Educational Codeforces Round 115 (Rated for Div. 2)

A. Computer Game

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Monocarp is playing a computer game. Now he wants to complete the first level of this game.

A level is a rectangular grid of 2 rows and n columns. Monocarp controls a character, which starts in cell $(1, 1)$ — at the intersection of the 1-st row and the 1-st column.

Monocarp's character can move from one cell to another in one step if the cells are adjacent by side and/or corner. Formally, it is possible to move from cell (x_1, y_1) to cell (x_2, y_2) in one step if $|x_1 - x_2| \le 1$ and $|y_1 - y_2| \le 1$. Obviously, it is prohibited to go outside the grid.

There are traps in some cells. If Monocarp's character finds himself in such a cell, he dies, and the game ends.

To complete a level, Monocarp's character should reach cell $(2, n)$ — at the intersection of row 2 and column n.

Help Monocarp determine if it is possible to complete the level.

Input

The first line contains a single integer t ($1 \le t \le 100$) — the number of test cases. Then the test cases follow. Each test case consists of three lines.

The first line contains a single integer n ($3 \le n \le 100$) — the number of columns.

The next two lines describe the level. The i-th of these lines describes the i-th line of the level — the line consists of the characters '0' and '1'. The character '0' corresponds to a safe cell, the character '1' corresponds to a trap cell.

Additional constraint on the input: cells $(1, 1)$ and $(2, n)$ are safe.

Output

For each test case, output YES if it is possible to complete the level, and NO otherwise.

Example

input
4 3 000 000 4 0011 1100 4 0111 1110 6 010101 101010
output
YES YES NO YES

Note

Consider the example from the statement.

In the first test case, one of the possible paths is $(1, 1) \to (2, 2) \to (2, 3)$.

In the second test case, one of the possible paths is $(1, 1) \to (1, 2) \to (2, 3) \to (2, 4)$.

In the fourth test case, one of the possible paths is $(1, 1) \to (2, 2) \to (1, 3) \to (2, 4) \to (1, 5) \to (2, 6)$.

B. Groups

time limit per test: 4 seconds
memory limit per test: 256 megabytes
input: standard input

output: standard output

n students attended the first meeting of the Berland SU programming course (n is even). All students will be divided into two groups. Each group will be attending exactly one lesson each week during one of the five working days (Monday, Tuesday, Wednesday, Thursday and Friday), and the days chosen for the groups must be different. Furthermore, both groups should contain the same number of students.

Each student has filled a survey in which they told which days of the week are convenient for them to attend a lesson, and which are not.

Your task is to determine if it is possible to choose two different week days to schedule the lessons for the group (the first group will attend the lesson on the first chosen day, the second group will attend the lesson on the second chosen day), and divide the students into two groups, so the groups have equal sizes, and for each student, the chosen lesson day for their group is convenient.

Input

The first line contains a single integer t ($1 \le t \le 10^4$) — the number of testcases.

Then the descriptions of t testcases follow.

The first line of each testcase contains one integer n ($2 \le n \le 1\,000$) — the number of students.

The i-th of the next n lines contains 5 integers, each of them is 0 or 1. If the j-th integer is 1, then the i-th student can attend the lessons on the j-th day of the week. If the j-th integer is 0, then the i-th student cannot attend the lessons on the j-th day of the week.

Additional constraints on the input: for each student, at least one of the days of the week is convenient, the total number of students over all testcases doesn't exceed 10^5.

Output

For each testcase print an answer. If it's possible to divide the students into two groups of equal sizes and choose different days for the groups so each student can attend the lesson in the chosen day of their group, print "YES" (without quotes). Otherwise, print "NO" (without quotes).

Example

input
2 4 1 0 0 1 0 0 1 0 0 1 0 0 0 1 0 0 1 0 1 0 2 0 0 0 1 0 0 0 0 1 0
output
YES NO

Note

In the first testcase, there is a way to meet all the constraints. For example, the first group can consist of the first and the third students, they will attend the lessons on Thursday (the fourth day); the second group can consist of the second and the fourth students, and they will attend the lessons on Tuesday (the second day).

In the second testcase, it is impossible to divide the students into groups so they attend the lessons on different days.

C. Delete Two Elements

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Monocarp has got an array a consisting of n integers. Let's denote k as the mathematic mean of these elements (note that it's possible that k is not an integer).

The mathematic mean of an array of n elements is the sum of elements divided by the number of these elements (i. e. sum divided by n).

Monocarp wants to delete exactly two elements from a so that the mathematic mean of the remaining $(n-2)$ elements is still equal to k.

Your task is to calculate the number of pairs of positions $[i, j]$ ($i < j$) such that if the elements on these positions are deleted, the mathematic mean of $(n-2)$ remaining elements is equal to k (that is, it is equal to the mathematic mean of n elements of the original array a).

Input

The first line contains a single integer t ($1 \le t \le 10^4$) — the number of testcases.

