Problem A. DFS Order

Input file:	`standard input`
Output file:	`standard output`
Time limit:	3 seconds
Memory limit:	1024 megabytes

Prof. Pang has a rooted tree which is rooted at 1 with n nodes. These n nodes are numbered from 1 to n.

Now he wants to start the depth-first search at the root. He wonders for each node v, what is the minimum and the maximum position it can appear in the **depth-first search order**. The depth-first search order is the order of nodes visited during the depth-first search. A node appears in the j-th ($1 \le j \le n$) position in this order means it is visited after $j-1$ other nodes. Because sons of a node can be iterated in arbitrary order, multiple possible depth-first orders exist. Prof. Pang wants to know for each node v, what are the minimum value and the maximum value of j such that v appears in the j-th position.

Following is a pseudo-code for the depth-first search on a rooted tree. After its execution, `dfs_order` is the depth-first search order.

```
let dfs_order be an empty list

def dfs(vertex x):
    append x to the end of dfs_order.
    for (each son y of x): // sons can be iterated in arbitrary order.
        dfs(y)

dfs(root)
```

Input

The first line contains a single integer T ($1 \le T \le 10^6$) denoting the number of test cases.

For each test case, the first line contains an integer n ($1 \le n \le 10^5$). Each of the next $n-1$ lines contains two integers x and y, indicating node x is node y's parent ($1 \le x, y \le n$). These edges form a tree rooted at 1.

It is guaranteed that the sum of n over all test cases is no more than 10^6.

Output

For each test case, print n lines. The i-th line contains two integers denoting the minimum and the maximum position node i can appear in the depth-first search order.

Example

standard input	standard output

2
4
1 2
2 3
3 4
5
1 2
2 3
2 4
1 5
	1 1
	2 2
	3 3
	4 4
	1 1
	2 3
	3 5
	3 5
	2 5

Problem B. Beautiful String

Input file:	`standard input`
Output file:	`standard output`
Time limit:	3 seconds
Memory limit:	1024 megabytes

Prof. Pang recently got a dictionary of the elvish language, including many strings representing their words. He thinks a partition of string s is beautiful if both of the following conditions are satisfied:

- $s = s_1 + s_2 + s_3 + s_4 + s_5 + s_6$, where $s_i (1 \le i \le 6)$ are nonempty substrings. $a + b$ means the concatenation of string a and b here.
- $s_1 = s_2 = s_5, s_3 = s_6$.

For example, you can partition the string "114514" into 6 parts : "114514" = "1" + "1" + "4" + "5" + "1" + "4". The first, second, fifth parts are the same, and the third and sixth parts are the same. Thus, the partition of $s =$"114514" into $s_1 =$"1", $s_2 =$"1", $s_3 =$"4", $s_4 =$"5", $s_5 =$"1", and $s_6 =$"4" is beautiful.

Accordingly, the beauty of a string s is defined as the number of beautiful partitions of s.

Given a string t, please help Prof. Pang to figure out the sum of beauties of all substrings of t.

Input

The first line contains a single integer T $(1 \le T \le 50)$ indicating the number of test cases.

For each test case, there is one single line containing the string t, consisting of digits from '0' to '9'.

It is guaranteed that the length of each t in each test case will not exceed 5000 and the total length will not exceed 30000.

Output

For each test case, output a single line containing an integer, indicating the sum of beauties of all substrings of t.

Example

standard input	standard output
2 114514 0000000	1 3

Problem C. Bus

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	1024 mebibytes

Sherlock is on the case of the cheating at the bus payments.

He chose the bus and counted the passengers inside.

The bus fare equals one shilling in London. However, not everything is that easy — **no more than one** child can ride for free with each grown-up passenger. That means that a grown-up passenger who rides with his k ($k > 0$) children, pays overall k rubles: a ticket for himself and $(k-1)$ tickets for his children. Also, a grown-up can ride without children, in this case he only pays one shilling. Sherlock knows that in England children can't ride in a bus unaccompanied by grown-ups.

