2020 ECNU Campus Online Invitational Contest

China, Shanghai

May 23, 2020

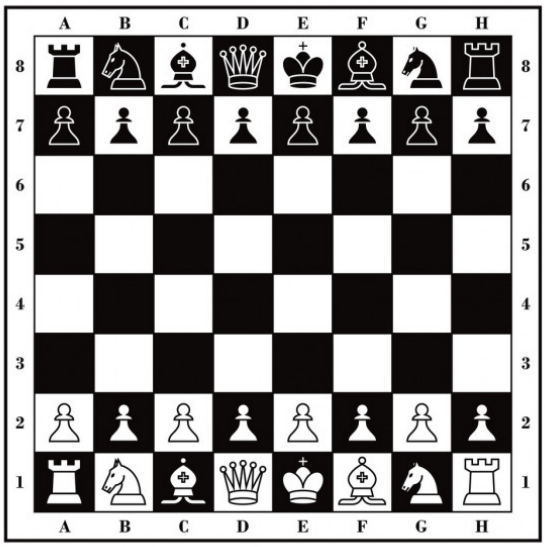
Important Notes:

- If you want to get prize from the contest sponsor, please complete the required information (real name, school and e-mail) in EOJ personal settings.
- The problemset consists of 9 problems in 19 pages (including this page).
- The statements contain no time and memory limits. Please visit the Problems page on the EOJ to view them.

Problem A. Amateur Chess Players

Input file: `standard input`
Output file: `standard output`

Chess is a two-player board game played on a chessboard (a square-checkered board with 64 squares arranged in an eight-by-eight grid). In a chess game, each player begins with sixteen pieces: one king, one queen, two rooks, two knights, two bishops, and eight pawns. The object of the game is to checkmate the opponent's king, whereby the king is under immediate attack (in "check") and there is no way to remove or defend it from attack, or force the opposing player to forfeit.



Cuber QQ and Quber CC are two amateur chess players, who know almost nothing about all the fancy rules in chess, perhaps except how the chessboard looks like, and they have no interest in it. Instead, they invent their own chess game. At the beginning, Cuber QQ, who has the white pieces, and Quber CC, who has the black pieces, place some of their pieces on the chessboard. Then they start to remove those pieces by turn. Cuber QQ moves first. In each turn, they must remove at least one of their own pieces (Cuber QQ can only remove white and Quber CC can only remove black). Two or more pieces can be removed together in one turn if and only if these pieces are collinear on the chessboard, meaning they should lie in the same line. Note that this line does NOT have to be in horizontal or vertical or diagnoal direction. The one who fails to make a move loses the game.

Now Cuber QQ and Quber CC are both desperate to win the game. So they will do it smartly and make optimal decisions. Who do you think will win the game, eventually?

Input

The input consists of four lines:

- The first line is an integer n ($1 \le n \le 16$), the number of white pieces on the chessboard.
- The second line consists of n space-separated positions, which are the positions of white pieces.
- The third line is an integer m ($1 \le m \le 16$), the number of black pieces on the chessboard.
- The last line consists of m space-separated positions, which are the positions of black pieces.

Each position is a upper case letter in "A" to "H", followed by a digit in "1" to "8".

It is guaranteed that there are no overlapping pieces, that is, all pieces are located at different positions.

Output

If Cuber QQ is going to win, output "`Cuber QQ`" without quotes. Otherwise output "`Quber CC`".

Examples

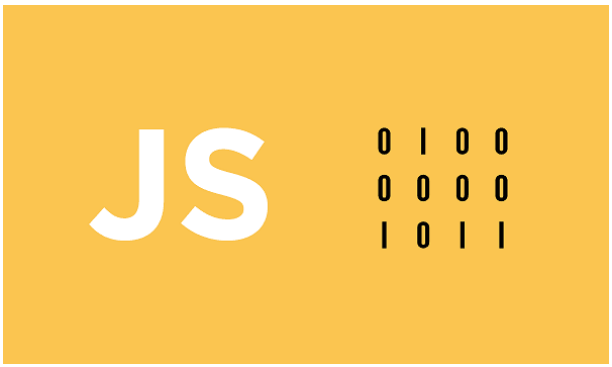
standard input	standard output
4 A1 B2 D4 C3 3 A8 D6 H7	Cuber QQ
4 A1 B2 C3 D5 4 A8 C7 E6 G5	Quber CC

Problem B. Binary String

Input file:	`standard input`
Output file:	`standard output`

This is an interactive problem.

