A. X Axis

Input file: standard input
Output file: standard output
Time limit: 2 seconds
Memory limit: 256 megabytes

You are given three points with integer coordinates x_1, x_2, and x_3 on the X axis ($1 \le x_i \le 10$). You can choose any point with an integer coordinate a on the X axis. Note that the point a may coincide with x_1, x_2, or x_3. Let $f(a)$ be the total distance from the given points to the point a. Find the smallest value of $f(a)$.

The distance between points a and b is equal to $|a - b|$. For example, the distance between points $a = 5$ and $b = 2$ is 3.

Input

Each test consists of multiple test cases. The first line contains a single integer t ($1 \le t \le 10^3$) — the number of test cases. Then follows their descriptions.

The single line of each test case contains three integers x_1, x_2, and x_3 ($1 \le x_i \le 10$) — the coordinates of the points.

Output

For each test case, output the smallest value of $f(a)$.

Standard Input	Standard Output
8 1 1 1 1 5 9 8 2 8 10 9 3 2 1 1 2 4 1 7 3 5 1 9 4	0 8 6 7 1 3 4 8

Note

In the first test case, the smallest value of $f(a)$ is achieved when $a = 1$:
$f(1) = |1 - 1| + |1 - 1| + |1 - 1| = 0$.

In the second test case, the smallest value of $f(a)$ is achieved when $a = 5$:
$f(5) = |1 - 5| + |5 - 5| + |9 - 5| = 8$.

In the third test case, the smallest value of $f(a)$ is achieved when $a = 8$:
$f(8) = |8 - 8| + |2 - 8| + |8 - 8| = 6$.

In the fourth test case, the smallest value of $f(a)$ is achieved when $a = 9$:
$f(10) = |10 - 9| + |9 - 9| + |3 - 9| = 7$.

,

B. Matrix Stabilization

Input file: standard input
Output file: standard output
Time limit: 2 seconds
Memory limit: 256 megabytes

You are given a matrix of size $n \times m$, where the rows are numbered from 1 to n from top to bottom, and the columns are numbered from 1 to m from left to right. The element at the intersection of the i-th row and the j-th column is denoted by a_{ij}.

Consider the algorithm for stabilizing matrix a:

1. Find the cell (i, j) such that its value is strictly greater than the values of all its neighboring cells. If there is no such cell, terminate the algorithm. If there are multiple such cells, choose the cell with the smallest value of i, and if there are still multiple cells, choose the one with the smallest value of j.
2. Set $a_{ij} = a_{ij} - 1$.
3. Go to step 1.

In this problem, cells (a, b) and (c, d) are considered neighbors if they share a common side, i.e., $|a - c| + |b - d| = 1$.

Your task is to output the matrix a after the stabilization algorithm has been executed. It can be shown that this algorithm cannot run for an infinite number of iterations.

Input

Each test consists of multiple sets of input data. The first line contains a single integer t ($1 \le t \le 10^4$) — the number of sets of input data. This is followed by their description.

The first line of each set of input data contains two integers n and m ($1 \le n, m \le 100, n \cdot m > 1$) — the number of rows and columns of matrix a.

The next n lines describe the corresponding rows of the matrix. The i-th line contains m integers $a_{i1}, a_{i2}, \ldots, a_{im}$ ($1 \le a_{ij} \le 10^9$).

It is guaranteed that the sum of $n \cdot m$ over all sets of input data does not exceed $2 \cdot 10^5$.

Output

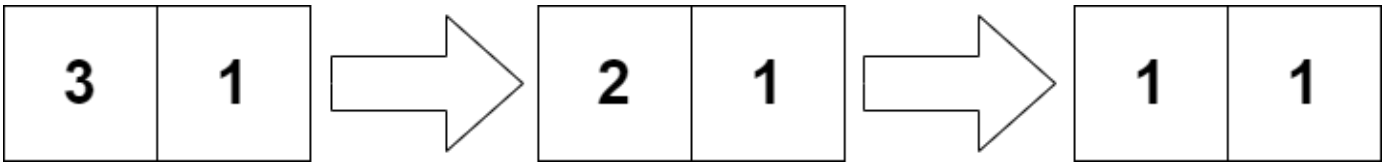
For each set of input data, output n lines with m numbers in each line — the values of the cells of matrix a after the stabilization algorithm.