Help Sherlock count the minimum and the maximum sum in shillings, that all passengers of this bus could have paid in total.

Input

The input file consists of a single line containing two space-separated numbers n and m ($0 \le n, m \le 10^5$) — the number of the grown-ups and the number of the children in the bus, correspondingly.

Output

If n grown-ups and m children could have ridden in the bus, then print on a single line two space-separated integers — the minimum and the maximum possible total bus fare, correspondingly.

Otherwise, print "`Impossible`" (without the quotes).

Examples

standard input	standard output
`1 2`	`2 2`
`0 5`	`Impossible`
`2 2`	`2 3`

Problem D. Two Walls

Input file:	`standard input`
Output file:	`standard output`
Time limit:	4 seconds
Memory limit:	1024 megabytes

Prof. Pang has bought a humanoid cleaning robot to clean his yard. The robot is not very sophisticated. It can either move forward or change its direction at a time, all controlled by Prof. Pang's controller.

Prof. Pang's yard is a 2D plane. The robot needs to move from its current location A to the destination B to fulfill some "cleaning" needs of Prof. Pang. However, there are two straight walls CD and EF in Prof. Pang's yard. Since the robot is clumsy, it will fall over if it touches any of the walls (even at endpoints).

Now that Prof. Pang is lazy, he wants to minimize the number of times the robot changes its direction. Can you help him?

Input

The first line contains a single integer T ($1 \le T \le 10^5$) denoting the number of test cases.

For each test case, the first line contains two integers x_A, y_A, the coordinates of point A. The second line contains two integers x_B, y_B, the coordinates of point B. The third line contains four integers x_C, y_C, x_D, y_D, the coordinates of point C and D which are the endpoints of the first wall. The fourth line contains four integers x_E, y_E, x_F, y_F, the coordinates of point E and F which are the endpoints of the second wall.

It is guaranteed that neither the current location A nor the destination B of the robot are on any of the walls. A wall may degenerate to a point. It can be proved that the robot can always move from A to B without touching any of the walls. All values lie within $[-10^9, 10^9]$.

Output

For each test case, print one number d in one line, denoting the minimum number of turns.

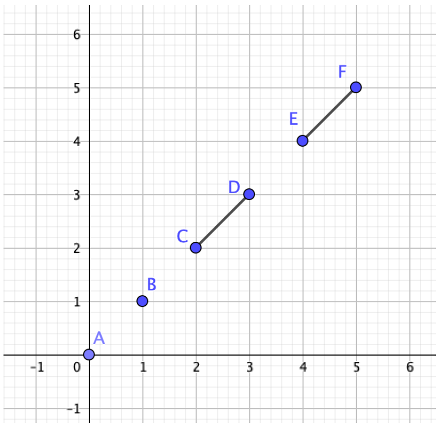
Example

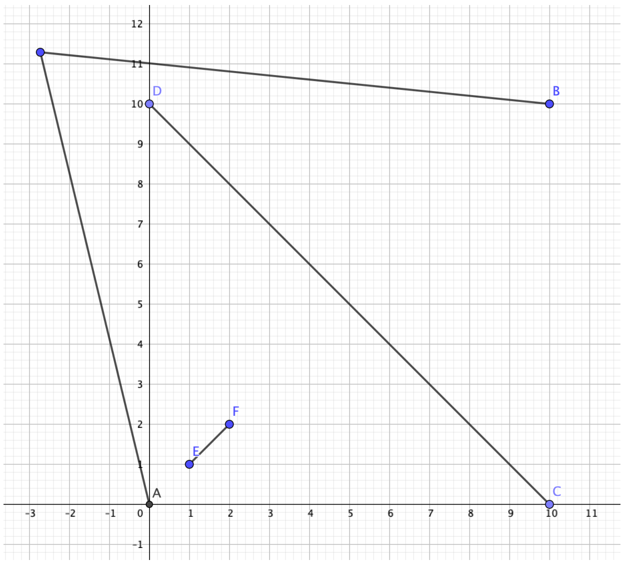
standard input	standard output

3
0 0
1 1
2 2 3 3
4 4 5 5
0 0
1 1
2 2 3 3
2 2 3 3
0 0
10 10
10 0 0 10
1 1 2 2
	0
	0
	1

Note

The following are illustrations for the first sample and the third sample.