Cuber QQ, who was the master of subsequence, has recently come up with a new string game which immediately received much attention, well, at least your attention.



The rules of this game are straightforward. Cuber QQ will generate a binary string of length n, i.e., a string consisting of n characters '`0`' and '`1`'. He will tell you the length of binary string he has generated. After that, he gives you some opportunities to ask him a binary string whose length is at most $\lfloor \frac{n}{2} \rfloor + 1$, he will answer if your binary string is a subsequence (not necessarily continuous) of his binary string.

You can ask at most 1023 questions if your binary string is a subsequence of Cuber QQ's binary string, and after that, you will get one and only one chance to make your guess of Cuber QQ's binary string. Unless you answer is correct, you will lose the game.

A non-empty sequence a is a subsequence of a sequence b if a can be obtained from b by deletion of several (possibly zero) elements.

Interaction Protocol

In the first line, you are given one positive integer n ($1 \le n \le 1000$) — the length of the binary string generated by Cuber QQ.

To ask a question, print "`?` s" , s ($1 \le |s| \le \lfloor \frac{n}{2} \rfloor + 1$) is a binary string that you are interested whether it is a subsequence of the binary string generated by Cuber QQ.

The interactor will respond with a single integer:

- `0` — the binary string you gave is not a subsequence of the binary string generated by Cuber QQ.
- `1` — the binary string you gave is a subsequence of the binary string generated by Cuber QQ.

When you are ready to answer, print "`!` s", s is the binary string that you guess Cuber QQ generated. Note that the interactor doesn't print anything in response to you printed answer, so you have to terminate the program instantly.

You can ask at most 1024 questions (including the last one) per test case.

After printing every query do not forget to put an end to the line and flush the output. Otherwise, you will get `Idleness limit exceeded`. To do this, use:

- `fflush(stdout)` or `cout.flush()` in C++;
- `System.out.flush()` in Java;

- `flush(output)` in Pascal;
- `stdout.flush()` in Python;
- see documentation for other languages.

If the interactor responds with "`-1`" instead of "`0`" or "`1`", it means that you've made an invalid query. Exit immediately after receiving "`-1`" and you will see `Wrong answer` verdict. Otherwise you can get an arbitrary verdict because your solution tries to read from a closed stream.

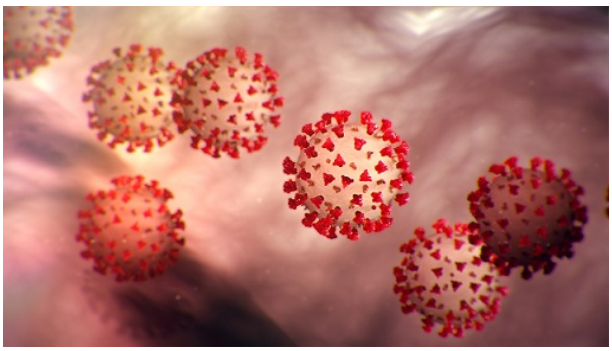
Example

standard input	standard output
2 1 1 0 1 0 0 0	 ? 0 ? 1 ? 00 ? 01 ? 10 ? 11 ! 01

Problem C. Coronavirus Battle

Input file: `standard input`
Output file: `standard output`

As COVID-19 is rampantly infecting more and more people, casting a devastating impact on human-world, Cuber QQ, as a super-man, is busy travelling around the world saving lives, and meanwhile putting his own health under threat.



However, as a super-man, Cuber QQ has a different immune system from a normal human being. For simplicity, Cuber QQ's immune system can be modeled as thousands of leukocytes, sophisticatedly arranged into a phalanx, in a three dimensional space. Each of them has a distinct location (x, y, z) where x, y and z are positive integers. When the virus initiates one attack, it always comes from negative-x, negative-y and negative-z direction. A leukocyte can survive this attack, if and only if it is protected by another cell whose location is more exposed to virus, i.e., a leukocyte with location (x', y', z') where $x' \leq x, y' \leq y$ and $z' \leq z$, and at least one of $x' < x, y' < y, z' < z$ is satisfied. If a leukocyte is not protected during this round of attack, it will die immediately and cannot protect others any more in future. Otherwise, it will survive.

