1. ROBO MAZE

Write a **C** program which will simulate robot movements in a maze. The maze has corridors and walls, and robot can move only in one of the four different directions: North, South, East, and West.

As an input you will be given a maze of rectangular shape divided into cells of fixed width. You are guaranteed that the maze will not exceed 6060 dimensions. The maze will be given in the input, with a grid of characters `X' and `.', where `X' denotes a wall and `.' denotes an empty space.

The robot can move through empty spaces, but not through the wall. You can also assume that the robot cannot leave the maze; which is equivalent to the whole maze being surrounded by a wall.

After the maze you will be given the initial position of the robot with two numbers specifying row and column (the top left position is (1,1)). You will assume that the robot is facing North (up on the maze grid). Following this, you will be given a sequence of commands consisting of letters, with possibly having whitespace between them. The letter commands are as follows:

R is a command to rotate the robot 90 degrees clockwise (to the right),
L is a command to rotate the robot 90 degrees counter-clockwise (to the left),
F moves the robot forward one cell, unless there is a wall preventing this, in which case the robot does nothing, and

Q denotes the final position. You should print out the current robot row, column, and orientation, and assume that the next test case follows. You will also print the original map with the letter `R' being placed on all squares visited by the robot.

Input

The input consists of several test cases. Each test case starts with a line with two integer numbers, r and c, where r is the number of rows and c is the number of columns. These numbers will be positive integers, except in the last line, which will be \0 0", indicating the end of input.

After these numbers, there will be r lines describing the maze, each line corresponding to one row of cells. Each row of cells will contain a sequence of c characters describing the content of the cell. The X letter (`X') indicates a wall, and a dot character (`.') indicates an empty cell.

After the maze, the input will include two integers, which are the starting robot position, followed by the commands to move the robot. The commands consist of the letter commands species earlier, with exactly one occurrence of `Q' at the end. The letter commands may be broken by arbitrary whitespaces.

Output

You program must print the final row, column, and orientation of the robot for each test case, followed by the map of robot movements. The orientation is denoted as N (for North or up), S (for South or down), E (for East or left), and W (for West or right). The map that follows should have the letter `R' placed in all squares visited by the robot.

Sample Input

2 2

..

..

1 1

RFRFQ

7 8

XXXXXXXX

X.X.X.XX

X.X... X

X.X.XX.X

X.X.X..X

X.. X.XX

XXXXXXXX

2 4

RRFLFF FFR

FF

RFFQ

0 0

Sample Output

2 2 S

RR

.R

5 6 W

XXXXXXXX

X.XRX.XX

X.XRRRRX

X.X.XXRX

X.X.XRRX

X.. X.XX

XXXXXXXX

---

2. Train and Peter

Peter likes to travel by train. He likes it so much that on the train he falls asleep.

Once in summer Peter was going by train from city A to city B, and as usual, was sleeping. Then he woke up, started to look through the window and noticed that every railway station has a flag of a particular colour.

The boy started to memorize the order of the flags' colours that he had seen. But soon he fell asleep again. Unfortunately, he didn't sleep long, he woke up and went on memorizing the colours. Then he fell asleep again, and that time he slept till the end of the journey.

At the station he told his parents about what he was doing, and wrote two sequences of the colours that he had seen before and after his sleep, respectively.

Peter's parents know that their son likes to fantasize. They give you the list of the flags' colours at the stations that the train passes sequentially on the way from A to B, and ask you to find out if Peter could see those sequences on the way from A to B, or from B to A. Remember, please, that Peter had two periods of wakefulness.

Peter's parents put lowercase Latin letters for colours. The same letter stands for the same colour, different letters — for different colours.

Input

The input data contains three lines. The first line contains a non-empty string, whose length does not exceed 105, the string consists of lowercase Latin letters — the flags' colours at the stations on the way from A to B. On the way from B to A the train passes the same stations, but in reverse order.

The second line contains the sequence, written by Peter during the first period of wakefulness. The third line contains the sequence, written during the second period of wakefulness. Both sequences are non-empty, consist of lowercase Latin letters, and the length of each does not exceed 100 letters. Each of the sequences is written in chronological order.

Output

Output one of the four words without inverted commas:

- «`forward`» — if Peter could see such sequences only on the way from A to B;
- «`backward`» — if Peter could see such sequences on the way from B to A;
- «`both`» — if Peter could see such sequences both on the way from A to B, and on the way from B to A;
- «`fantasy`» — if Peter could not see such sequences.

Examples

input

```
atob
a
b
```

output

```
forward
```

input

```
aaacaaa
aca
aa
```

output

```
both
```

Note

- It is assumed that the train moves all the time, so one flag cannot be seen twice. There are no flags at stations A and B.

