisa, completing the previously unfinished trilogy with Noli and Fili, uniting the country under a single banner and ushering in an era of peace and prosperity for the Philippines. Jose Rizal also joined the Avengers initiative, and was witnessed personally leading the charge at the Battles of New York and Sokovia, wearing his signature Medal of Honor. It is said that he saved the lives of countless civilians from Loki and Ultron's troops.

Man, I Love Jose Rizal.

Jose Rizal's signature ability is to summon a molave tree: firm, resilient, staunch, rising on the hillside, unafraid, strong in its own fiber, yes, the Molave! The Molave begins as a simple seed, but out of the silent dreaming from the seven thousand fold silence, it shall emerge!

Formally, the Molave is a tree (in the graph theory sense): it is a collection of nodes, where each node has a **label** and **strength** value, both integers. Also, each node (except the so-called **root**) has a **parent**, which is some other node in the tree. The **path to the root** from some node u is the collection of nodes visited by starting at u and then repeatedly "climbing" from the current node to its parent, until one reaches the root (the node u itself is always in this path).

The Molave begins as a single root node with label 0 and strength 0, and no parent. Then, Rizal has four commands:

- "`GROW` i ℓ" means a new "branch" of length ℓ grows from the node labeled i. Formally, let n be the largest label among the nodes *currently* on the tree (we see later that nodes may be deleted from the tree). Construct nodes with labels $n+1$, $n+2$, $\ldots$, $n+\ell$ such that:
  - The parent of node $n+1$ is node i
  - For $n+1 < u \leq n+\ell$, the parent of the node u created by this command is node $u-1$.

  The strengths of all these nodes are 0.
- "`CUT` i" means to "cut" node i and all nodes in its subtree. Formally, consider the set of all nodes u such that the path to the root from u contains the node i—remove all these nodes from the tree.
- "`EDIT` i s" means that the strength of node i becomes s.

- "`ROOT` i" means you should output a value: the sum of the strengths of all nodes in the path to the root from i.

Rizal believes that the future of a nation is not secured unless we can confidently pass the reins down to the next generation. Thus, he tasks you with recreating his molave technique in code!

Input Format

The first line of input contains a single integer q, the number of commands to be performed. The next q lines each describe a command, each in one of the formats described above.

Output Format

For each "`ROOT` i" command, output a single integer: the sum of the strengths of all nodes in the path to the root from node i, given the current state of the molave when the command was made.

Constraints

For all subtasks
$0 \le q \le 100\,000$ $1 \le \ell \le 10^9$ and $1 \le s \le 10^9$ in each command where they appear The node i always exists at that time, for each i in each command. For all `CUT` commands, $i \neq 0$.

Subtask	Points	Constraints
1	**38**	$q \le 1000$ and $\ell = 1$ for all `GROW` commands.
2	**25**	$\ell = 1$ for all `GROW` commands.
3	**27**	$q \le 1000$
4	**10**	No further constraints.

Sample I/O

Input	Output
4 GROW 0 2 CUT 2 EDIT 1 5 ROOT 1	5

Problem D
Infinity War

The onslaught is fierce and unrelenting. The din of the surrounding chaos fades into a dull noise. From the midst of his army, a titan arises, his gauntlet resplendent with the powers of the cosmos. The villain nears the end of his decade-long journey, one heartbeat away from victory. Only you stand in his way. You are the last line of defense. The last light of hope.

This is it. The climax. The final battle.

You are not alone. You had embarked on a journey of your own, leading up to this moment. The bonds you had forged will never leave you. *Every single NOI.PH character from the years 2014 to 2024* stands by your side, the fruit of a decade-long legacy.

You will be given a file (called `noi.txt`, downloadable from CMS) containing n names, the full list of NOI.PH characters who are with you in this final battle. But Thanos is mighty—whenever he snaps his fingers, *half of the existing characters vanish*. Formally, the following happens:

- Let ℓ be the number of characters remaining.
- Consider all subsequences of length $\lfloor \ell/2 \rfloor$, **not necessarily contiguous** *(it seems the Doctors had a miscalculation...)*
- One of these subsequences is independently selected uniformly at random.
- All characters in that subsequence *vanish*, trapped inside the Soul Stone.
    - All remaining characters "move to close the gaps".

Thanos secretly selects an integer k uniformly at random from 1 to $\lfloor \log_2 n \rfloor$, and then snaps his finger k times. You do not know k, and your goal is to *guess* k to the best of your ability, based on which characters managed to survive.

This is an interactive problem. You do not have access to the complete list of the still-living characters. You only have one type of query available:

- Ask for the name of the ith still-living character.
    - The list is 1-indexed.
    - If i exceeds the number of remaining characters, the judge instead responds with a name uniformly randomly selected from the names of all still-living characters.

Note that the relative ordering of the still-living characters will be the same as their ordering in the original list.

Your score is dependent on two factors: how close your guess is to the actual value of k (the closer the better), and how many queries you made before you gave your guess (the fewer the better). The exact scoring formula is given below.