Sophisticated as the system is, with the virus attacking over and over again, eventually all the leukocytes will die and Cuber QQ's immune system will crash, after which he shall be very sick. Nevertheless, Cuber QQ is a super-man who is willing to push his limits and fight as he can. He wants to know how much time he has got left. And you, as the best programmer in Cuber-ECNU-Joint-Force-Facility, are just given 4 hours to figure out how many rounds of attacks our world's hero will be put through before he falls down.

Input

The first and only line of the input contains three space-separated integers n ($1 \leq n \leq 10^5$), k_1 and k_2 ($10^8 \leq k_1, k_2 \leq 10^{12}$).

The following code describes the algorithm Cuber QQ's immune system used to arranage the cells. It is written in C++, and surely you can convert it to any other language you would like to use. With this algorithm, $(x_1, y_1, z_1), \ldots, (x_n, y_n, z_n)$ ($0 \leq x_i, y_i, z_i < 2^{64}$) can be generated.

```
unsigned long long k1, k2;
unsigned long long CoronavirusBeats() {
    unsigned long long k3 = k1, k4 = k2;
    k1 = k4;
    k3 ^= k3 << 23;
    k2 = k3 ^ k4 ^ (k3 >> 17) ^ (k4 >> 26);
    return k2 + k4;
}

for (int i = 1; i <= n; ++i) {
    x[i] = CoronavirusBeats();
    y[i] = CoronavirusBeats();
```

```
    z[i] = CoronavirusBeats();
}
```

Except for the examples, k_1 and k_2 are randomly chosen. And it is guaranteed that all locations generated will be distinct.

Output

Output the number of attacks t before Cuber QQ's immune system is destroyed.

To increase the credibility of your answer, in the next line, you also need to print n integers $a_1, a_2, \ldots, a_n$ ($0 \le a_i < t$). a_i is the number of attacks the i-th leukocytes will survive.

Examples

standard input	standard output
5 998244353 1000000007	2 0 1 1 1 1
5 100000000 1000000000000	1 0 0 0 0 0
5 1000000000000 100000000	2 1 0 0 0 1

Note

For the first example, the locations are:

8373945454928271 8388672858446274 535243480518893803
13645619435845943757 2297581721369636385 2065542152320352085
15149090731305068611 13847141931455896476 940493480722232539
17454784661911327411 15090004400591888349 684714220550566680
11704920635736733272 11725064299927455615 11820794347659711998

After the first one died, the rest can no longer protect each other, and won't survive the second attack.

Problem D. Decay of Signals

Input file: `standard input`
Output file: `standard output`

When ECNU was first built, there were n buildings, but only $n-1$ roads connecting them, due to limited budgets. I'm sure those who are proficient in graph theory will figure out that there would be exactly one path between any two buildings. Yes, that's true. And back then, ECNUers did not have so much population and so much traffic going around like nowadays, so they didn't think that one day it would be a problem.

At least not until, Cuber QQ, who was an old professor studying communication and signal processing, raised his concern and decided to deal with it by building a communication system in the campus, although some archaeologists believe that's not true — the main reason why he did this in the first place was he would rather stay in the lab with his cat rather than exhaustively ride a bicycle around the campus only to send a short message.

In particular, he built n signal amplification stations, with amplification ratio $a_1, a_2, \ldots, a_n$, one in each building, and buried $n-1$ cables, one under each road, each connecting two amplifiers. When a signal was sent from one building u to another building v, it would go in the shorted path. The initial strength of signal was 1, then multiplied by the amplification ratio a_u, decayed along the cables and amplified by intermediate stations, until passing the amplifier at v at the receiving end.

Formally, the strength of a signal sent by u, and received by v is,

$$\frac{a_{p_1} a_{p_2} \cdots a_{p_m}}{m}$$

where $p_1, p_2, \ldots, p_m$ is the shorted path from u to v. m is at least 1, and $p_1 = u$, $p_m = v$.

After many years, Quber CC, who is Cuber QQ is grand-grand-child, is officially a freshman at ECNU. When he looked into this faded history, he couldn't help wondering whether the system works at all. He is especially interested in finding out how bad the system can be, i.e., he wants to find two buildings u and v, such that the signal sent from u to v would have the smallest strength. To this end, he immersed himself in the library and collected all the information needed.