Standard Input	Standard Output
6 1 2 3 1 2 1 1 1 2 2 1 2 3 4 2 3	1 1 1 1 1 2 3 3 4 4 5 1 8 8 74 74 31 31 74 74 17 7 31 17 17 3

7 4 5 1 8 10 5 4 92 74 31 74 74 92 17 7 31 17 92 3 74 7 3 92 7 31 1 1 3 3 1000000000 1 1000000000 1 1000000000 1 1000000000 1 1000000000	31 7 3 3 7 7 1 1 1 1 1 1 1 1 1 1 1

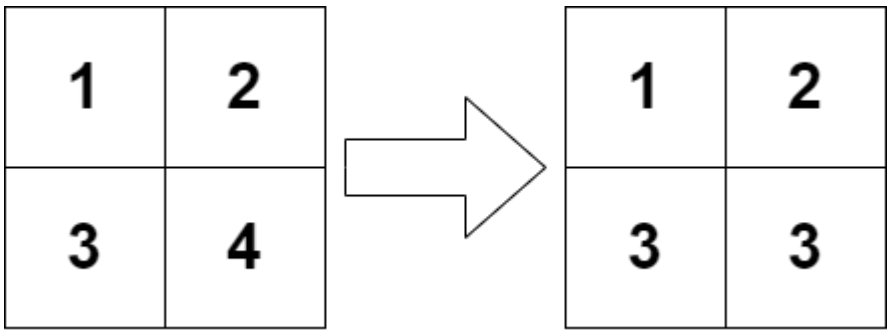
Note

In the first set of input data, the algorithm will select the cell $(1, 1)$ twice in a row and then terminate.