A
B
C
D
E
F

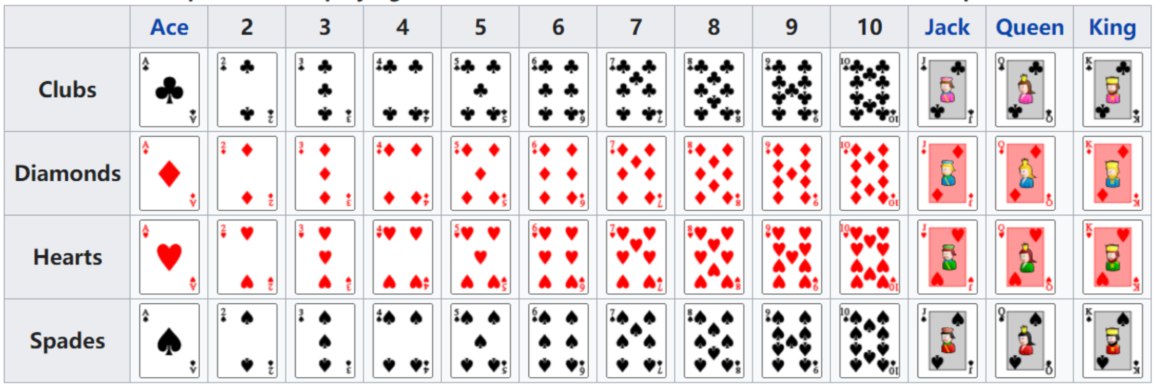

A
B
C
D
E
F

Problem E. Prof. Pang and Poker

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	1024 megabytes

Prof. Pang is playing a card game with his two friends Alice and Bob. All cards are drawn from a standard 52-card deck. A standard 52-card deck comprises 13 ranks in each of the four French suits: clubs (♣), diamonds (♢), hearts (♡) and spades (♠). Each suit includes an Ace (A), a King (K), a Queen (Q), and a Jack (J), each depicted alongside a symbol of its suit; and numerals or pip cards from the Deuce (Two) to the Ten, with each card depicting that many symbols (pips) of its suit. **No card can be drawn more than once.**

Example set of 52 playing cards; 13 of each suit: clubs, diamonds, hearts, and spades



Individual cards are ranked as follows (high-to-low): A, K, Q, J, 10, 9, 8, 7, 6, 5, 4, 3, 2. **Suits do not affect ranks of the cards.** For example, Ace of diamonds and Ace of clubs have the same rank. No one of them is ranked strictly higher than the other.

Initially, Alice and Bob will hold one or more cards while Prof. Pang will hold exactly one card. **Each player can see cards held by himself/herself and cards held by other players.** They will play the game in the following multi-round rule:

- The initiative player chooses one card and put it out to start one round.
- The next player can pass or put out a new card, then the player after the next can also pass or put out a new card, and so on. The only constraint is that the rank of the newly put card should be strictly higher than all previous cards in this round.
- The round ends when two players choose to pass consecutively. The one who put out the last card becomes the initiative player in the next round.
- If someone put out all the cards in his/her hand, the game ends immediately.

In this game, Alice is the initiative player in the first round. Bob, Prof. Pang, and Alice are the next players of Alice, Bob, and Prof. Pang respectively. Prof. Pang will be happy if and only if he is the one that first put out all the cards. (Prof. Pang wants to be happy, of course.) Alice wants to drink milk tea so she decides to make Prof. Pang happy and then asks him to buy milk tea for her. However, Bob doesn't want it to happen, so he decides to avoid Prof. Pang from being happy. If they play the game optimally for themselves, will Prof. Pang be happy in the end?