TL;DR, given an undirected tree with values on node, find the path with minimal "product of values divided by the length of path".

Input

The first line contains an integer n ($1 \le n \le 10^6$) — the number of buildings in the campus.

Each of the following $n-1$ lines contains two integers u_i and v_i ($1 \le u_i, v_i \le n$, $u_i \ne v_i$) — the i-th road in the campus.

The i-th of the following n lines contains one integer a_i ($0 \le a_i \le 10^9$) — the amplification ratio of the amplifier built in the i-th building.

Output

The only line of output is the minimal answer in the form of a completely reduced fraction "x/y", where x and y are relatively prime integers.

Examples

standard input	standard output
2 1 2 1 2	1/1
3 1 2 1 3 2 2 2	2/1
3 1 2 1 3 0 1 2	0/1

Problem E. Even Degree

Input file: `standard input`
Output file: `standard output`

As a expert in graph visualization, Cuber QQ has implemented a graph editing tool that supports adding/deleting nodes, adding/deleting edges and instant graph layout. The following image is a demo of the tool he has built.

These days, he noticed that there was a bug in his system. He cannot delete an edge (u, v) when both of u and v are odd-degree nodes, i.e., he can only delete the edge when at least one of (u, v) is connected to k edges, where k is any even number.

Now, instead of fixing this bug, Cuber QQ wants to play a little game. He tries to delete as many edges as possible, without using extra operations like adding edges or adding nodes or deleting nodes. He thinks such challenge is a piece of cake to him, so he wants to test you and see if you can solve this problem.

Input

The first line contains two integers n and m ($1 \le n \le 5 \cdot 10^5$, $0 \le m \le 5 \cdot 10^5$) — the number of vertices and edges in the graph.

The i-th of the next m lines contains two space-separated integers u_i and v_i ($1 \le u_i, v_i \le n$) — the i-th edges in the graph.

It is guaranteed that the vertices in the initial graph all have even degrees, and there are no self-loops or multiple edges in the given graph.

Output

In the first line, output a single integer k — how many edges at most you can delete.

In the second line, output k distinct integers $e_1, e_2, \ldots, e_k$ ($1 \le e_i \le m$) in order of deletion, where e_i is the index of i-th edge to delete.

If there are multiple answers, print any of them.

Example

standard input	standard output
3 3 1 2 1 3 2 3	2 1 3

This page is intentionally left blank

Problem F. Find / -type f -or -type d

Input file: `standard input`
Output file: `standard output`

Cuber QQ wants to know the number of files with extension `.eoj` in his computer. This can be easily done with a command `find / -type f | grep '\.eoj$' | wc -l`. However, Cuber QQ is a grep-hater, who would rather write his own program than using grep. So he decides to take a detour: what if the command starts with something else? Is it still possible to recover the results?

If you are not familiar with command usages in Linux, all you need to know is that `ls` and `find` are two easy-to-use commands to inspect files and subdirectories a directory contains. For example, when you are trying to get a list of everything in your computer, you might try to use: `find / -type f -or -type d`, which will give you a list like:

```
cuberQQ@ECNU:~$ find / -type f -or -type d
/i
/i/am
/i/am/a
/i/am/a/genious
/i/am/an
/i/am/an/ecnu
/i/am/an/ecnu/student
/i/am/an/idiot
```

To make the problem even more interesting, Cuber QQ adds another `shuf` after `find`, so that the list is shuffled into a random order and his secrets will stay covered. Cuber QQ is wondering whether it's possible to write a program `cuber-qq-grep` that filters out all the files with extension `.eoj` from the given shuffled list, which is his initial intention. Still, instead of giving the filtered list directly, Cuber QQ wants to know the length of this list, i.e., the number of files found. In other words, the following two commands will be almost equivalent:

- `find / -type f | grep '\.eoj$' | wc -l`
- `find / -type f -or -type d | shuf | cuber-qq-grep`

Well, there can be some subtle differences in input/output formats, but that's not essential.

One more thing, on your file system, directory is only a logical concept. This means, a directory is created only when there is a file which relies on this directory is created and a directory cannot exist without files.

TL;DR, given the randomly shuffled list of all directories and files on a computer, count the number of files that ends with `.eoj`.

Input

The input starts with a line of one number n ($1 \le n \le 10^5$), which is the length of the following list.

