Problem A. Code Names

Input file:	`stdin`
Output file:	`stdout`
Time limit:	0.5 second
Memory limit:	64 megabytes

Imagine that you have to prepare a problem set for the forthcoming programming contest and you have already chosen the problems you will use in it. Before you start writing problem statements, preparing tests, and writing author solutions, you must give *code names* to all the problems.

A code name is a string that uniquely identifies the problem. For example, instead of saying "problem about the cipher grille," you can simply say "problem `grille`."

The problems in a problem set are arranged according to the lexicographical order of their code names. However, the program committee wants to get a fixed order of the problems. For example, the easiest problem should be put on the first place so that all the teams will be able to find it, or the letter `D` can be assigned to a really Difficult problem. Therefore, to obtain some predefined order of the problems in the problem set, the program committee needs to carefully choose the code names. This is just what you have to do.

To make your task easier, the program committee proposed three variants of the code name for each of the n problems in the problem set. You only have to choose an appropriate variant for each problem.

Input

The first line contains the number n of problems in the problem set ($1 \leqslant n \leqslant 16$). The i-th of the following n lines contains three possible code names for the i-th problem. The variants are separated with a space. The last line contains a permutation of the numbers from 1 to n. This is the order in which the problems must be arranged in the problem set. Each code name consists of lowercase Latin letters and its length is at most 20. All the code names are different.

Output

Output n lines. The i-th line should contain the code name of the problem that will have number i in the problem set. If there are several possible answers, output any of them. If it is impossible to choose the code names as required, output "`IMPOSSIBLE`".

Examples

stdin	stdout
11 cipher grille kamkohob names codenames codes newtests rejudge timus size volume summit watchmen braineater twosides solution random yesorno keywords subversion commands bosses shooting shaitan game strategy playgame mnemonic palindromes bestname eligibility rectangle rules 2 1 7 10 9 6 11 3 8 4 5	codenames grille keywords mnemonic playgame random rectangle rejudge shaitan volume watchmen
3 problems in the first sample are ordered not randomly 1 2 3	IMPOSSIBLE

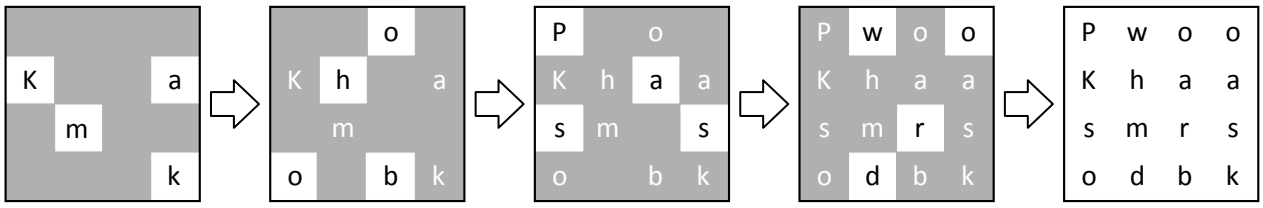
Problem B. Cipher Grille

Input file:	`stdin`
Output file:	`stdout`
Time limit:	0.5 second
Memory limit:	64 megabytes

Our program committee uses different tools for problem development: a mailing list, a version control system, an administration system of the *Timus Online Judge* website, and many others. Chairman of the program committee must constantly keep the passwords for these systems in his head. Of course, the passwords must be kept secret from the contestants, otherwise the problems may become known to them before the contest starts.

Not trusting his memory, the chairman wants to write down one of the passwords in a ciphered form. To do this, he plans to use a cipher grille he read about in one entertaining book.

A cipher grille is a 4×4 paper square in which four windows are cut out. Putting the grille on a paper sheet of the same size, the chairman writes down the first four symbols of his password in the windows (see fig. below). After that the chairman turns the grille clockwise by 90 degrees. The symbols written earlier become hidden under the grille and clean paper appears in the windows. He writes down the next four symbols of the password in the windows and again turns the grille by 90 degrees. Then he writes down the following four symbols and turns the grille once more. After that he writes down the last four symbols of the password. Now, without the same cipher grille, it is very difficult to restore the password from the resulting square with 16 symbols. Thus, the chairman is sure that no contestant will get access to the problems too early.