Input

The first line contains a single integer T ($1 \le T \le 10^4$) denoting the number of test cases. For each test case:

The first line contains two integers n, m ($1 \le n, m \le 50$) denoting the number of cards in Alice's hand and Bob's hand initially.

The second line contains n strings a_i ($1 \le i \le n$) denoting the cards in Alice's hand.

The third line contains m strings b_i ($1 \le i \le m$) denoting the cards in Bod's hand.

The fourth line contains one string p denoting the card in Prof. Pang's hand.

For each card, the first character of its corresponding string denotes its rank. (Possible ranks are '2' ,'3','4','5','6','7','8','9','T','J','Q','K','A'. 'T' denotes 10.) The second character denotes its suit. 'C' denotes clubs. 'D' denotes diamonds. 'H' denotes hearts. 'S' denotes spades.

It is guaranteed that each card appears at most once in one test case.

Output

For each test case, print one line. Print "`Pang`" if Prof. Pang will be happy. Otherwise, print "`Shou`".

Example

standard input	standard output
2 2 2 2H 2D 3H 3D 4S 2 2 2H 2D 3H 4D 4S	Pang Shou

Note

- For the first case, Prof. Pang can always put out his only card "4S".
- For the second case, Bob can put out "4D" and become the initiative player in the second round regardless of the card Alice put out in the first round, then Bob put out "3H" and the game ends.

Problem F. Byteazar's Cypher

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	1024 mebibytes

To ensure that his mathematical papers will not be stolen, professor Byteazar used special cypher for them. More, the same operation scrambles the message as unscrambles it.

This operation is performed by replacing vowels in the sequence

(a i y e o u)

with the vowel three advanced, cyclicly, while preserving case (i.e., lower or upper). Similarly, consonants are replaced from the sequence

(b k x z n h d c w g p v j q t s r l m f)

by advancing ten letters.

The fascinating thing about this transformation is that the resulting language yields pronounceable words, so it looks as the paper is written on some rare language.

For this problem, you will write code to translate Byteazar's papers into plain text.

Input

First line of the input contains integer T $(1 \le T \le 20)$ — number of test cases.

Each test case consists of a single line containing up to 100 characters, representing some text written by Byteazar. All characters will be plain ASCII, in the range space (32) to tilde (126), plus a newline terminating each line.

Output

For each input test case, print its translation into plaintext. The output should contain exactly the same number of lines and characters as the input.

Example

standard input	standard output
1 Fokat otnafad T.	Given integer N.

Problem G. The Magic Number

Input file:	*standard input*
Output file:	textslstandard output
Time limit:	2 seconds
Memory limit:	1024 mebibytes

Given an integer n. Find the smallest positive integer k such that for any digit (0 through 9) there is at least one of the integers $n, 2n, 3n, 4n, 5n, \ldots, kn$ that contains it in the decimal representation.

Input

First line of the input contains integer T ($1 \le T \le 30$) — number of the testcases. Each of next n lines contains one integer n ($1 \le n \le 2 \cdot 10^8$).

Output

For each test case, print the key k at the separate line.

Example

standard input	textslstandard output
3 1 10 3141592	10 9 5

Problem H. Continuous Fraction

Input file: *standard input*
Output file: *standard output*
Time limit: 1 second
Memory limit: 1024 mebibytes

Given the formula

$$\cfrac{b_1}{a_1 + \cfrac{b_2}{a_2 + \cfrac{b_3}{\cdots \cfrac{\cdots}{a_{n-1} + \frac{b_n}{a_n}}}}}$$

and the values of a_i and b_i, calculate the result.

Input

The first line contains only one integer T $(1 \le T \le 500)$, which indicates the number of test cases.

For each test case, the first line contains one integer n $(n \le 8)$. The second line contains n integers: $a_1, a_2, \ldots, a_n$ $(1 \le a_i \le 10)$. The third line contains n integers: $b_1, b_2, \ldots, b_n$ $(1 \le b_i \le 10)$.