In the following n lines, each line contains one string, which is an absolute path to a file or a directory. The path starts with `/`, and is composed of multiple tokens (file names and directory names) concatenated with `/`. The tokens always start with a lowercase letter, followed by no more than 9 lowercase letters or dots. The root folder alone will not be included in this list.

It is guaranteed that the total length of n lines will be no longer than 10^6.

Output

Output the number of files satisfying the above-mentioned condition, in one line.

Examples

standard input	standard output
3 /secret/eoj /secret /secret.eoj	1
8 /i/am/an/ecnu/student /i/am/an/ecnu /i /i/am/a /i/am/an/idiot /i/am/an /i/am/a/genious /i/am	0
2 /cuber.eoj/qq.eoj /cuber.eoj	1

Problem G. Geralt of Rivia

Input file: `standard input`
Output file: `standard output`

The master witcher Geralt of Rivia is going to hunt Cuber QQ, who is a notorious monster in Jungle-Brain. Geralt's attack value is denoted by an integer a_g, and his defense value is denoted by an integer d_g. Similarily, Cuber QQ's attack value and defense values are denoted by two integer a_c, d_c. Meanwhile, the initial Health Point (HP) of Cuber QQ is an integer n, and Geralt has infinite amount of HP, because he can drink potions to recover from damage.

The damage of an attack will be reduced by the receiver's defense value, i.e., if the attack value is a and the opponent's defense value is d, the actual damage caused is $\max(0, a-d)$. The combat will go as follows. Geralt and Cuber QQ take turns to attack. Assuming currently Cuber QQ's HP is hp. Geralt will attack first, and Cuber QQ will receive a damage of $\max(0, a_g - d_c)$. and its HP would be $hp - \max(0, a_g - d_c)$. Once Cuber QQ's HP reaches zero or less, he will be dead and Geralt wins the combat, otherwise Cuber QQ will fight back, causing damage $\max(0, a_c - d_g)$.

Geralt can pay crowns (some kind of money used in Jungle-Brain, to upgrade his attack value and defense value. He can pay a crowns to increase his attack value by 1, and increase his defense value by 1 at the cost of b crowns. However, he only has m crowns so the total number of crowns he pays must be no more than m. **Note that he can add any rational number of attack and defense**, e.g., he can pay $\frac{3a}{5}$ crowns to increase $\frac{3}{5}$ attack.

Geralt wants to minimize the damage he receives. Please tell him the minimal damage he shall receive if he plans optimally to upgrade. If Geralt can't defeat the monster, output -1 instead.

Input

The first line of the input contains one integer t ($1 \le t \le 10^4$), denoting the number of test cases. Then t test cases follow.

Each case contains two lines.

The first line contains four space-separated integers a_g, d_g, a_c, d_c ($1 \le a_g, d_g, a_c, d_c \le 10^4$), denoting the attack value and the defense value of Geralt and Cuber QQ respectively.

The second line contains another four space-separated integers n, m, a, b ($1 \le n, m, a, b \le 10^4$) denoting Cuber QQ's initial HP, the amount of crowns Geralt has, the unit cost of attack upgrade and the unit cost of defense upgrade.

Output

For each case, output the answer in the form of a completely reduced fraction "x/y", where x and y are relatively prime integers in a line denoting the minimal amount of damage Geralt will receive from Cuber

QQ or -1 if Geralt cannot win at all.

Example

standard input	standard output
3 1 1 2 2 1 1 1 1 2 2 1 1 1 1 1 1 6 6 66 66 66 666 6 666	-1 0/1 2214/37

Problem H. Heat Pipes

Input file: `standard input`
Output file: `standard output`

Cuber QQ is opening a vegetable base in ECNU, providing fresh fruits and vegetables during the lockdown and in case of future emergencies. The vegetable base has many greenhouses. Some of them are connected with heat pipes. Each greenhouse has a temperature controlling system, that can tune the air temperature within the greenhouse.

The ECNUers are picky eating for vegetables. As different people might have different appetites, Cuber QQ has to plant all kinds of vegetables in his base to satisfy their needs. Each vegetable, has a best temperature, under which it will be thriving. As a perfectionist, Cuber QQ won't allow any vegetable to grow in a greenhouse with an imperfect air temperature.