Assume that you obtained the grille used by the chairman and the resulting square with 16 symbols. Your task is to recover the chairman's password.

Input

The first four lines contain the chairman's cipher grille. The window in it is denoted by the symbol "`X`" and the paper is denoted by "`.`". The position of this grille corresponds to the position from which the chairman starts writing down his password. It is guaranteed that the grille is correct, which means that in the process of ciphering only empty cells appear in the windows. It is also known that the grille is connected, i.e. it is a single piece of paper.

The next four lines contain the square with the ciphered password. All the symbols in the square are lowercase or uppercase Latin letters.

Output

Output the password of the chairman of the program committee as a string consisting of 16 symbols.

Example

stdin	stdout
.... X..X .X.. ...X Pwoo Khaa smrs odbk	KamkohobPassword

Problem C. Key Substrings

Input file:	stdin
Output file:	stdout
Time limit:	2 seconds
Memory limit:	64 megabytes

Although the program committee works as one team, heated debates arise frequently enough. For example, there is no agreement upon which client of the version control system is more convenient to use: a graphic interface program or a console client.

Let us consider some command of a console client. A substring of this command that is not a substring of any other command of this client can be called a *key* substring because it uniquely identifies the command. In the latest versions of the client, it is not necessary to type the whole command; it is sufficient to type any of its key substrings.

A supporter of the console client wants to convince the program committee to use it. In order to show how fast and convenient the work with this client is, he wants to find a key substring of minimal length for each command. Help him do it.

Input

The first line contains the number n of commands in the console client ($2 \leqslant n \leqslant 1\,000$). Each of the following n lines contains one command of the client. Each command is a nonempty string consisting of lowercase Latin letters and its length is at most 100. No command is a substring of another command.

Output

Output n lines. The i-th line should contain any of the shortest key substrings of the i-th command (the commands are numbered in the order they are given in the input).

Example

stdin	stdout
3 abcm acm bcd	ab ac d

Problem D. Mnemonics and Palindromes 2

Input file:	stdin
Output file:	stdout
Time limit:	3 seconds
Memory limit:	64 megabytes

At last, Vasechkin had graduated from the university and it was the time to choose his future. Vasechkin recalled all the inadequate outcomes, unsolvable problems, and incomprehensible problem statements that he encountered at programming contests, so he decided to join a program committee. Soon he was asked to prepare a problem for the forthcoming student contest, which would be dedicated to binary alphabets. The problem had to fall under that topic. However, Vasechkin wanted the participants to remember his problem for a long time, so he decided to give the problem an unusual and complicated name.

Vasechkin decided that the name had to consist of the letters "a" and "b" only and contain exactly n letters. In addition, the name had to be as *complex* as possible. The *complexity* of a name is defined as the minimal number of palindromes into which it can be decomposed. Help Vasechkin to invent the most complex name for his problem.

Input

The only line contains an integer n ($1 \leqslant n \leqslant 1\,000$).

Output

Output the required name of length n consisting of the letters "a" and "b" only. If there are several such names, output any of them.

Example

stdin	stdout
6	aababb

The complexity of the string "aababb" is 3 because it can be decomposed into three palindromes (for example, "a", "aba", and "bb") and it cannot be decomposed into a smaller number of palindromes.

Problem E. Another Ball Killer

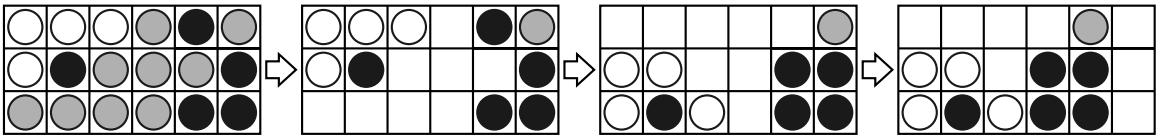
Input file:	`stdin`
Output file:	`stdout`
Time limit:	2 seconds
Memory limit:	64 megabytes

Contestants often wonder what jury members do during a contest. Someone thinks that they spend all the contest fixing bugs in tests. Others say that the jury members watch the contest excitedly and even bet on who will win. However, in reality, the jury members prefer to play computer games, giving a complete control of the contest to heartless machines.