3. Amusing Joke

So, the New Year holidays are over. Santa Claus and his colleagues can take a rest and have guests at last. When two "New Year and Christmas Men" meet, their assistants cut out of cardboard the letters from the guest's name and the host's name in honour of this event. Then the hung the letters above the main entrance. One night, when everyone went to bed, someone took all the letters of our characters' names. Then he may have shuffled the letters and put them in one pile in front of the door.

The next morning it was impossible to find the culprit who had made the disorder. But everybody wondered whether it is possible to restore the names of the host and his guests from the letters lying at the door? That is, we need to verify that there are no extra letters, and that nobody will need to cut more letters.

Help the "New Year and Christmas Men" and their friends to cope with this problem. You are given both inscriptions that hung over the front door the previous night, and a pile of letters that were found at the front door next morning.

Input

The input file consists of three lines: the first line contains the guest's name, the second line contains the name of the residence host and the third line contains letters in a pile that were found at the door in the morning. All lines are not empty and contain only uppercase Latin letters. The length of each line does not exceed 100.

Output

Print "YES" without the quotes, if the letters in the pile could be permuted to make the names of the "New Year and Christmas Men". Otherwise, print "NO" without the quotes.

Examples

input

SANTACLAUS
DEDMOROZ
SANTAMOROZDEDCLAUS

Output YES

input

PAPAINOEL
JOULUPUKKI
JOULNAPAOILELUPUKKI

Output NO

input

BABBONATALE
FATHERCHRISTMAS
BABCHRISTMASBONATALLEFATHER

Output

NO

Note

- In the first sample the letters written in the last line can be used to write the names and there won't be any extra letters left.
- In the second sample letter "P" is missing from the pile and there's an extra letter "L".
- In the third sample there's an extra letter "L".

4. Registration system

A new e-mail service "Berlandesk" is going to be opened in Berland in the near future. The site administration wants to launch their project as soon as possible, that's why they ask you to help. You're suggested to implement the prototype of site registration system. The system should work on the following principle.

Each time a new user wants to register, he sends to the system a request with his name. If such a name does not exist in the system database, it is inserted into the database, and the user gets the response OK, confirming the successful registration. If the name already exists in the system database, the system makes up a new user name, sends it to the user as a prompt and also inserts the prompt into the database. The new name is formed by the following rule. Numbers, starting with 1, are appended one after another to name (name1, name2, ...), among these numbers the least i is found so that name does not yet exist in the database.

Input

The first line contains number n ($1 \leq n \leq 105$). The following n lines contain the requests to the system. Each request is a non-empty line, and consists of not more than 32 characters, which are all lowercase Latin letters.

Output Print n lines, which are system responses to the requests: OK in case of successful registration, or a prompt with a new name, if the requested name is already taken.

Examples

input

4
abacaba
acaba
abacaba
acab

output

OK
OK
abacaba1
OK

input

6
first
first
second
second
third
third

output

OK
first1
OK
second1
OK
third1

---

5. Game Winner

The winner of the card game popular in Berland "Berlogging" is determined according to the following rules. If at the end of the game there is only one player with the maximum number of points, he is the winner. The situation becomes more difficult if the number of such players is more than one. During each round a player gains or loses a particular number of points. In the course of the game the number of points is registered in the line "name score", where name is a player's name, and score is the number of points gained in this round, which is an integer number. If score is negative, this means that the player has lost in the round. So, if two or more players have the maximum number of points (say, it equals to m) at the end of the game, than wins the one of them who scored at least m points first. Initially each player has 0 points. It's guaranteed that at the end of the game at least one player has a positive number of points.

Input

The first line contains an integer number n ($1 \le n \le 1000$), n is the number of rounds played. Then follow n lines, containing the information about the rounds in "name score" format in chronological order, where name is a string of lower-case Latin letters with the length from 1 to 32, and score is an integer number between -1000 and 1000, inclusive.

Output

Print the name of the winner.

Examples

input

3
mike 3
andrew 5
mike 2

output

andrew

input

3
andrew 3
andrew 2
mike 5

output

andrew

---

6. Sinking Ship

The ship crashed into a reef and is sinking. Now the entire crew must be evacuated. All n crew members have already lined up in a row (for convenience let's label them all from left to right with positive integers from 1 to n) and await further instructions. However, one should evacuate the crew properly, in a strict order. Specifically:

The first crew members to leave the ship are rats. Then women and children (both groups have the same priority) leave the ship. After that all men are evacuated from the ship. The captain leaves the sinking ship last.

If we cannot determine exactly who should leave the ship first for any two members of the crew by the rules from the previous paragraph, then the one who stands to the left in the line leaves the ship first (or in other words, the one whose number in the line is less).