The first line of each testcase contains one integer n ($3 \le n \le 2 \cdot 10^5$) — the number of elements in the array.

The second line contains a sequence of integers $a_1, a_2, \ldots, a_n$ ($0 \le a_i \le 10^9$), where a_i is the i-th element of the array.

The sum of n over all testcases doesn't exceed $2 \cdot 10^5$.

Output
Print one integer — the number of pairs of positions $[i, j]$ ($i < j$) such that if the elements on these positions are deleted, the mathematic mean of $(n - 2)$ remaining elements is equal to k (that is, it is equal to the mathematic mean of n elements of the original array a).

Example

input
4 4 8 8 8 8 3 50 20 10 5 1 4 7 3 5 7 1 2 3 4 5 6 7
output
6 0 2 3

Note
In the first example, any pair of elements can be removed since all of them are equal.

In the second example, there is no way to delete two elements so the mathematic mean doesn't change.

In the third example, it is possible to delete the elements on positions 1 and 3, or the elements on positions 4 and 5.

D. Training Session

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Monocarp is the coach of the Berland State University programming teams. He decided to compose a problemset for a training session for his teams.

Monocarp has n problems that none of his students have seen yet. The i-th problem has a topic a_i (an integer from 1 to n) and a difficulty b_i (an integer from 1 to n). All problems are different, that is, there are no two tasks that have the same topic and difficulty at the same time.

Monocarp decided to select exactly 3 problems from n problems for the problemset. The problems should satisfy **at least one** of two conditions (possibly, both):

- the topics of all three selected problems are different;
- the difficulties of all three selected problems are different.

Your task is to determine the number of ways to select three problems for the problemset.

Input
The first line contains a single integer t ($1 \le t \le 50000$) — the number of testcases.

The first line of each testcase contains an integer n ($3 \le n \le 2 \cdot 10^5$) — the number of problems that Monocarp have.

In the i-th of the following n lines, there are two integers a_i and b_i ($1 \le a_i, b_i \le n$) — the topic and the difficulty of the i-th problem.

It is guaranteed that there are no two problems that have the same topic and difficulty at the same time.

The sum of n over all testcases doesn't exceed $2 \cdot 10^5$.

Output
Print the number of ways to select three training problems that meet either of the requirements described in the statement.

Example

input
2 4 2 4 3 4

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

output

```
3
10
```

Note

In the first example, you can take the following sets of three problems:

- problems 1, 2, 4;
- problems 1, 3, 4;
- problems 2, 3, 4.

Thus, the number of ways is equal to three.

E. Staircases

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

You are given a matrix, consisting of n rows and m columns. The rows are numbered top to bottom, the columns are numbered left to right.

Each cell of the matrix can be either free or locked.

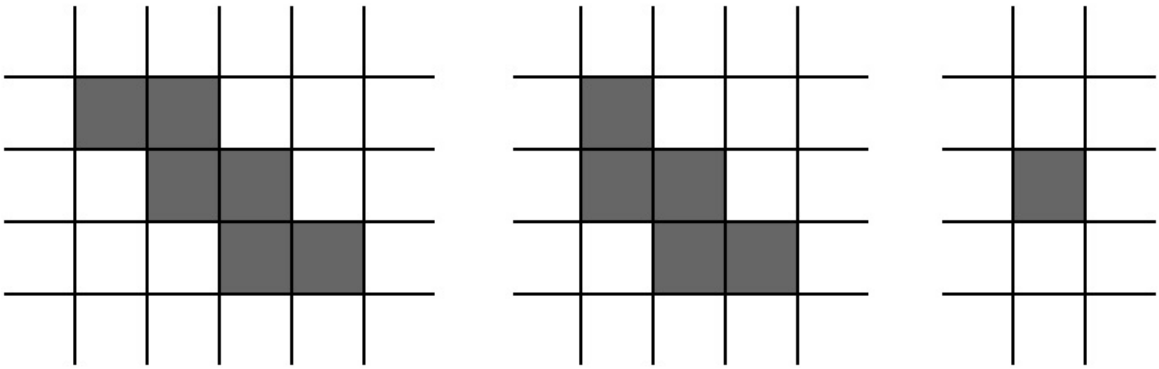
Let's call a path in the matrix *a staircase* if it:

- starts and ends in the free cell;
- visits only free cells;
- has one of the two following structures:

  1. the second cell is 1 to the right from the first one, the third cell is 1 to the bottom from the second one, the fourth cell is 1 to the right from the third one, and so on;
  2. the second cell is 1 to the bottom from the first one, the third cell is 1 to the right from the second one, the fourth cell is 1 to the bottom from the third one, and so on.

In particular, a path, consisting of a single cell, is considered to be a staircase.

Here are some examples of staircases:



Initially all the cells of the matrix are **free**.