"`Another Ball Killer`" is one of the favorite games of the jury. Its rules are very simple:

1. The game is played by one person on a rectangular field of size $n \times m$. At the initial moment, each cell of the field contains a ball of one of five colors: blue, green, red, white, or yellow.
2. At each move, the player chooses some *figure* (a connected group of two or more balls of the same color; balls are called connected if their cells have a common side) and removes it from the field. After that the balls that were above the removed balls fall down. If a column without the balls appears, then all the columns on its right are shifted to the left.

   The following image shows how the field changes after the removal of the largest figure.



   The player is awarded $k \times (k-1)$ points for his move, where k is the size of the removed figure, i.e. the number of balls in it.
3. The game is finished when there are no figures left on the field. The goal is to get as many points as possible by the end of the game.

Lazy jury members play "`Another Ball Killer`" using the following algorithm:

```
01  Choose the color of one of the balls in the field as the main color.
02  While there is at least one figure:
03      While there is at least one figure of a color different from the main color:
04          Remove the largest figure of a color different from the main color.
05      If there is a figure of the main color:
06          Remove the largest figure of the main color.
```

If there are several ways to remove the figure in lines 04 and 06, one should choose the largest figure containing the bottommost ball (if there are several such figures, then one should choose among them the figure that contains the leftmost of such balls).

Chairman is the laziest person in the jury. He doesn't even think about which color he should choose as the main one. By pressing one key, he launches a program that calculates for every color present in the field the number of points that will be awarded if this color is chosen as the main one. Your task is to write such a program.

Input

The first line contains the dimensions of the field n and m ($1 \leqslant n, m \leqslant 50$). Each of the following n lines contains m letters denoting the color of the ball in the corresponding cell of the field (`B` for blue, `G` for green, `R` for red, `W` for white, and `Y` for yellow). The rows of the playing field are given in the order from the top row to the bottom row.

Output

Output one line for each color present in the field: first output the letter denoting the color, then a colon, a space, and the number of points the chairman of the jury will get if he chooses this color as the main one. The colors must be considered in the following order: blue, green, red, white, yellow.

Example

stdin	stdout
3 6 WWWGBG WBGGGB GGGGBB	B: 74 G: 92 W: 74

Problem F. Alternative Solution

Input file:	stdin
Output file:	stdout
Time limit:	1 second
Memory limit:	64 megabytes

Valentine is a veteran of programming contests and he's been working in the program committee for many years. He is very busy this week: the bike is under repair, some problems with Indian colleagues have to be solved, and five student groups are to be examined in philosophical problems of mathematics at the university. To crown it all, the new chairman of the program committee asked Valentine to write an alternative solution for one of the problems of the forthcoming contest.

Valentine was so busy that he had no time to read the problem statement. He only glanced at the output format and understood that it was required to output either "YES", or "NO".

Fortunately, Valentine was well acquainted with the testing system used in the contest. The system successively runs a solution on all tests of a problem, and for each test the checking process goes as follows. The input is copied to the file input.txt. Then the solution is launched. It reads the input from the file input.txt and writes the result to the file output.txt. When it finishes, the correct answer is copied to the file answer.txt. If the contents of the files answer.txt and output.txt match, the test is assumed to be passed; otherwise, the test is not passed.

Valentine decided to write a program that would operate as follows. If the folder containing the program doesn't contain the file answer.txt (i.e. the program is run on the first test), then the program outputs "YES". Otherwise, the program outputs the contents of the file answer.txt.