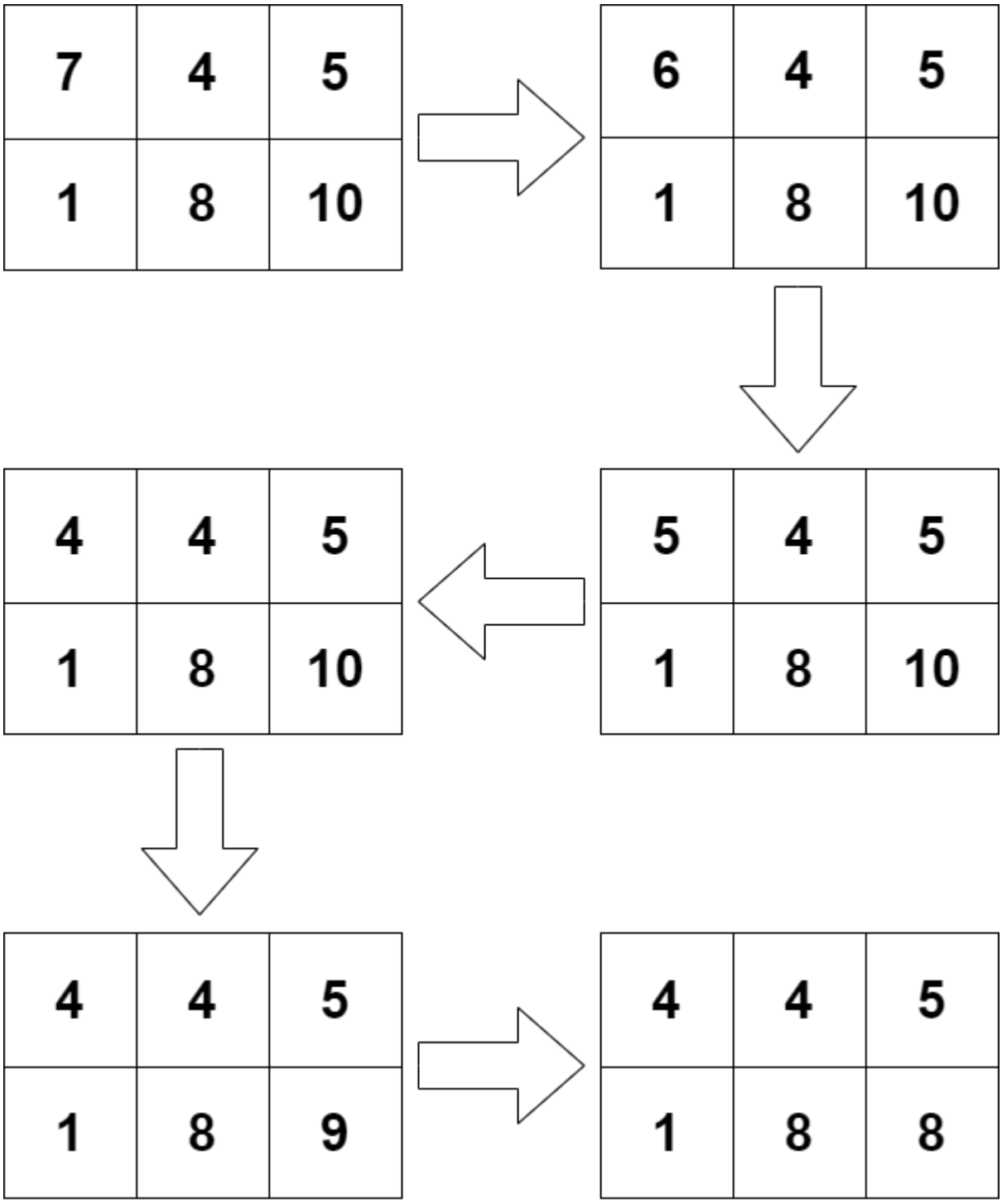


In the second set of input data, there is no cell whose value is strictly greater than the values of all neighboring cells.

In the third set of input data, the algorithm will select the cell $(2, 2)$ and then terminate.



In the fourth set of input data, the algorithm will select the cell $(1, 1)$ three times and then the cell $(2, 3)$ twice.


7 4 5
1 8 10
6 4 5
1 8 10
5 4 5
1 8 10
4 4 5
1 8 10
4 4 5
1 8 9
4 4 5
1 8 8

,

C. Update Queries

Input file: standard input
Output file: standard output
Time limit: 2 seconds
Memory limit: 256 megabytes

Let's consider the following simple problem. You are given a string s of length n, consisting of lowercase Latin letters, as well as an array of indices ind of length m ($1 \le ind_i \le n$) and a string c of length m, consisting of lowercase Latin letters. Then, in order, you perform the update operations, namely, during the i-th operation, you set $s_{ind_i} = c_i$. Note that you perform all m operations from the first to the last.

Of course, if you change the order of indices in the array ind and/or the order of letters in the string c, you can get different results. Find the lexicographically smallest string s that can be obtained after m update operations, if you can rearrange the indices in the array ind and the letters in the string c as you like.

A string a is lexicographically less than a string b if and only if one of the following conditions is met:

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

Input

Each test consists of several sets of input data. The first line contains a single integer t ($1 \le t \le 10^4$) — the number of sets of input data. Then follows their description.

The first line of each set of input data contains two integers n and m ($1 \le n, m \le 10^5$) — the length of the string s and the number of updates.

The second line of each set of input data contains a string s of length n, consisting of lowercase Latin letters.

The third line of each set of input data contains m integers $ind_1, ind_2, \ldots, ind_m$ ($1 \le ind_i \le n$) — the array of indices ind.

The fourth line of each set of input data contains a string c of length m, consisting of lowercase Latin letters.

It is guaranteed that the sum of n over all sets of input data does not exceed $2 \cdot 10^5$. Similarly, the sum of m over all sets of input data does not exceed $2 \cdot 10^5$.

Output

For each set of input data, output the lexicographically smallest string s that can be obtained by rearranging the indices in the array ind and the letters in the string c as you like.

Standard Input	Standard Output
4 1 2 a 1 1 cb 4 4 meow	b cwoz abdcmbn ccdeefo

1 2 1 4 zcwz 7 4 abacaba 1 3 5 7 damn 7 10 traktor 7 6 5 4 3 2 1 6 4 2 codeforces	

Note

In the first set of input data, you can leave the array ind and the string c unchanged and simply perform all operations in that order.