Output

For each case, print the result as irreducible fraction p/q. If the result is integer, consider $q = 1$.

Example

standard input	standard output

1
2
1 1
2 3
	1/2

Problem I. Future Coder

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	1024 megabytes

Prof. Pang builds his famous coding team recently. To pursue a gold medal in ICPC, hundreds of pupils join his team. Unfortunately, one of Prof. Pang's students believes that for any integers a and b, $a \times b \geq a + b$. To disprove this proposition, Prof. Pang writes n numbers $a_1, a_2, \ldots, a_n$ on a paper and wants you to count how many pairs of numbers (a_i, a_j) $(1 \leq i < j \leq n)$ satisfies $a_i \times a_j < a_i + a_j$.

Input

The first line contains a single integer T $(1 \leq T \leq 10^6)$ denoting the number of test cases.

For each test case, the first line contains a single integer n $(1 \leq n \leq 10^6)$. The second line contains n integers $a_1, a_2, \ldots, a_n$ $(-10^9 \leq a_i \leq 10^9)$.

It is guaranteed that the sum of n over all test cases will not exceed 10^6.

Output

For each test case, print one line containing the answer.

Example

standard input	standard output

2
8
3 -1 4 1 -5 9 2 -6
1
0
	19
	0

Problem J. Elden Ring

Input file:	`standard input`
Output file:	`standard output`
Time limit:	3 seconds
Memory limit:	1024 megabytes

Prof. Pang is getting addicted to the game called Elden Ring, in which the world is a connected graph including n vertices indexed from 1 to n and m undirected edges. Players start at vertex 1 and travel across the world to slay the god on vertex n.

However, it's not that easy. For any vertex i except vertex 1, there is exactly one boss whose level is l_i, and the player starts the game with level l_1. For each day, the player can travel to any vertex i from vertex 1 and challenge the boss there. If the current level of the player is greater than the boss, the boss will be eliminated from the world (inactivated) and the level of the player will be increased by A. Notice that traveling through a vertex that has an active boss is forbidden. (In other words, Prof. Pang can travel from vertex 1 to vertex i if there is a path in the graph from vertex 1 to vertex i such that each vertex on this path, except for vertex i, has no active boss.) Meanwhile, at the beginning of each day, all the remaining bosses in the world will also be promoted by B levels.

To finish a playthrough of the game, you need to slay the boss on vertex n (Elden Beast). Given the information of the world, Prof. Pang is wondering how many days he needs at least to do so.

The Player can only challenge one boss each day.

Input

The first line contains a single integer T ($1 \le T \le 10^5$) denoting the number of test cases.

For each test case, the first line includes four integers n, m, A, B ($2 \le n \le 2 \times 10^5, 1 \le m, A, B \le 2 \times 10^5$). In next m lines, each line contains two integers a_i, b_i ($1 \le a_i, b_i \le n$), denoting the endpoints of the i-th undirected edge. The last line contains n integers l_i ($1 \le l_i \le 2 \times 10^5$), representing the initial levels of the player and bosses mentioned above.

It is guaranteed that the sum of n over all test cases will not exceed 10^6 and the sum of m over all test cases will not exceed 10^6.

Output

For each test case, output a single line containing an integer, indicating the minimum number of days Prof. Pang needs to finish the game. If it is impossible to do so, please output -1.

Example

standard input	standard output

2
5 4 5 8
1 2
1 3
1 4
4 5
15 1 1 1 1
5 4 10 5
1 2
1 3
1 4
4 5
10 4 4 4 19
	2
	4

Problem K. Morse Alphabet

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 second
Memory limit:	1024 mebibytes

Your task is to implement a decoder of the famous Morse alphabet. As most of you know, the Morse code represents characters as variable-length sequences of short and long signals ("beeps"), often written as dots and dashes. The following table shows the Morse code sequences for all letters:

```
A .-     B -...   C -.-.   D -..    E .      F ..-.
G --.    H ....   I ..     J .---   K -.-    L .-..
M --     N -.     O ---    P .--.   Q --.-   R .-.
S ...    T -      U ..-    V ...-   W .--    X -..-
Y -.--   Z --..
```

If more letters are to be transferred, they are separated by a short pause, typically written as a slash. A space between words is represented by an even longer pause, written as two slashes.