Valentine plans to tell the chairman of the program committee that there is a nontrivial mistake in his program, and this mistake, fortunately, shows itself when the program is run on the excellent hard tests prepared by the author of the problem. However, first Valentine has to estimate the number of tests that his solution won't pass. Valentine doesn't have access to the tests, but he knows the number of tests and the total size of the files with answers. He also knows that the size of the file with the answer "YES" is 3 bytes, the size of the file with the answer "NO" is 2 bytes, and all the variants of the order of tests are equally probable. Help Valentine to calculate the average number of tests that his solution won't pass.

Input

The only line contains two integers n and s ($1 \leqslant n \leqslant 5\,000$; $2n \leqslant s \leqslant 3n$) which are the number of tests and the total size of the files with answers, respectively. The numbers are separated with a space.

Output

Output the average number of tests that Valentine's solution won't pass, accurate to 10^{-5}.

Example

stdin	stdout
3 7	2.0000000

One of the three answers is "YES" and two answers are "NO". If the order of tests is "YES-NO-NO", then Valentine's solution won't pass the second test only; if the order is "NO-YES-NO", then it will pass none of the tests; if the order is "NO-NO-YES", the solution won't pass the first and the third tests.

Problem G. Eligibility Rules

Input file:	stdin
Output file:	stdout
Time limit:	3 seconds
Memory limit:	64 megabytes

The program committee must be able to correctly estimate the level of teams that will participate in the forthcoming contest and choose the problems according to that level. It is not always easy to prepare a problem set that will please both school teams and the most experienced veterans of the programming contests, who won dozens of them.

In order to make his job easier, the chairman of the program committee asked the chairman of the jury to make the eligibility rules stricter and forbid too young or too experienced teams to participate in the contest.

For each team that had applied for participation, the jury calculated the average age of its members and the number of official contests this team participated in. In addition, the program committee estimated the satisfaction that each team would get from the problem set.

The jury decided to state the eligibility rules as follows: a team is eligible to participate if and only if its average age and the number of contests it participated in fall in certain ranges of values. It only remained to set the boundaries of these ranges so that the sum of satisfactions of the eligible teams would be as large as possible. Of course, there had to be at least two eligible teams, otherwise there would have been no sense in holding the contest.

Input

The first line contains the number n of teams that want to participate in the contest ($2 \leqslant n \leqslant 1\,500$). Each of the following n lines contains three space-separated integers a, f, and s ($4 \cdot 10^8 \leqslant a \leqslant 10^9$; $0 \leqslant f, |s| \leqslant 10^9$). They are the average age of participants (in seconds), the number of official contests the team participated in, and the satisfaction that the team will get from the problem set, respectively.

Output

In the first line output the minimal and maximal admissible average age of participants. In the second line output the minimal and maximal admissible number of contests. All these numbers must be nonnegative integers and must not exceed 10^9. If there are several possible answers, output any of them.

Example

stdin	stdout
4 500000000 1 1 510000000 3 -10 600000000 3 2 700000000 4 3	550000000 750000000 2 5

Problem H. Rejudge

Input file:	stdin
Output file:	stdout
Time limit:	0.5 second
Memory limit:	64 megabytes

At three o'clock in the morning Sasha (aka *Sandro*) discovered that there were only seven tests for one of the problems in the first volume of the *Timus Online Judge*. "That won't do," Sasha thought and added ten new tests for that problem. At four o'clock in the morning, Vova (aka *Vladimir Yakovlev*) discovered that the sixth test in the same problem was incorrect. "That won't do," Vova thought and deleted the sixth test.

The next day Sasha and Vova came to work at two o'clock in the afternoon. After some discussion they decided to rejudge all the solutions for that problem that had been submitted before the moment Sasha added new tests.

After the rejudge each author receives an e-mail with the list of all her solutions for which the outcome has changed. If that list is empty the e-mail is not sent. Help Sasha and Vova determine the minimal possible and maximal possible number of authors who will receive an e-mail with the rejudge results.

Input

The first line contains the number of solutions n that Sasha and Vova want to rejudge ($1 \leqslant n \leqslant 1\,000$). Each of the following n lines describes one solution and contains the name of its author and the previous outcome.