For each crew member we know his status as a crew member, and also his name. All crew members have different names. Determine the order in which to evacuate the crew.

Input

The first line contains an integer n, which is the number of people in the crew ($1 \le n \le 100$). Then follow n lines. The i-th of those lines contains two words — the name of the crew member who is i-th in line, and his status on the ship. The words are separated by exactly one space. There are no other spaces in the line. The names consist of Latin letters, the first letter is uppercase, and the rest are lowercase. The length of any name is from 1 to 10 characters. The status can have the following values: rat for a rat, woman for a woman, child for a child, man for a man, captain for the captain. The crew contains exactly one captain.

Output

Print n lines. The i-th of them should contain the name of the crew member who must be the i-th one to leave the ship.

Examples

input	output
6 Jack Captain Alice woman Charlie man Teddy rat Bob child Julia woman	Teddy Alice Bob julia Charlie Jack

7. History

Polycarpus likes studying at school a lot and he is always diligent about his homework. Polycarpus has never had any problems with natural sciences as his great-great-grandfather was the great physicist Seinstein. On the other hand though, Polycarpus has never had an easy time with history.

Everybody knows that the World history encompasses exactly n events: the i-th event had continued from the year ai to the year biinclusive ($a_i < b_i$). Polycarpus easily learned the dates when each of n events started and ended (Polycarpus inherited excellent memory from his great-great-granddad). But the teacher gave him a more complicated task: Polycaprus should know when all events began and ended and he should also find out for each event whether it includes another event. Polycarpus' teacher thinks that an event j includes an event i if $a_j < a_i$ and $b_i < b_j$. Your task is simpler: find the number of events that are included in some other event.

Input

The first input line contains integer n ($1 \le n \le 105$) which represents the number of events. Next n lines contain descriptions of the historical events, one event per line. The $i + 1$ line contains two integers ai and bi ($1 \le a_i < b_i \le 109$) — the beginning and the end of the i-th event. No two events start or finish in the same year, that is, $a_i \ne a_j, a_i \ne b_j, b_i \ne a_j, b_i \ne b_j$ for all i, j (where $i \ne j$). Events are given in arbitrary order.

Output

Print the only integer — the answer to the problem.

Examples

input

```
5
1 10
2 9
3 8
4 7
5 6
```

output

```
4
```

input

```
5
1 100
2 50
51 99
52 98
10 60
```

output

```
4
```

input
1
1 1000000000

output
0

Note

In the first example the fifth event is contained in the fourth. Similarly, the fourth event is contained in the third, the third — in the second and the second — in the first.
In the second example all events except the first one are contained in the first.
In the third example only one event, so the answer is 0.

---

8. Lucky Transformation

Petya loves lucky numbers. Everybody knows that lucky numbers are positive integers whose decimal representation contains only the lucky digits 4 and 7. For example, numbers 47, 744, 4 are lucky and 5, 17, 467 are not.

Petya has a number consisting of n digits without leading zeroes. He represented it as an array of digits without leading zeroes. Let's call it d. The numeration starts with 1, starting from the most significant digit. Petya wants to perform the following operation k times: find the minimum x ($1 \le x < n$) such that $dx = 4$ and $dx + 1 = 7$, if x is odd, then to assign $dx = dx + 1 = 4$, otherwise to assign $dx = dx + 1 = 7$. Note that if no x was found, then the operation counts as completed and the array doesn't change at all.

You are given the initial number as an array of digits and the number k. Help Petya find the result of completing k operations.

Input

The first line contains two integers n and k ($1 \le n \le 105, 0 \le k \le 109$) — the number of digits in the number and the number of completed operations. The second line contains n digits without spaces representing the array of digits d, starting with d1. It is guaranteed that the first digit of the number does not equal zero.

Output

In the single line print the result without spaces — the number after the k operations are fulfilled.

Examples

input
7 4
4727447

output
4427477

input

4 2

4478

output

4478

Note In the first sample the number changes in the following sequence: 4727447 → 4427447 → 4427477 → 4427447 → 4427477. In the second sample: 4478 → 4778 → 4478.

---

9. Anagram Search

A string t is called an anagram of the string s, if it is possible to rearrange letters in t so that it is identical to the string s. For example, the string "aab" is an anagram of the string "aba" and the string "aaa" is not.

The string t is called a substring of the string s if it can be read starting from some position in the string s. For example, the string "aba" has six substrings: "a", "b", "a", "ab", "ba", "aba".

You are given a string s, consisting of lowercase Latin letters and characters "?". You are also given a string p, consisting of lowercase Latin letters only. Let's assume that a string is good if you can obtain an anagram of the string p from it, replacing the "?" characters by Latin letters. Each "?" can be replaced by exactly one character of the Latin alphabet. For example, if the string p = «aba», then the string "a??" is good, and the string «?bc» is not.