In the second set of input data, you can set the array $ind = [1, 1, 4, 2]$ and $c =$ "zczw". Then the string s will change as follows: $meow \rightarrow zeow \rightarrow ceow \rightarrow ceoz \rightarrow cwoz$.

In the third set of input data, you can leave the array ind unchanged and set $c =$ "admn". Then the string s will change as follows: $abacaba \rightarrow abacaba \rightarrow abdcaba \rightarrow abdcmba \rightarrow abdcmbn$.

,

D. Mathematical Problem

Input file: standard input
Output file: standard output
Time limit: 2 seconds
Memory limit: 256 megabytes

You are given a string s of length $n > 1$, consisting of digits from 0 to 9. You must insert exactly $n - 2$ symbols $+$ (addition) or $\times$ (multiplication) into this string to form a valid arithmetic expression.

In this problem, the symbols cannot be placed before the first or after the last character of the string s, and two symbols cannot be written consecutively. Also, note that the order of the digits in the string cannot be changed. Let's consider $s = 987009$:

- From this string, the following arithmetic expressions can be obtained: $9 \times 8 + 70 \times 0 + 9 = 81$, $98 \times 7 \times 0 + 0 \times 9 = 0$, $9 + 8 + 7 + 0 + 09 = 9 + 8 + 7 + 0 + 9 = 33$. Note that the number 09 is considered valid and is simply transformed into 9.
- From this string, the following arithmetic expressions cannot be obtained: $+9 \times 8 \times 70 + 09$ (symbols should only be placed between digits), $98 \times 70 + 0 + 9$ (since there are 6 digits, there must be exactly 4 symbols).

The result of the arithmetic expression is calculated according to the rules of mathematics — first all multiplication operations are performed, then addition. You need to find the minimum result that can be obtained by inserting exactly $n - 2$ addition or multiplication symbols into the given string s.

Input

Each test consists of multiple test cases. The first line contains a single integer t ($1 \le t \le 10^4$) — the number of test cases. Then follows their description.

The first line of each test case contains a single integer n ($2 \le n \le 20$) — the length of the string s.

The second line of each test case contains a string s of length n, consisting of digits from 0 to 9.

Output

For each test case, output the minimum result of the arithmetic expression that can be obtained by inserting exactly $n - 2$ addition or multiplication symbols into the given string.

Standard Input	Standard Output
18 2 10 2 74 2 00 2 01 3 901 3 101	10 74 0 1 9 1 19 0 11 261 0 0 0

5 23311 6 987009 7 1111111 20 99999999999999999999 20 00000000000000000000 4 0212 18 057235283621345395 4 1112 20 19811678487321784121 4 1121 4 2221 3 011	12 93 12 24 0

Note

In the first four test cases, we cannot add symbols, so the answer will be the original number.

In the fifth test case, the optimal answer looks as follows: $9 \times 01 = 9 \times 1 = 9$.

In the sixth test case, the optimal answer looks as follows: $1 \times 01 = 1 \times 1 = 1$.

In the seventh test case, the optimal answer looks as follows: $2 + 3 + 3 + 11 = 19$.

In the eighth test case, one of the optimal answers looks as follows: $98 \times 7 \times 0 + 0 \times 9 = 0$.

,

E. Beautiful Array

Input file: standard input
Output file: standard output
Time limit: 2 seconds
Memory limit: 256 megabytes

You are given an array of integers $a_1, a_2, \ldots, a_n$ and an integer k. You need to make it beautiful with the least amount of operations.

Before applying operations, you can shuffle the array elements as you like. For one operation, you can do the following:

- Choose an index $1 \leq i \leq n$,
- Make $a_i = a_i + k$.

The array $b_1, b_2, \ldots, b_n$ is beautiful if $b_i = b_{n-i+1}$ for all $1 \leq i \leq n$.

Find the minimum number of operations needed to make the array beautiful, or report that it is impossible.

Input

Each test consists of several sets of input data. The first line contains a single integer t ($1 \leq t \leq 10^4$) — the number of sets of input data. Then follows their description.