The name of the author is a nonempty string of length not exceeding 30. It may contain Latin letters, digits, the underscore character, parentheses and square brackets. It is guaranteed that no two different authors have names which differ only in case of the letters.

The possible outcomes are: `AC`, `CE`, `ML X`, `TL X`, and `WA X`, where `X` is an integer from 1 to 7. The outcome `AC` means that the solution passed all the tests. The outcome `CE` means that the solution failed to compile and, therefore, was not launched on any of the tests. The outcomes `ML X`, `TL X`, and `WA X` mean that the solution passed the tests with numbers less than `X` but failed to pass the test `X` because it exceeded the memory limit or the time limit or because it produced a wrong answer. The outcomes which differ only in test number are considered different.

Output

Output the minimal possible and maximal possible number of authors who will receive e-mails with the rejudge results.

Example

stdin	stdout
5 [SPbSU_ITMO]_WiNGeR TL 6 Milanin_(TNU) WA 6 Vladimir_Yakovlev_(USU) AC Sandro_(USU) WA 1 Sandro_(USU) ML 3	0 3

Problem I. Kill the Shaitan-Boss

Input file:	`stdin`
Output file:	`stdout`
Time limit:	0.5 second
Memory limit:	64 megabytes

At the meeting where Maximilian and other members of the program committee were discussing problems for the forthcoming subregional contest, it turned out that the problem set lacked a geometrical problem. The chairman of the program committee asked Maximilian to invent such a problem before September 1.

On the night from August 30 to August 31, Maximilian was sitting in front of his computer trying to pass the final level of the "Kill the Shaitan-Boss" game. Maximilian was attacked by a horde of shaitan-bosses, but he was armed with a shaitan-tube and managed to shoot down almost all the bosses. Only four bosses were left. Maximilian froze them with a spell and was getting ready to kill them when he noticed that only three charges left in his shaitan-tube. "It doesn't matter," Maximilian thought. "I will kill two shaitan-bosses by two shots and after that I will reach the straight line passing through the two remaining bosses and kill them both with one shot. The main thing is not to miss." We must note here that the shaitan-tube shoots from both its ends at once and kills all the shaitan-bosses on the shooting line.

Maximilian aimed at the first shaitan-boss and was ready to press the "fire" button, when suddenly his mobile phone rang. Frightened Maximilian shot in the air, then cursed himself and the person who called him. After that he answered the call.

"Hello, how are you doing? Have you invented a geometrical problem?"

Of course, it was the chairman of the program committee.

"Oh yes, it's almost ready, there's very little left to do. I'll send you the statement tomorrow night."

"OK, remember that we have a lot to do, we'll have no time to think later on."

"Ah," Maximilian thought switching off. "Such a game lost! Almost all shaitan-bosses have been killed..." But there was still a chance to kill the remaining four bosses with two shots. For that he just needed to carefully calculate his movement and shooting directions. "After I finish the game, I will start thinking about the problem," Maximilian decided. Of course, he didn't even start inventing the problem. "I only need to shoot down these shaitan-bosses as soon as possible..."

Input

Assume that Maximilian and the shaitan-bosses are points on a plane. Let us introduce a coordinate system and denote the point where Maximilian stands as its origin. Then the i-th shaitan-boss will have coordinates (x_i, y_i). No three shaitan-bosses stand in the same straight line. Maximilian can stand in the same point with the shaitan-boss and he will kill this boss if he shoots from this point.

The input contains four lines, each containing two space-separated integers x_i and y_i ($|x_i|, |y_i| \leqslant 10\,000$).

Output

Output the minimal distance Maximilian must walk in order to kill all the shaitan-bosses, accurate to 10^{-5}.

Example

stdin	stdout
-2 0 2 0 -2 4 2 4	2.0000000000

Problem J. Summit Online Judge

Input file:	`stdin`
Output file:	`stdout`
Time limit:	0.5 second
Memory limit:	64 megabytes

The program committee of Yekaterinozavodsk programming contests decided to create a new website. In honor of the recent summit of the Harbin Cooperation Organization, they decided to name this website *Summit Online Judge*.