Your task is to find the number of good substrings of the string s (identical substrings must be counted in the answer several times).

Input

The first line is non-empty string s, consisting of no more than 105 lowercase Latin letters and characters "?". The second line is non-empty string p, consisting of no more than 105 lowercase Latin letters. Please note that the length of the string p can exceed the length of the string s.

Output

Print the single number representing the number of good substrings of string s.

Two substrings are considered different in their positions of occurrence are different. Thus, if some string occurs several times, then it should be counted the same number of times.

Examples

input

bb??x???

aab

output

2

input

ab?c

acb

output

2

Note Consider the first sample test. Here the string s has two good substrings: "b??" (after we replace the question marks we get "baa"), "???" (after we replace the question marks we get "baa"). Let's consider the second sample test. Here the string s has two good substrings: "ab?" ("?" can be replaced by "c"), "b?c" ("?" can be replaced by "a")

10. Divisibility Rules

Vasya studies divisibility rules at school. Here are some of them:

Divisibility by 2. A number is divisible by 2 if and only if its last digit is divisible by 2 or in other words, is even.

Divisibility by 3. A number is divisible by 3 if and only if the sum of its digits is divisible by 3.

Divisibility by 4. A number is divisible by 4 if and only if its last two digits form a number that is divisible by 4.

Divisibility by 5. A number is divisible by 5 if and only if its last digit equals 5 or 0.

Divisibility by 6. A number is divisible by 6 if and only if it is divisible by 2 and 3 simultaneously (that is, if the last digit is even and the sum of all digits is divisible by 3).

Divisibility by 7. Vasya doesn't know such divisibility rule.

Divisibility by 8. A number is divisible by 8 if and only if its last three digits form a number that is divisible by 8.

Divisibility by 9. A number is divisible by 9 if and only if the sum of its digits is divisible by 9.

Divisibility by 10. A number is divisible by 10 if and only if its last digit is a zero.

Divisibility by 11. A number is divisible by 11 if and only if the sum of digits on its odd positions either equals to the sum of digits on the even positions, or they differ in a number that is divisible by 11.

Vasya got interested by the fact that some divisibility rules resemble each other. In fact, to check a number's divisibility by 2, 4, 5, 8 and 10it is enough to check fulfilling some condition for one or several last digits. Vasya calls such rules the 2-type rules.

If checking divisibility means finding a sum of digits and checking whether the sum is divisible by the given number, then Vasya calls this rule the 3-type rule (because it works for numbers 3 and 9).

If we need to find the difference between the sum of digits on odd and even positions and check whether the difference is divisible by the given divisor, this rule is called the 11-type rule (it works for number 11).

In some cases we should divide the divisor into several factors and check whether rules of different types (2-type, 3-type or 11-type) work there. For example, for number 6 we check 2-type and 3-type rules, for number 66 we check all three types. Such mixed divisibility rules are called 6-type rules.

And finally, there are some numbers for which no rule works: neither 2-type, nor 3-type, nor 11-type, nor 6-type. The least such number is number 7, so we'll say that in such cases the mysterious 7-type rule works, the one that Vasya hasn't discovered yet.
Vasya's dream is finding divisibility rules for all possible numbers. He isn't going to stop on the decimal numbers only. As there are quite many numbers, ha can't do it all by himself. Vasya asked you to write a program that determines the divisibility rule type in the b-based notation for the given divisor d.

Input

The first input line contains two integers b and d ($2 \le b, d \le 100$) — the notation system base and the divisor. Both numbers are given in the decimal notation.

Output

On the first output line print the type of the rule in the b-based notation system, where the divisor is d: "2-type", "3-type", "11-type", "6-type" or "7-type". If there are several such types, print the one that goes earlier in the given sequence. If a number belongs to the 2-type, print on the second line the least number of the last b-based digits that we will need to use to check the divisibility.

Examples

input
10 10

output
2-type
1

input
2 3

output
11- type

Note The divisibility rule for number 3 in binary notation looks as follows: "A number is divisible by 3 if and only if the sum of its digits that occupy the even places differs from the sum of digits that occupy the odd places, in a number that is divisible by 3". That's an 11-type rule. For example, $2110 = 101012$. For it the sum of digits on odd positions equals $1 + 1 + 1 = 3$, an on even positions — $0 + 0 = 0$. The rule works and the number is divisible by 3.

In some notations a number can fit into the 3-type rule and the 11-type rule. In this case the correct answer is "3-type".

11. Fibonacci Strings

Fibonacci strings are defined as follows:

f1 = «a»

f2 = «b»

fn = fn - 1 fn - 2, n > 2

Thus, the first five Fibonacci strings are: "a", "b", "ba", "bab", "babba".

