Codeforces Round #755 (Div. 1, based on Technocup 2022 Elimination Round 2)

A. Two Arrays

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

You are given two arrays of integers $a_1, a_2, \ldots, a_n$ and $b_1, b_2, \ldots, b_n$.

Let's define a transformation of the array a:

1. Choose any non-negative integer k such that $0 \le k \le n$.
2. Choose k distinct array indices $1 \le i_1 < i_2 < \ldots < i_k \le n$.
3. Add 1 to each of $a_{i_1}, a_{i_2}, \ldots, a_{i_k}$, all other elements of array a remain unchanged.
4. Permute the elements of array a in any order.

Is it possible to perform some transformation of the array a **exactly once**, so that the resulting array is equal to b?

Input

The first line contains a single integer t ($1 \le t \le 100$) — the number of test cases. Descriptions of test cases follow.

The first line of each test case contains a single integer n ($1 \le n \le 100$) — the size of arrays a and b.

The second line of each test case contains n integers $a_1, a_2, \ldots, a_n$ ($-100 \le a_i \le 100$).

The third line of each test case contains n integers $b_1, b_2, \ldots, b_n$ ($-100 \le b_i \le 100$).

Output

For each test case, print "YES" (without quotes) if it is possible to perform a transformation of the array a, so that the resulting array is equal to b. Print "NO" (without quotes) otherwise.

You can print each letter in any case (upper or lower).

Example

input
3 3 -1 1 0 0 0 2 1 0 2 5 1 2 3 4 5 1 2 3 4 5
output
YES NO YES

Note

In the first test case, we can make the following transformation:

- Choose $k = 2$.
- Choose $i_1 = 1$, $i_2 = 2$.
- Add 1 to a_1 and a_2. The resulting array is $[0, 2, 0]$.
- Swap the elements on the second and third positions.

In the second test case there is no suitable transformation.

In the third test case we choose $k = 0$ and do not change the order of elements.

B. Guess the Permutation

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

This is an interactive problem.

Jury initially had a sequence a of length n, such that $a_i = i$.

The jury chose three integers i, j, k, such that $1 \le i < j < k \le n$, $j - i > 1$. After that, Jury reversed subsegments $[i, j-1]$ and $[j, k]$ of the sequence a.

Reversing a subsegment $[l, r]$ of the sequence a means reversing the order of elements $a_l, a_{l+1}, \ldots, a_r$ in the sequence, i. e. a_l is swapped with a_r, a_{l+1} is swapped with a_{r-1}, etc.

You are given the number n and you should find i, j, k after asking some questions.

In one question you can choose two integers l and r ($1 \le l \le r \le n$) and ask the number of inversions on the subsegment $[l, r]$ of the sequence a. You will be given the number of pairs (i, j) such that $l \le i < j \le r$, and $a_i > a_j$.

Find the chosen numbers i, j, k after at most 40 questions.

The numbers i, j, and k are fixed before the start of your program and do not depend on your queries.

Input

Each test consists of multiple test cases. The first line contains a single integer t ($1 \le t \le 100$) — the number of test cases. Description of the test cases follows.

The single line of each test case contains a single integer n ($4 \le n \le 10^9$). After reading it you should start an interaction process by asking questions for that test case. After giving an answer you should:

- Terminate your program if that is the last test case.
- Proceed to the next test case otherwise.

Interaction

To ask number of inversions on a subsegment $[l, r]$, print `"? l r"`, where ($1 \le l \le r \le n$). You can ask at most 40 questions in each test case. As a result you should read a single integer x.

- If $x = -1$, your program made an invalid question or you exceeded the number of questions for that test case. Your program should terminate immediately (otherwise it can get any verdict instead of `"Wrong Answer"`).
- Otherwise x is equal to the number of inversions on the subsegment $[l, r]$ of the sequence a.

To give the answer, print `"! i j k"`, where i, j, k are the numbers you found. You should continue solving the next test cases or terminate the program after that.

After printing a question or an answer do not forget to print the end of line and flush the output. Otherwise, you will get an `"Idleness limit exceeded"` verdict. To do this, use:

- `fflush(stdout)` or `cout.flush()` in C++;
- `System.out.flush()` in Java;
- `flush(output)` in Pascal;
- `stdout.flush()` in Python;
- See the documentation for other languages.

Hacks

To make a hack, use the following format:

The first line contains a single integer t ($1 \le t \le 100$) — the number of test cases.

Each of the next t lines contains four integers n, i, j, k ($4 \le n \le 10^9$, $1 \le i < j < k \le n$, $j - i > 1$).

Example

input
2 5 4 3 3 5 2 2 1
output
  ? 1 5 ? 2 5

```
? 3 5

! 1 3 5

? 1 5

? 2 5

? 3 5

! 2 4 5
```

Note

In the first test case, $i = 1$, $j = 3$, $k = 5$, so the sequence a is $[2, 1, 5, 4, 3]$.

In the second test case, $i = 2$, $j = 4$, $k = 5$, so the sequence a is $[1, 3, 2, 5, 4]$.

C. Game with Stones

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

Bob decided to take a break from calculus homework and designed a game for himself.

The game is played on a sequence of piles of stones, which can be described with a sequence of integers $s_1, \ldots, s_k$, where s_i is the number of stones in the i-th pile. On each turn, Bob picks a pair of non-empty adjacent piles i and $i + 1$ and takes one stone from each. If a pile becomes empty, its adjacent piles **do not become adjacent**. The game ends when Bob can't make turns anymore. Bob considers himself a winner if at the end all piles are empty.