The website will contain a problem archive, which will be regularly extended with problems from contests held at this website. According to the current rules, the problem set of any contest must contain at least x and at most y problems.

The contests at the *Summit Online Judge* will be held often enough, which means that the number of problems in the archive will increase fast. For convenience, it was decided to divide the archive into volumes of equal size. The size of the volume is called *correct* if there is at least one way to hold several contests so that the total number of problems in these contests would be equal to the size of the volume. In addition, the size must be in the range from l to r.

Your task is to find the number of ways to choose a correct size of the volume.

Input

The only line contains four space-separated integers x, y, l, r ($1 \leqslant x, y, l, r \leqslant 10^{18}$; $x \leqslant y$; $l \leqslant r$).

Output

Output the number of ways to choose a correct size of the volume.

Example

stdin	stdout
4 5 7 13	5

In the example above, the size of the volume can be equal to 8, 9, 10, 12, or 13.

Problem K. Two sides of the same coin

Input file:	`stdin`
Output file:	`stdout`
Time limit:	1 second
Memory limit:	64 megabytes

Side one

Poll flew above the City. Bright street lights flashed by below. He was winding his way around needle-sharp spires of communication towers and wicker grid radiators of thermonuclear power plants. Express mail helicopters rushed past him. Suddenly, neat street layout broke down. Streets, which used to be straight and parallel, started to twist and twirl like a bunch of glowing tentacles. Poll soared upwards. As seen from above, it wasn't streets anymore, it was a neural ensemble, alive and pulsating with energy. One of the neurons flashed and faded to black. Impulses, generated by the flash, reached its neighbours and lit them too. The wave of flashes was propagating through the neural tissue, consuming and annihilating it. Soon, Poll found himself alone in the dark. The darkness surrounded him, filled his eye sockets and mouth, squeezed his chest. Poll couldn't breath, couldn't think and the only thing he felt was absolute, total terror...

Poll woke up screaming. His dreams became even worse that week. Vivid, detailed, indistinguishable from reality, but at the same time incoherent and perplexed. Actually, it was a well-known symptom, one of those that were featured prominently in the sermons of street preachers from doomsday cults. Poll tried to deny it all, but his condition became too obvious to ignore. He typed «neurointerface autoinduction syndrome» in his favourite search engine and in a short time found a specialized clinic nearby. He registered on its site and booked an appointment that evening.

Poll went to work on foot. Private vehicles were forbidden in the City, and he didn't want to suffocate in overcrowded carriages. But the streets were far from deserted either. The buzzing of the crowd was complemented by the screaming ads running on huge screens which were profusely hanged above the pavement. Poll noticed painfully familiar logo on one of the screens, and a pleasant female voice informed him that «Subregional Programming Contest will be held in the City in less than a week. Make sure you're registered!»

Poll had entered a high-rise building and went up to floor 70. The floor housed the jury of the contest from the advertisement, and Poll was the chairman of the jury. Several people were sitting behind their desktops in a big room. Most of them were using neurointerfaces, so no clicking of keyboards could be heard. Those people were Poll's hand-picked team, they were preparing the problems for the contest. All of them were professional, licensed problemsetters with the rank no less than 3. Nevertheless Poll always verified and approved every problem by himself before a contest. He trusted his team, but the stakes were high and a second pair of eyes couldn't hurt.

Poll went to his office, connected to the internal network via his neurointerface and plunged into the work. The day passed by. Poll noticed that it got dark outside only when headlights of a fire helicopter flying by his window illuminated the room. He looked at his watch, and realized that he was late for the doctor's appointment. He logged off and hurried to the clinic.