You are given a Fibonacci string and m strings si. For each string si, find the number of times it occurs in the given Fibonacci string as a substring.

Input

The first line contains two space-separated integers k and m — the number of a Fibonacci string and the number of queries, correspondingly.

Next m lines contain strings si that correspond to the queries. It is guaranteed that strings si aren't empty and consist only of characters "a" and "b".

The input limitations for getting 30 points are:

$1 \le k \le 3000$

$1 \le m \le 3000$

The total length of strings si doesn't exceed 3000

The input limitations for getting 100 points are:

$1 \le k \le 1018$

$1 \le m \le 104$

The total length of strings si doesn't exceed 105

Please do not use the %lld specifier to read or write 64-bit integers in C++. It is preferred to use cin, cout streams or the %I64dspecifier.

Output

For each string si print the number of times it occurs in the given Fibonacci string as a substring. Since the numbers can be large enough, print them modulo 1000000007 (109 + 7). Print the answers for the strings in the order in which they are given in the input.

Examples

input

```
6 5
a
b
ab
ba
aba
```

output

```
3
5
3
3
1
```

12. Rank List

Another programming contest is over. You got hold of the contest's final results table. The table has the following data. For each team we are shown two numbers: the number of problems and the total penalty time. However, for no team we are shown its final place.

You know the rules of comparing the results of two given teams very well. Let's say that team a solved pa problems with total penalty time ta and team b solved pb problems with total penalty time tb. Team a gets a higher place than team b in the end, if it either solved more problems on the contest, or solved the same number of problems but in less total time. In other words, team a gets a higher place than team b in the final results' table if either $pa > pb$, or $pa = pb$ and $ta < tb$.

It is considered that the teams that solve the same number of problems with the same penalty time share all corresponding places. More formally, let's say there is a group of x teams that solved the same number of problems with the same penalty time. Let's also say that yteams performed better than the teams from this group. In this case all teams from the group share places $y + 1, y + 2, ..., y + x$. The teams that performed worse than the teams from this group, get their places in the results table starting from the $y + x + 1$-th place.

Your task is to count what number of teams from the given list shared the k-th place.

Input

The first line contains two integers n and k ($1 \le k \le n \le 50$). Then n lines contain the description of the teams: the i-th line contains two integers pi and ti ($1 \le pi, ti \le 50$) — the number of solved problems and the total penalty time of the i-th team, correspondingly. All numbers in the lines are separated by spaces.

Output

In the only line print the sought number of teams that got the k-th place in the final results' table.

Examples

input

```
7 2
4 10
4 10
4 10
3 20
2 1
2 1
1 10
```

output

```
3
```

input

```
5 4
3 1
3 1
```

```
5 3
3 1
3 1
```

output

```
4
```

Note

The final results' table for the first sample is:

- 1-3 places — 4 solved problems, the penalty time equals 10
- 4 place — 3 solved problems, the penalty time equals 20
- 5-6 places — 2 solved problems, the penalty time equals 1
- 7 place — 1 solved problem, the penalty time equals 10

The table shows that the second place is shared by the teams that solved 4 problems with penalty time 10. There are 3 such teams.

The final table for the second sample is:

- 1 place — 5 solved problems, the penalty time equals 3
- 2-5 places — 3 solved problems, the penalty time equals 1

The table shows that the fourth place is shared by the teams that solved 3 problems with penalty time 1. There are 4 such teams.

---

13. Reading

Vasya is going to the Olympics in the city Ntown by train. The boy wants to read the textbook to prepare for the Olympics. He counted that he needed k hours for this. He also found that the light in the train changes every hour. The light is measured on a scale from 0 to 100, where 0 is very dark, and 100 is very light.

Vasya has a train lighting schedule for all n hours of the trip — n numbers from 0 to 100 each (the light level in the first hour, the second hour and so on). During each of those hours he will either read the whole time, or not read at all. He wants to choose k hours to read a book, not necessarily consecutive, so that the minimum level of light among the selected hours were maximum. Vasya is very excited before the upcoming contest, help him choose reading hours.

Input

The first input line contains two integers n and k ($1 \le n \le 1000$, $1 \le k \le n$) — the number of hours on the train and the number of hours to read, correspondingly. The second line contains n space-separated integers ai ($0 \le ai \le 100$), ai is the light level at the i-th hour.

Output

In the first output line print the minimum light level Vasya will read at. In the second line print k distinct space-separated integers b1, b2, ..., bk, — the indexes of hours Vasya will read at ($1 \le bi \le n$). The hours are indexed starting from 1. If there are multiple optimal solutions, print any of them. Print the numbers bi in an arbitrary order.