As the seeds of plants haven't arrived yet, Cuber QQ needs to prepare for all possibilities. He assumes that all the best temperatures are between a degrees centigrade and b. This means, at least one greenhouse should be tuned to temperature t for all integers t between a and b, inclusively. Obviously, if he has at least $b - a + 1$ greenhouses and he assigns a specific temperature to each greenhouse, he should be able to do that.

However, the heat pipes between the greenhouses are making everything difficult. The heat pipes do save energy by sharing heats, but at the same time, they impose constraints that if two greenhouses are connected with a pipe, the difference of their temperature should be **exactly one** degrees centigrade. Believe it or not, engineers have put many efforts into maintaining this balance and seeking for the best trade-off between efficiency and controlling precision, but it has been many years since they have built this, and no one knows what they were thinking about when they built this.

Now all we care about is to solve our problem: whether it is possible to assign temperatures to greenhouses that cover all integers from a to b and satisfy the constraints of heat pipes. Also note that, please do not assign a temperature outside this interval as it would be a complete waste and not acceptable.

Input

Each test contains multiple test cases. The first line contains the number of test cases t ($1 \le t \le 1000$).

The first line of each test case contains four space-separated integers n, m, a and b ($1 \le n \le 2\ 000$, $0 \le m \le 50\ 000$, $0 \le a \le b \le n$), the number of greenhouses, the number of heat pipes, and the temperature interval, respectively.

The i-th of the next m lines contains two integers u_i and v_i ($1 \le u_i, v_i \le n$, $u_i \ne v_i$), which is the i-th heat pipe.

It is guaranteed that there is at most one heat pipe between two greenhouses. The sum of n over all test cases does not exceed 2 000, and the sum of m over all test cases does not exceed 50 000 .

Output

For each test case:

- If there is no solution, print "`No`" (without quotes) in a single line.
- Otherwise, print "`Yes`" (without quotes) in the first line, and the second line contains n space-separated integers $x_1, x_2, \ldots, x_n$ ($a \leq x_i \leq b$), where x_i is the temperature you want to tune the i-th greenhouse to.
- If there are multiple answers, print any of them.

You can output "`Yes`" or "`No`" in any cases.

Example

standard input	standard output
2 3 3 1 2 1 2 2 3 3 1 3 2 1 2 1 2 2 3	No Yes 1 2 1

Problem I. Idiotic Suffix Array

Input file: `standard input`
Output file: `standard output`

Cuber QQ did not know about the *Suffix Array* before, now he knows. So he eagerly wants to teach you.

Now he takes a string containing only lowercase letters as an example and builds a *Suffix Array* with the following C++ code on a supercomputer.

```
std::vector<size_t> build(const std::string& s) {
    std::vector<std::pair<std::string, size_t> > suffixes;
    for (size_t i = 0; i < s.length(); ++i) {
        suffixes.emplace_back(s.substr(i), i);
    }
    std::sort(suffixes.begin(), suffixes.end());
    std::vector<size_t> sa(s.length());
    for (size_t i = 0; i < s.length(); ++i) {
        sa[i] = suffixes[i].second;
    }
    return sa;
}
```

For example, the *Suffix Array* of "`ecnu`" is $\{1, 0, 2, 3\}$ and the *Suffix Array* of "`cubercsl`" is $\{2, 5, 0, 3, 7, 4, 6, 1\}$.

It's definitely not satisfying enough for Cuber QQ to show off his algorithm skills, so he wants to test you with a problem.

He will give you two integers n and k ($1 \le k \le n$), you are required to answer him a string of length n containing only lowercase letters and satisfying `sa[k - 1] == 0`, where `sa` is the *Suffix Array* of the string built by the code above.

TL;DR, you are required to answer a string of length n containing only lowercase letters and it ranks k-th smallest among all its suffixes in lexicographical order.

We can show that an answer always exists.

A string a is lexicographically smaller than a string b if and only if one of the following holds:

- a is a prefix of b, but $a \neq b$;
- in the first position where a and b differ, the string a has a letter that appears earlier in the alphabet than the corresponding letter in b.

Input

Two integers n, k ($1 \le k \le n \le 10^5$) — the length of the string and the rank of the string itself among all its suffixes.

Output

Output the answer, which is a string of length n only containing lowercase letters.

If there are multiple answers, print any of them.

Examples

standard input	standard output
4 2	ecnu
8 3	cubercsl