A nice nurse greeted Poll at the reception and directed him to the doctor Wasat's office. Dr. Wasat turned out to be a friendly, sympathetic aged medic, who could easily gain the confidence of his patients. He listened carefully and immediately went to the heart of the problem. After some consideration, he suggested Poll to undergo brain scanning right then, to confirm or deny any issues with the neurointerface. Poll felt relieved that his case had been in hands of a specialist and he agreed to the procedure. They went to the neighbouring room, with a sophisticated device resembling an electric chair in it. Poll sat down on it. The doctor fixed his head between two hemispheres and stepped away to control panel. Poll saw nothing but a plain white wall in front of him. He sensed that something connected to his neurointerface.

“Don’t resist it,” said the doctor gently.

Poll relaxed. The device hummed to life, sending warm waves through Poll’s body. Nothing happened for a while, but then suddenly Poll noticed thin sharp spikes extending from the white wall. They were slowly growing, coming closer and closer to him. All of them were aimed at his heart. He tried to break free, to dodge, but to no avail. Thousands of white spikes penetrated his chest, pierced his heart and started to pump bright white light into him. The light displaced blood in his arteries, reached all corners of his body, flooded his mouth, poured out of his eyes. Poll shut his eyes tight and violently writhed.

“Poll! Poll! Calm down! It’s OK, it’s over now!” the doctor’s voice sounded concerned.

Poll opened his eyes. Doctor Wasat was holding his shoulders and watched him closely. There were neither spikes nor the light around.

A nurse took Poll back to the office and brought him hot tea. He was sitting there alone for a long time, until doctor Wasat finally returned. The doctor was gloomy. He sat behind his desk and categorically stated:

“I don’t want to scare you, but your condition is very serious indeed. I’m sure you understand it too. It’ll get worse. You’ll experience visions not only when you’re sleeping but when you’re awake too. To be precise, you won’t be able to distinguish dreams and reality.”

They were talking for a couple of hours. The doctor promised to help in any way possible, but he couldn’t guarantee anything. Modern medical science still wasn’t able to treat such conditions.

Poll left the clinic very depressed. The doctor suggested to lift him home in a medical helicopter, but Poll declined. He didn’t want to go home. He went straight to the local council building, which was the tallest in the neighborhood. Although its roof was off-limits to the public, Poll happened to know the access code. He climbed the roof, walked up to the edge and breathed the cool night air. He could see the city up to the horizon. Not the twisted city from his dreams, but the usual bustling city of real people. The Reality. With all its flaws, Poll didn’t want to get detached from it. Better be dead than end up in a straitjacket inside an asylum.

“Gonna jump?” A female voice from behind caught Poll by surprise.

Poll turned back. A girl, mid-20’s, was sitting on an air duct. She grinned and looked down at him.

“Don’t know,” said Poll with a smile.

The girl jumped off the duct and walked up to Poll. She stepped on the edge near him and looked down in awe.

“So high... I’m Alhena. You?”

“Poll.”

They were standing shoulder to shoulder for a few minutes. Poll started to feel nervous. Alhena broke the silence first:

“So, Poll. Gonna invite me for a date? But not tonight,” she added hastily.

“All right, how about tomorrow?” Poll replied captivatedly.

“That’d be great.”

They exchanged their numbers and went down to the street together. Poll tried to follow her, but soon lost her in the crowd.

Next morning Poll was late for work. When he finally arrived, he found his old friend Castor waiting for him in the office. They had been arranging contests together for a long time. But Castor wasn’t a problemsetter. From young age he was much into protection of everything. He was responsible for physical and information security of the contests. People called him «Castor-hound» behind his back, but Poll liked

him. Maybe because Castor helped Poll to deal with several delicate situations and difficult people, earning his deep trust and respect.

From time to time, Castor came to each employee to personally update the access codes through the neurointerface. They called it «Castor eats our brains.» And when the codes weren't updated for some time, someone always asked «When will Castor eat our brains again?»

Today was the day. Castor connected to Poll's neurointerface and while the codes were being uploaded, they chatted randomly. When Castor left, Poll was feeling much better. A chat with a friend calmed him down and he was ready to get into gear.