Examples

Input

5 3
20 10 30 40 10

output

20
1 3 4

Input

6 5
90 20 35 40 60 100

output

35
1 3 4 5 6

Note In the first sample Vasya should read at the first hour (light 20), third hour (light 30) and at the fourth hour (light 40). The minimum light Vasya will have to read at is 20.

---

14. Petya and Staircases

Little boy Petya loves stairs very much. But he is bored from simple going up and down them — he loves jumping over several stairs at a time. As he stands on some stair, he can either jump to the next one or jump over one or two stairs at a time. But some stairs are too dirty and Petya doesn't want to step on them.

Now Petya is on the first stair of the staircase, consisting of n stairs. He also knows the numbers of the dirty stairs of this staircase. Help Petya find out if he can jump through the entire staircase and reach the last stair number n without touching a dirty stair once.

One has to note that anyway Petya should step on the first and last stairs, so if the first or the last stair is dirty, then Petya cannot choose a path with clean steps only.

Input

The first line contains two integers n and m ($1 \le n \le 109$, $0 \le m \le 3000$) — the number of stairs in the staircase and the number of dirty stairs, correspondingly. The second line contains m different space-separated integers d1, d2, ..., dm ($1 \le di \le n$) — the numbers of the dirty stairs (in an arbitrary order).

Output

Print "YES" if Petya can reach stair number n, stepping only on the clean stairs. Otherwise print "NO".

Examples

input

10 5
2 4 8 3 6

output

NO

input

10 5
2 4 5 7 9

output

YES

---

15. Prison Transfer

The prison of your city has n prisoners. As the prison can't accommodate all of them, the city mayor has decided to transfer c of the prisoners to a prison located in another city.

For this reason, he made the n prisoners to stand in a line, with a number written on their chests. The number is the severity of the crime he/she has committed. The greater the number, the more severe his/her crime was.

Then, the mayor told you to choose the c prisoners, who will be transferred to the other prison. He also imposed two conditions. They are,

The chosen c prisoners has to form a contiguous segment of prisoners.

Any of the chosen prisoner's crime level should not be greater then t. Because, that will make the prisoner a severe criminal and the mayor doesn't want to take the risk of his running away during the transfer.

Find the number of ways you can choose the c prisoners.

Input

The first line of input will contain three space separated integers n ($1 \le n \le 2 \cdot 10^5$), t ($0 \le t \le 10^9$) and c ($1 \le c \le n$). The next line will contain n space separated integers, the i^{th} integer is the severity i^{th} prisoner's crime. The value of crime severities will be non-negative and will not exceed 10^9.

Output Print a single integer — the number of ways you can choose the c prisoners.

Examples

input

4 3 3
2 3 1 1

output
2

input
1 1 1
2

output
0

input
11 4 2
2 2 0 7 3 2 2 4 9 1 4

output
6

---

16. Harry's assignment

Harry is a bright student. To prepare thoroughly for exams, he completes all the exercises in his book! Now that the exams are approaching fast, he is doing book exercises day and night. He writes down and keeps updating the remaining number of exercises on the back cover of each book.

Harry has a lot of books messed on the floor. Therefore, he wants to pile up the books that still have some remaining exercises into a single pile. He will grab the books one-by-one and add the books that still have remaining exercises to the top of the pile.

Whenever he wants to do a book exercise, he will pick the book with the minimum number of remaining exercises from the pile. In order to pick the book, he has to remove all the books above it. Therefore, if there are more than one books with the minimum number of remaining exercises, he will take the one which requires the least number of books to remove. The removed books are returned to the messy floor. After he picks the book, he will do all the remaining exercises and trash the book.

Since number of books is rather large, he needs your help to tell him the number of books he must remove, for picking the book with the minimum number of exercises.
Note that more than one book can have the same name.

Input

The first line contains a single integer N denoting the number of actions. Then N lines follow. Each line starts with an integer. If the integer is -1, that means Harry wants to do a book exercise. Otherwise, the integer is number of the remaining exercises in the book he grabs next. This is followed by a string denoting the name of the book.

Output

For each -1 in the input, output a single line containing the number of books Harry must remove, followed by the name of the book that Harry must pick.

Constraints

$1 < N \leq 1{,}000{,}000$
$0 \leq$ (the number of remaining exercises of each book) $< 100{,}000$
The name of each book consists of between 1 and 15 characters 'a' - 'z'. Whenever he wants to do a book exercise, there is at least one book in the pile.

Example
Input:
6

9 english

6 mathematics

8 geography

-1

3 graphics

-1

Output:
1 mathematics

0 graphics

---

17. Teddy and Tracy

Teddy and Tracy like to play a game based on strings. The game is as follows. Initially, Tracy writes a long random string on a whiteboard. Then, each player starting with Teddy makes turn alternately. Each turn, the player must erase a contiguous substring that exists in the dictionary. The dictionary consists of N words.