The first line of each set of input data contains two integers n and k ($1 \leq n \leq 10^5$, $1 \leq k \leq 10^9$) — the size of the array a and the number k from the problem statement.

The second line of each set of input data contains n integers $a_1, a_2, \ldots, a_n$ ($1 \leq a_i \leq 10^9$) — the elements of the array a.

It is guaranteed that the sum of n over all sets of input data does not exceed $2 \cdot 10^5$.

Output

For each set of input data, output the minimum number of operations needed to make the array beautiful, or -1 if it is impossible.

Standard Input	Standard Output
11 1 1000000000 1 2 1 624323799 708290323 3 1 3 2 1 4 1 7 1 5 3 5 1 11 2 15 7 10 7 1 1 8 2 16 8 16 31 13 1 2 1 1 3 3 11 12 22 45 777 777 1500 74	0 83966524 1 4 6 1 48 -1 0 14 0

10 2 1 2 1 2 1 2 1 2 1 2 11 2 1 2 1 2 1 2 1 2 1 2 1 13 3 2 3 9 14 17 10 22 20 18 30 1 4 28 5 1 2 3 5 3 5	

Note

In the first set of input data, the array is already beautiful.

In the second set of input data, you can shuffle the array before the operations and perform the operation with index $i = 1$ for 83966524 times.

In the third set of input data, you can shuffle the array a and make it equal to $[2, 3, 1]$. Then apply the operation with index $i = 3$ to get the array $[2, 3, 2]$, which is beautiful.

In the eighth set of input data, there is no set of operations and no way to shuffle the elements to make the array beautiful.

In the ninth set of input data, the array is already beautiful.

,

F. Non-academic Problem

Input file: standard input
Output file: standard output
Time limit: 2 seconds
Memory limit: 256 megabytes

You are given a connected undirected graph, the vertices of which are numbered with integers from 1 to n. Your task is to minimize the number of pairs of vertices $1 \leq u < v \leq n$ between which there exists a path in this graph. To achieve this, you can remove exactly one edge from the graph.

Find the smallest number of pairs of vertices!

Input

Each test consists of several sets of input data. The first line contains a single integer t ($1 \leq t \leq 10^4$) — the number of sets of input data. Then follows their description.

The first line of each set of input data contains two integers n and m ($2 \leq n \leq 10^5$, $n - 1 \leq m \leq \min(10^5, \frac{n \cdot (n-1)}{2})$) — the number of vertices in the graph and the number of edges.

Each of the next m lines contains two integers u and v ($1 \leq u, v \leq n, u \neq v$), indicating that there is an undirected edge in the graph between vertices u and v.

It is guaranteed that the given graph is connected and without multiple edges.

It is guaranteed that the sum of n and the sum of m over all sets of input data does not exceed $2 \cdot 10^5$.

Output

For each set of input data, output the smallest number of pairs of reachable vertices, if exactly one edge can be removed.

Standard Input	Standard Output
6 2 1 1 2 3 3 1 2 2 3 1 3 5 5 1 2 1 3 3 4 4 5 5 3 6 7 1 2 1 3 2 3 3 4 4 5	0 3 4 6 6 21

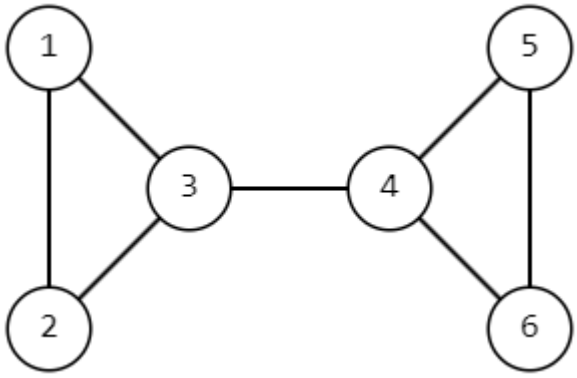
4 6 5 6 5 5 1 2 1 3 2 3 2 4 3 5 10 12 1 2 1 3 2 3 2 4 4 5 5 6 6 7 7 4 3 8 8 9 9 10 10 8	