He worked hard all day. In the evening he called Alhena and invited her for dinner. They went to the best restaurant in the neighborhood. Alhena turned out to be very sociable, able to find common ground even with such an extreme introvert as Poll. They talked about a range of things, and Poll felt he really got to know her. The only thing that troubled him was that he couldn't figure out what she was doing on the roof, and he didn't know how to ask.

After the restaurant they went to her place. She lived alone in a small, tidy apartment. Poll settled on a couch while Alhena poured out two glasses of wine. They toasted the evening and sipped a little. She leaned forward to kiss him...

Side two

She leaned forward to kiss him, when suddenly her head jerked up and she gave a shrill scream. Her whole body was shaking with convulsions and she tumbled down to the floor.

In a moment, mayhem broke out in the apartment. The main door was pulled out, windows were broken, and a dozen of armed men wearing masks and body armor poured inside. Four of them grabbed Alhena and dragged her away. Two were holding Poll down tight. But Poll didn't even try to struggle, he was dumbfounded by the unexpected twist of the story.

A man in the same body armor but without a mask entered the room. Poll, to his utter surprise, recognized his buddy Castor. Castor sat down opposite Poll and grinned. By then, the initial shock was wearing off, and Poll burst out on him:

"Castor, what the hell? What's going on here?"

"Me saving your ass again is going on here," said Castor favorably and in a commanding voice ordered, "Release him."

The two let Poll off and stepped out. Poll rubbed his left shoulder and asked, frowning:

"What's up with Alhena?"

"She was smashed by the virus I planted into your head today."

"I don't understand."

"It's really easy," said Castor, making himself comfortable. "As a matter of fact, you're not ill. They somehow managed to hack your brain and infect it with a virus which made you believe you've got a syndrome. Besides, the virus allowed them to make a copy of your memory via the neurointerface without you even noticing it. She just tried to do this very thing. She tried to download your today's memories, memories of all the problems you worked on. But she didn't know that I erased their virus from your system and planted mine. I call it «Castor-hound»," and he winked at Poll. "Whenever someone tries to connect to the backdoor in your head, Castor-hound starts to eat his brain. You saw the result."

Poll started to connect the dots.

"And doctor Wasat..."

"...is not a doctor at all. You were his first and last patient. He downloaded a good chunk of information from your head, all the work you did this month. We captured him eventually, but it was too late, he

already sold the data. There is no way we can stop the leak now. I'm afraid the contest should be canceled."

Poll was fuming. He was tricked like a big silly. He stared at Castor and said categorically:

"No, we won't cancel the contest. We'll prepare new problems."

"Do you have enough time? How many problems can you prepare simultaneously?"

That was a good question. The contest rules state that every problem should be prepared by two persons exactly: one should write a statement, the other one should prepare a set of tests. Generally, professional problemsetters have license for only one of the tasks, but some of them can do both. Besides, to eliminate a possibility of conflicts between the problemsetters working on the same problem, the difference of their ranks should be equal to 2.

They had not much time left, so each of them could prepare only one problem. Poll urgently needed to answer the question: how many problems would the jury be able to prepare for the contest?

Input

The first line contains the number n of problemsetters in the program committee ($2 \leqslant n \leqslant 1\,000$). Each of the following n lines contains a problemsetter's name, his specialization and his rank separated with spaces. The name consists of up to 20 Latin letters. There are no two problemsetters with the same name. Specialization is denoted by one of the three words: "`statements`", "`testdata`", or "`anything`", corresponding to the people able to write statements, people able to prepare the tests and to the people able to do both. The rank is an integer in range from 3 to 1 000.

Output

In the first line output the maximal number of problems that can be prepared for the contest. For each of these problems output a line containing the name of a person writing the statement, a space, and the name of a person preparing the tests. If there are several ways to prepare such an amount of problems output any one.

Example

stdin	stdout
7 Poll anything 8 Tejat statements 6 Mebsuta testdata 6 Propus testdata 4 Alzir anything 7 Mekbuda anything 3 Kunot testdata 9	3 Poll Mebsuta Tejat Propus Alzir Kunot