Of course, the player that can't erase any substring in his turn loses the game, and the other player is declared the winner.

Note that after a substring R is erased, the remaining substring becomes separated, i.e. they cannot erase a word that occurs partially to the left of R and partially to the right of R.
Determine the winner of the game, assuming that both players play optimally.

Input

The first line contains a single integer T, the number of test cases. T test cases follow. The first line of each test case contains a string S, the string Tracy writes on the whiteboard. The next line

contains a single integer N. N lines follow. The i-th line contains a single string wi, the i-th word in the dictionary.

Output

For each test case, output a single line containing the name of the winner of the game.

Example

Input:

3

codechef

2

code

chef

foo

1

bar

mississippi

4

ssissi

mippi

mi

ppi

Output:

Tracy

Tracy

Teddy

Constraints

- 1 <= T <= 5
- 1 <= N <= 30
- 1 <= $|S|$ <= 30
- 1 <= $|w_i|$ <= 30
- S and w_i contain only characters 'a'-'z'

18. Hourglass

You are given a 2D array with dimensions 6*6. An hourglass in an array is defined as a portion shaped like this:

```
a b c
  d
e f g
```

For example, if we create an hourglass using the number 1 within an array full of zeros, it may look like this:

```
1 1 1 0 0 0
0 1 0 0 0 0
1 1 1 0 0 0
0 0 0 0 0 0
0 0 0 0 0 0
0 0 0 0 0 0
```

Actually, there are many hourglasses in the array above. The three leftmost hourglasses are the following:

```
1 1 1    1 1 0    1 0 0
  1        0        0
1 1 1    1 1 0    1 0 0
```

The sum of an hourglass is the sum of all the numbers within it. The sums for the hourglasses above are 7, 4, and 2, respectively. In this problem, you have to print the largest sum among all the hourglasses in the array.

Input Format

There will be exactly 6 lines of input, each containing 6 integers separated by spaces. Each integer will be between -9 and 9, inclusively.

Output Format

Print the answer to this problem on a single line.

Sample Input

```
1 1 1 0 0 0
0 1 0 0 0 0
1 1 1 0 0 0
0 0 2 4 4 0
0 0 0 2 0 0
0 0 1 2 4 0
```

Sample Output

Explanation

The hourglass possessing the largest sum is:

```
2 4 4
  2
1 2 4
```

19. HOTEL MANAGEMENT SYSTEM

Develop an application for hotel management system with the following modules using structures, pointers to structure variables, passing structure pointers to function.

1. Get availability
2. Features of room
3. Room allocation
4. Show customer details
5. Room deallocation
6. Restaurant
7. Billing.

Make your own assumptions for this project, design and implement hotel management system.

20. Array Manimpulation

Project Description: By Using this Program User can able to do different following operations using Arrays. 1. Insertion, 2. Deletion, 3. Sorting, 4. Searching, 5. Update, 6. Retrieve, 7. Merging, 8. Append, 9. Exit. This Menu Driven Program will enhance the array operations capability in c programming. In this program User can insert, delete, sort, search, update, retrieve, merge, and append the elements in array with respect to the user choice.

21. Modern Periodic Table

This project will help you to understand file handling in C i.e. creating a file and accessing the stored data in the file, modifying and removing the stored data. It will also help you to understand the use of functions as well as different parameters of C programming language.

The key **features** of Modern Periodic Table mini project in C are briefly described below:

- **Storage of Element Information:** In the project, you can add any new element with its name, symbol, atomic number, atomic weight and its some important properties. When new element information is to be added to this Modern Periodic Table, you have to enter 1 in the main menu and input information in given format. This information is stored in file created on the hard disk of computer by program itself.

- **Exploration of element Information:** Another main function of this project is to explore or to display the stored information. You can search an element by using any of the following method:
    1. By name of element
    2. By symbol of element
    3. By atomic number of element
    4. By atomic weight of element

- If you press 3 in the main menu, the program will be terminated.

22. Mayor's Race

The results from the mayor's race have been reported by each precinct as follows:

Precinct	Candidate A	Candidate B	Candidate C	Candidate D
1	192	48	206	37
2	147	90	312	21
3	186	12	121	38
4	114	21	408	39
5	267	13	382	29

Write a program to do the following:

a. Read the raw vote totals from a data file that contains one row for each precinct.

b. Display the table with appropriate headings for the rows and columns.

c. Compute and display the total number of votes received by each candidate and the percent of the total votes cast.

d. If any one candidate received over 50% of the votes, the program should print a message declaring that candidate the winner.

e. If no candidate received 50% of the votes, the program should print a message declaring a run-off between the two candidates receiving the highest number of votes; the two candidates should be identified by their letter names.

f. For testing, run the program with the above data, and also with another data file where Candidate C receives only 108 votes in precinct 4.