Note

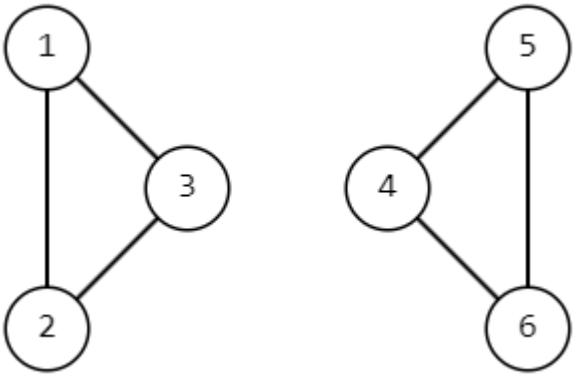
In the first set of input data, we will remove the single edge $(1, 2)$ and the only pair of vertices $(1, 2)$ will become unreachable from each other.

In the second set of input data, no matter which edge we remove, all vertices will be reachable from each other.

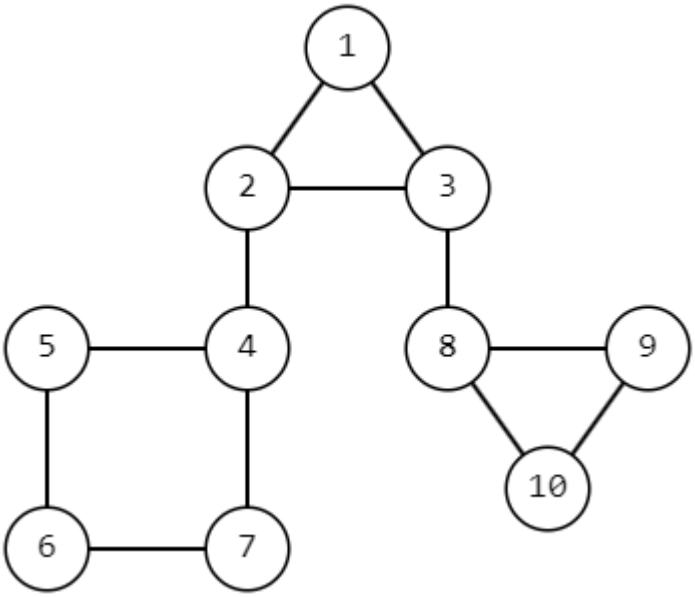
In the fourth set of input data, the graph looks like this initially.



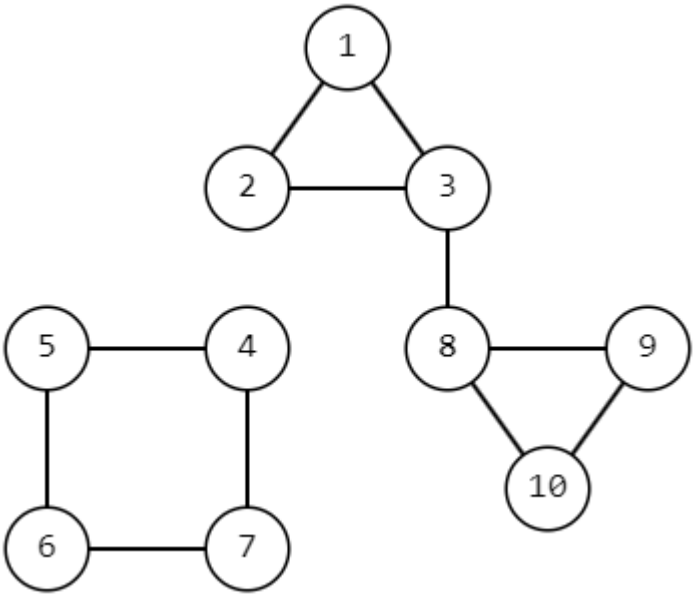
We will remove the edge $(3, 4)$ and then the only reachable pairs of vertices will be $(1, 2)$, $(1, 3)$, $(2, 3)$, $(4, 5)$, $(4, 6)$, $(5, 6)$.



In the sixth set of input data, the graph looks like this initially.



After removing the edge $(2, 4)$, the graph will look like this. Thus, there will be 21 pairs of reachable vertices.