Input

The input contains no more than 1234 test cases. Each test case is specified on one line with at most 1000 characters, which describes a valid Morse code transmission:

- The line consists only of dashes ('`-`'), dots ('`.`'), and slashes ('`/`').
- There is at least one character.
- The first and last characters will never be a slash.
- There will never be more than two slashes together.
- Each non-empty sequence between two slashes contains a valid Morse code of one letter.

Output

For each test case, print one line containing the decoded message in uppercase letters.

Example

standard input	standard output
`../-.-./.--./-.-.`	`ICPC`
`--./---/---/-..//.-../..-/-.-./-.-`	`GOOD LUCK`

Problem L. Fenwick Tree

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	1024 megabytes

Prof. Pang is giving a lecture on the Fenwick tree (also called binary indexed tree).

In a Fenwick tree, we have an array $c[1 \dots n]$ of length n which is initially all-zero ($c[i] = 0$ for any $1 \le i \le n$). Each time, Prof. Pang can call the following procedure for some position pos ($1 \le pos \le n$) and value val:

```
def update(pos, val):
    while (pos <= n):
        c[pos] += val
        pos += pos & (-pos)
```

Note that `pos & (-pos)` equals to the maximum power of 2 that divides `pos` for any positive integer `pos`.

In the procedure, val can be **any real** number. After calling it some (zero or more) times, Prof. Pang forgets the exact values in the array c. He only remembers whether $c[i]$ is zero or not for each i from 1 to n. Prof. Pang wants to know what is the minimum possible number of times he called the procedure assuming his memory is accurate.

Input

The first line contains a single integer T ($1 \le T \le 10^5$) denoting the number of test cases.

For each test case, the first line contains an integer n ($1 \le n \le 10^5$). The next line contains a string of length n. The i-th character of the string is `1` if $c[i]$ is nonzero and `0` otherwise.

It is guaranteed that the sum of n over all test cases is no more than 10^6.

Output

For each test case, output the minimum possible number of times Prof. Pang called the procedure. It can be proven that the answer always exists.

Example

standard input	standard output
3 5 10110 5 00000 5 11111	3 0 3

Note

For the first example, Prof. Pang can call `update(1,1)`, `update(2,-1)`, `update(3,1)` in order.

For the third example, Prof. Pang can call `update(1,1)`, `update(3,1)`, `update(5,1)` in order.

Problem M. Two Sequences

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	1024 mebibytes

Given two sequences $a_1, a_2, \ldots, a_n$ and $b_1, b_2, \ldots, b_m$ and a number p. Calculate the number of positions q such that sequence $b_1, b_2, \ldots, b_m$ is exactly the sequence $a_q, a_{q+p}, a_{q+2p}, \ldots, a_{q+(m-1)p}$, where $q + (m-1)p \le n$ and $q \ge 1$.

Input

The first line contains only one integer $T \le 100$, which indicates the number of test cases. Each test case contains three lines.

The first line contains three space-separated integers $1 \le n \le 4 \cdot 10^4$, $1 \le m \le 4 \cdot 10^4$ and $1 \le p \le 4 \cdot 10^4$. The second line contains n integers $a_1, a_2, \ldots, a_n$ $(1 \le a_i \le 10^9)$. The third line contains m integers $b_1, b_2, \ldots, b_m$ $(1 \le b_i \le 10^9)$.

It is guaranteed that sum of all n in the input does not exceed $2 \cdot 10^6$.

Output

For each test case, print on the separate line one integer y which is is the number of valid q's.

Example

standard input	standard output

2
6 3 1
1 2 3 1 2 3
1 2 3
6 3 2
1 3 2 2 3 1
1 2 3
	2
	1