---

23. Square Code

One classic method for composing secret messages is called a square code. The spaces are removed from the English text and the characters are written into a square (or rectangle). For example, the sentence "If man was meant to stay on the ground god would have given us roots" is 54 characters long, so it is written into a rectangle with 7 rows and 8 columns.

ifmanwas
meanttos
tayonthe
groundgo
dwouldha
vegivenu
sroots

The coded message is obtained by reading down the columns going left to right. For example, the message above is coded as:

imtgdvs fearwer mayoogo anouuio ntnnlvt wttddes aohghn sseoau

In your program, have the user enter a message in english with no spaces between the words.

Have the maximum message length be 81 characters. Display the encoded message. (Watch out that no "garbage" characters are printed.) Here are some more examples:

Input	Output
haveaniceday	hae and via ecy
feedthedog	fto ehg ee dd
chillout	clu hlt io.

25. Implementation of stack Applications

Description: The main aim of this project is to implement the various applications of stack data structure in C programming language. There are three modules in this project

- **Conversion of infix expression to post fix expression**: user enters infix expression as input, it converts given infix expression to postfix expression
- **Evaluation of postfix expressions**: it takes the postfix expression generated in above module and evaluates that postfix expression
- **Balancing of symbols**: user enters an expression with all kinds of symbols, it validates whether all the brackets in given expression are properly used or not

Requirements: To implement this project student should have knowledge on

- Working of stack data structure
- Pointers, strings, functions and structures in C Programming Language
- Operator precedence and associativity rules

26. Implementation of Linked List Applications

Description: The main aim of this project is to implement one of the most important applications of linked list such as polynomial operations such as addition, subtraction, multiplication and derivation. To perform these operations each polynomial needs to represent in one linked list and each node in the list contains three parts to store coefficient, exponent and link to next term of polynomial respectively. There are four modules in in this project

- polynomial Addition
- polynomial subtraction
- polynomial multiplication
- polynomial derivation

Requirements: To implement this project student should have knowledge on

- creating using linked lists
- pointers, self-referential structures
- polynomial operations

27. Movie Ticket Booking

Description: The main purpose of online ticket booking system is to provide another way for the customer to buy cinema ticket. It is an automatic system, where we will automate the reservation of tickets and enquiries about availability of tickets. After inserting the data to file, staff need not to worry about the orders received through the system and hence reduces the manual work. One of the best features of the system is to refund the amount on cancellation of tickets by customer.

The modules in this project are

- To provide an anytime anyplace service for the customer
- To provide refund
- To minimize the number of staff at the ticket box
- To promote the film on the internet
- To increase the profit To obtain statistic information from the booking record

---

28. Employee Management

Description: Create an employee Management system using Linked List. The data should be kept in a file.

Do the following operations.

1. Creation
2. Insertion
3. Deletion
4. Search
5. Update
6. Sort b
7. Display the data
8. Merge two separate lists i.e Merge 2 departments data

---

29. Tricky Tower

There are three circular disks having a hole in the center and placed on a peg and there are two empty pegs. The three disks are of different sizes. It is now required to transfer the three disks (named A, B, and C : C is smallest) from the source peg P1 to the target peg P2 using a standby peg P3 such that

(i) only one disk can be moved at a time and

(ii) no disk can be kept on top of a disk with smaller diameter.

Build a system that gives the solution for the above problem.

[Hint: Assume that there are three stacks for three pegs SP1, SP2, and SP3. Transferring a disk

from one peg to other involves ADD and DELETE operations of stacks]

30. Stock Table

Companies and people often buy and sells stocks. Often they buy the same stock for different prices at different times. Say a person owns 1000 shares a certain stock (such as Checkpoint), she may have bought the stock in amounts of 100 shares over 10 different times with 10 different prices.

In this assignment, you will be using a stack for LIFO accounting. You should use an array based implementation for your stack based implementation or a linked list for implementing your stack. Your stack should have records with the following fields:

- The name of the stock (a string or int)
- The number of shares of a stock (an int)
- The purchase price (can be a decimal)

You can assume that the first element of the structure is the security bought first, the second was bought second, etc.

Create a program that should have the user able to enter information about various stocks, the amount of shares, and the price. The user can then enter a query about a certain stock and the cost according to the LIFO accounting methods for a certain number of shares.

The following could be your menu:

- Press 1 to enter a new stock.
- Press 2 to find the LIFO price for a stock.
- If 1 is pressed, the user needs to enter the stock symbol, and the number of shares, and the price.
- If 2 is pressed, the user needs to enter the stock symbol being queried and the number of shares in question.