,

G1. Permutation Problem (Simple Version)

```
Input file:     standard input
Output file:    standard output
Time limit:     3 seconds
Memory limit:   128 megabytes
```

This is a simple version of the problem. The only difference is that in this version $n \leq 10^5$ and the sum of n for all sets of input data does not exceed 10^5.

You are given a permutation p of length n. Calculate the number of index pairs $1 \leq i < j \leq n$ such that $p_i \cdot p_j$ is divisible by $i \cdot j$ without remainder.

A permutation is a sequence of n integers, where each integer from 1 to n occurs exactly once. For example, $[1]$, $[3, 5, 2, 1, 4]$, $[1, 3, 2]$ are permutations, while $[2, 3, 2]$, $[4, 3, 1]$, $[0]$ are not.

Input

Each test consists of multiple sets of input data. The first line contains a single integer t ($1 \leq t \leq 10^4$) — the number of sets of input data. Then follows their description.

The first line of each set of input data contains a single integer n ($1 \leq n \leq 10^5$) — the length of the permutation p.

The second line of each set of input data contains n distinct integers $p_1, p_2, \ldots, p_n$ ($1 \leq p_i \leq n$) — the permutation p.

It is guaranteed that the sum of n for all sets of input data does not exceed 10^5.

Output

For each set of input data, output the number of index pairs $1 \leq i < j \leq n$ such that $p_i \cdot p_j$ is divisible by $i \cdot j$ without remainder.

Standard Input	Standard Output
6 1 1 2 1 2 3 2 3 1 5 2 4 1 3 5 12 8 9 7 12 1 10 6 3 2 4 11 5 15 1 2 4 6 8 10 12 14 3 9 15 5 7 11 13	0 1 1 3 9 3

Note

In the first set of input data, there are no index pairs, as the size of the permutation is 1.

In the second set of input data, there is one index pair $(1, 2)$ and it is valid.

In the third set of input data, the index pair $(1, 2)$ is valid.

In the fourth set of input data, the index pairs $(1, 2)$, $(1, 5)$, and $(2, 5)$ are valid.

,

G2. Permutation Problem (Hard Version)

Input file: standard input
Output file: standard output
Time limit: 3 seconds
Memory limit: 128 megabytes

This is the hard version of the problem. The only difference is that in this version $n \leq 5 \cdot 10^5$ and the sum of n for all sets of input data does not exceed $5 \cdot 10^5$.

You are given a permutation p of length n. Calculate the number of index pairs $1 \leq i < j \leq n$ such that $p_i \cdot p_j$ is divisible by $i \cdot j$ without remainder.

A permutation is a sequence of n integers, in which each integer from 1 to n occurs exactly once. For example, $[1]$, $[3, 5, 2, 1, 4]$, $[1, 3, 2]$ are permutations, while $[2, 3, 2]$, $[4, 3, 1]$, $[0]$ are not.

Input

Each test consists of several sets of input data. The first line contains a single integer t ($1 \leq t \leq 10^4$) — the number of sets of input data. Then follows their description.

The first line of each set of input data contains a single integer n ($1 \leq n \leq 5 \cdot 10^5$) — the length of the permutation p.

The second line of each set of input data contains n distinct integers $p_1, p_2, \ldots, p_n$ ($1 \leq p_i \leq n$) — the permutation p.

It is guaranteed that the sum of n for all sets of input data does not exceed $5 \cdot 10^5$.

Output

For each set of input data, output the number of index pairs $1 \leq i < j \leq n$ such that $p_i \cdot p_j$ is divisible by $i \cdot j$ without remainder.

Standard Input	Standard Output
6 1 1 2 1 2 3 2 3 1 5 2 4 1 3 5 12 8 9 7 12 1 10 6 3 2 4 11 5 15 1 2 4 6 8 10 12 14 3 9 15 5 7 11 13	0 1 1 3 9 3

Note

In the first set of input data, there are no index pairs, as the size of the permutation is 1.

In the second set of input data, there is one index pair $(1, 2)$ and it is valid.

In the third set of input data, the index pair $(1, 2)$ is valid.

In the fourth set of input data, the index pairs $(1, 2)$, $(1, 5)$, and $(2, 5)$ are valid.