You have to process q queries, each of them flips the state of a single cell. So, if a cell is currently free, it makes it locked, and if a cell is currently locked, it makes it free.

Print the number of different staircases after each query. Two staircases are considered different if there exists such a cell that appears in one path and doesn't appear in the other path.

Input

The first line contains three integers n, m and q ($1 \le n, m \le 1000$; $1 \le q \le 10^4$) — the sizes of the matrix and the number of queries.

Each of the next q lines contains two integers x and y ($1 \le x \le n$; $1 \le y \le m$) — the description of each query.

Output

Print q integers — the i-th value should be equal to the number of different staircases after i queries. Two staircases are considered different if there exists such a cell that appears in one path and doesn't appear in the other path.

Examples

input
2 2 8 1 1 1 1 1 1 2 2 1 1 1 2 2 1 1 1
output
5 10 5 2 5 3 1 0

input
3 4 10 1 4 1 2 2 3 1 2 2 3 3 2 1 3 3 4 1 3 3 1
output
49 35 24 29 49 39 31 23 29 27

input
1000 1000 2 239 634 239 634
output
1332632508 1333333000

F. RBS

time limit per test: 3 seconds
memory limit per test: 512 megabytes
input: standard input
output: standard output

A bracket sequence is a string containing only characters "(" and ")". A regular bracket sequence (or, shortly, an RBS) is a bracket sequence that can be transformed into a correct arithmetic expression by inserting characters "1" and "+" between the original characters of the sequence. For example:

- bracket sequences "()()" and "(())" are regular (the resulting expressions are: "(1)+(1)" and "((1+1)+1)");
- bracket sequences ")(", "(" and ")" are not.

Let's denote the concatenation of two strings x and y as $x + y$. For example, "()()" + ")(" = "()())(".

You are given n bracket sequences $s_1, s_2, \ldots, s_n$. You can rearrange them in any order (you can rearrange only the strings themselves, but not the characters in them).

Your task is to rearrange the strings in such a way that the string $s_1 + s_2 + \cdots + s_n$ has as many non-empty prefixes that are RBS as possible.

Input

The first line contains a single integer n ($1 \le n \le 20$).

Then n lines follow, the i-th of them contains s_i — a bracket sequence (a string consisting of characters "(" and/or ")". All

sequences s_i are non-empty, their total length does not exceed $4 \cdot 10^5$.

Output
Print one integer — the maximum number of non-empty prefixes that are RBS for the string $s_1 + s_2 + \cdots + s_n$, if the strings $s_1, s_2, \ldots, s_n$ can be rearranged arbitrarily.

Examples

input
2 ( )
output
1

input
4 ()()()) (( )
output
4

input
1 (())
output
1

input
1)(()
output
0

Note
In the first example, you can concatenate the strings as follows: "(" + ")" = "()", the resulting string will have one prefix, that is an RBS: "()".

In the second example, you can concatenate the strings as follows: "(" + ")" + "()()())" + "(" = "()()()()()(", the resulting string will have four prefixes that are RBS: "()", "()()", "()()()", "()()()()".

The third and the fourth examples contain only one string each, so the order is fixed.

G. The Sum of Good Numbers

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Let's call a positive integer *good* if there is no digit 0 in its decimal representation.

For an array of a *good* numbers a, one found out that the sum of some two neighboring elements is equal to x (i.e. $x = a_i + a_{i+1}$ for some i). x had turned out to be a *good* number as well.

Then the elements of the array a were written out one after another without separators into one string s. For example, if $a = [12, 5, 6, 133]$, then $s = 1256133$.

You are given a string s and a number x. Your task is to determine the positions in the string that correspond to the adjacent elements of the array that have sum x. If there are several possible answers, you can print any of them.

Input
The first line contains the string s ($2 \le |s| \le 5 \cdot 10^5$).

The second line contains an integer x ($2 \le x < 10^{200000}$).

An additional constraint on the input: the answer always exists, i.e you can always select two adjacent substrings of the string s so that if you convert these substrings to integers, their sum is equal to x.

Output
In the first line, print two integers l_1, r_1, meaning that the first term of the sum (a_i) is in the string s from position l_1 to position r_1.

In the second line, print two integers l_2, r_2, meaning that the second term of the sum (a_{i+1}) is in the string s from position l_2 to position r_2.

Examples

input
1256133 17
output
1 2 3 3

input
9544715561 525
output
2 3 4 6

input
239923 5
output
1 1 2 2

input
1218633757639 976272
output
2 7 8 13

Note

In the first example $s[1;2] = 12$ and $s[3;3] = 5$, $12 + 5 = 17$.

In the second example $s[2;3] = 54$ and $s[4;6] = 471$, $54 + 471 = 525$.

In the third example $s[1;1] = 2$ and $s[2;2] = 3$, $2 + 3 = 5$.

In the fourth example $s[2;7] = 218633$ and $s[8;13] = 757639$, $218633 + 757639 = 976272$.