Attachment

The file `noi.txt` (downloadable in the CMS interface) lists all n NOI.PH characters from 2014 to 2024. It contains one string per line, each of which is considered to be a *distinct* NOI.PH character. Thus, it has exactly n lines. It is guaranteed that every name does not contain any whitespace, and that no two names are exactly the same.

Interaction

Your program will be run exactly once and be made to interact with the judge program. There will be $t = 2\,500$ test cases. The maximum number of queries that can be performed is $q = 250\,000$.

The interaction starts with the judge printing three space-separated integers t, q and n on a line. (Note that n is the number of lines of the `noi.txt` attachment.) After that, t test cases are run.

For each test case, the judge first uniformly chooses the integer k from 1 to $\lfloor \log_2 n \rfloor$ and simulates k snaps. The judge then waits for queries.

To query, print the line "`Q` i", where i is an integer. The judge responds by printing the name of the ith still-living NOI character. If i exceeds the number of still-living NOI characters, a random still-living NOI character's name will be given.

You can answer the test case by printing the line "`A` k", where k is your guess for the value of k. The judge responds to this line with the line "`OK`", and the test case ends.

For a query to be valid, the integer i must be between 1 and 10^9, inclusive. For a guess for k to be valid, it must be between 1 and 100, inclusive. If either of these conditions fails, or if your program performs a query after already having done q queries, then the judge prints the line "`FAIL`" and exits immediately, and your program should exit as well.

Notes:

- Scores are reproducible; the judge uses a fixed random seed.
- Make sure to **flush** your output stream after printing; otherwise, the judge will not be able to receive your output, and it will wait forever.
    - In C++, use `fflush(stdout);` to flush the stream, or `cout << endl;` to print a newline character and then flush the stream.
    - In Python, use `sys.stdout.flush()` to flush the stream, or use `print` with `flush=True` to print a newline character and then flush the stream.
    - For more details, including for other languages, ask a question/clarification through CMS.

Scoring

For each interaction, if the judge prints `FAIL`, or if your program fails via one of the other common errors (compilation failed, runtime error, time limit exceeded, etc.), your score will be 0. Otherwise, your score is computed as follows:

- For the ith test case, let d_i be the absolute difference between your guess for k and the actual value of k for that case, and let

$$s_i = \begin{cases} 1 & \text{if } d_i = 0, \\ 1/10 & \text{if } d_i = 1, \\ 1/20 & \text{if } d_i = 2, \\ 0 & \text{if } d_i > 2. \end{cases}$$

- Let s be the *average* of all s_is.
- Let a be the *average* number of queries you made across all test cases.
- Your score is computed as

$$50s + \min\left(50, \frac{440s}{a+6}\right)$$

  rounded down to 2 decimal places.

Thus, the maximum number of points that can ever be obtained from this problem is 100.

Sample Interaction

Note: The blank lines are only here to illustrate the chronology. The judge doesn't print blank lines. Your solution shouldn't print blank lines either.

Judge	Your Program
`1 100000 7`	
	`Q 2`
`Grumpy`	
	`Q 6`
`Happy`	
	`Q 9`
`Grumpy`	
	`A 2`
`OK`	

Explanation

This example interaction doesn't follow the constraints; it assumes that $t = 1$, $L = 7$, and that the names are `Happy`, `Doc`, `Grumpy`, `Dopey`, `Bashful`, `Sleepy`, `Sneezy`, in that order. (The real ordered list of names is in the attachment `noi.txt`.)

In the first (and only) test case, suppose the actual value of k is 1, and that the remaining alive characters are `Happy`, `Grumpy`, `Sleepy`, and `Sneezy`. Since the guessed value for k is 2, we have $d_1 = 1$, $s_1 = 1/10$ and $s = 1/10$. The average number of queries is $a = 3$. Therefore, the score is

$$50 \cdot (1/10) + \min\left(50, \frac{440 \cdot (1/10)}{3+6}\right) = 9.888888\ldots,$$

so the score rounded down to two decimal places is 9.88.

Local Testing Tools

An interactive testing tool is provided to aid local testing, downloadable in the CMS interface. Download the following files:

- `judge.py`. This program acts as the judge, as described in the Interaction section above. Note that you can also run this program on its own and interact with it by manually typing "`Q` i" (for an integer i) and "`A` k" (for an integer k).
- `interactive_runner.py`. This makes two programs interact with each other, connecting the output of one program to the input of the other, and vice versa.

Read the comments at the top of these files to learn more.

To run the judge program on its own, run the following in your terminal:

```
pypy3 judge.py noi.txt
```

where `noi.txt` is the file containing the character names.

To run the judge program and your solution program simultaneously and then make them interact, run the following in your terminal:

```
pypy3 interactive_runner.py  pypy3 judge.py noi.txt  --  [COMMAND]
```

where `pypy3` can be replaced by the command you use to invoke Python 3 in your terminal, and you replace "`[COMMAND]`" with the command that invokes your solution program.

The judge program also accepts arguments such as `-t` and `-q` helpful for debugging; run the following for more information:

```
pypy3 judge.py --help
```

If your solution would be marked as Wrong Answer, the judge program will output an appropriate message. At the end of the execution, it will output the values of s and a for this interaction.


NOIVENGERS