We consider a sequence of piles **winning** if Bob can start with it and win with some sequence of moves.

You are given a sequence $a_1, \ldots, a_n$, count the number of subsegments of a that describe a winning sequence of piles. In other words find the number of segments $[l, r]$ $(1 \leq l \leq r \leq n)$, such that the sequence $a_l, a_{l+1}, \ldots, a_r$ is winning.

Input

Each test consists of multiple test cases. The first line contains a single integer t $(1 \leq t \leq 3 \cdot 10^5)$ — the number of test cases. Description of the test cases follows.

The first line of each test case contains a single integer n $(1 \leq n \leq 3 \cdot 10^5)$.

The second line of each test case contains n integers $a_1, a_2, \ldots, a_n$ $(0 \leq a_i \leq 10^9)$.

It is guaranteed that the sum of n over all test cases does not exceed $3 \cdot 10^5$.

Output

Print a single integer for each test case — the answer to the problem.

Example

input
6 2 2 2 3 1 2 3 4 1 1 1 1 4 1 2 2 1 4 1 2 1 2 8 1 2 1 2 1 2 1 2
output
1 0 4 2 1 3

Note

In the first test case, Bob can't win on subsegments of length 1, as there is no pair of adjacent piles in an array of length 1.

In the second test case, every subsegment is not winning.

In the fourth test case, the subsegment $[1, 4]$ is winning, because Bob can make moves with pairs of adjacent piles: $(2, 3)$, $(1, 2)$, $(3, 4)$. Another winning subsegment is $[2, 3]$.

D. Strange LCS

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

You are given n strings $s_1, s_2, \ldots, s_n$, each consisting of lowercase and uppercase English letters. In addition, it's guaranteed that each character occurs in each string **at most twice**. Find the longest common subsequence of these strings.

A string t is a subsequence of a string s if t can be obtained from s by deletion of several (possibly, zero or all) symbols.

Input

Each test consists of multiple test cases. The first line contains a single integer t ($1 \le t \le 5$) — the number of test cases. Description of the test cases follows.

The first line of each test case contains a single integer n ($2 \le n \le 10$) — the number of strings.

Each of the next n lines contains the corresponding string s_i. Each s_i is non-empty, consists only of uppercase and lowercase English letters, and no character appears more than twice in each string.

Output

For each test case print the answer in two lines:

In the first line print the length of the longest common subsequence.

In the second line print the longest common subsequence. If there are multiple such subsequences, print any of them.

Example

input
4 2 ABC CBA 2 bacab defed 3 abcde aBcDe ace 2 codeforces technocup
output
1 A 0 3 ace 3 coc

Note

In the first test case, the longest common subsequence is "A". There are no common subsequences of length 2.

In the second test case, sets of characters of strings don't intersect, so any non-empty string can't be a common subsequence.

E. Eligible Segments

time limit per test: 3 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

You are given n **distinct** points $p_1, p_2, \ldots, p_n$ on the plane and a positive integer R.

Find the number of pairs of indices (i, j) such that $1 \le i < j \le n$, and for every possible k ($1 \le k \le n$) the distance from the point p_k to the **segment** between points p_i and p_j is at most R.

Input

The first line contains two integers n, R ($1 \le n \le 3000$, $1 \le R \le 10^5$) — the number of points and the maximum distance between a point and a segment.

Each of the next n lines contains two integers x_i, y_i ($-10^5 \le x_i, y_i \le 10^5$) that define the i-th point $p_i = (x_i, y_i)$. All points are distinct.

It is guaranteed that the answer does not change if the parameter R is changed by at most 10^{-2}.

Output
Print the number of suitable pairs (i, j).

Examples

input
4 2 0 1 0 -1 3 0 -3 0
output
1

input
3 3 1 -1 -1 -1 0 1
output
3

Note
In the first example, the only pair of points $(-3, 0)$, $(3, 0)$ is suitable. The distance to the segment between these points from the points $(0, 1)$ and $(0, -1)$ is equal to 1, which is less than $R = 2$.

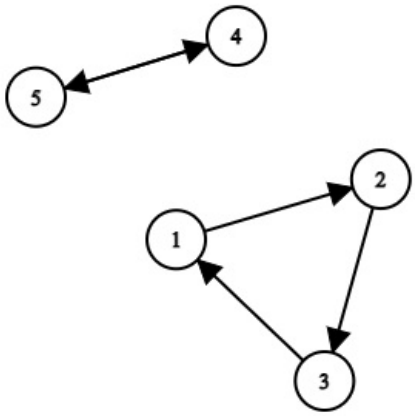
In the second example, all possible pairs of points are eligible.

F. Jumping Through the Array

time limit per test: 8 seconds
memory limit per test: 512 megabytes
input: standard input
output: standard output

You are given an array of integers a of size n and a permutation p of size n. There are q queries of three types coming to you:

1. For given numbers l and r, calculate the sum in array a on the segment from l to r: $\sum_{i=l}^{r} a_i$.
2. You are given two numbers v and x. Let's build a directed graph from the permutation p: it has n vertices and n edges $i \to p_i$. Let C be the set of vertices that are reachable from v in this graph. You should add x to all a_u such that u is in C.
3. You are given indices i and j. You should swap p_i and p_j.



The graph corresponding to the permutation $[2, 3, 1, 5, 4]$.

Please, process all queries and print answers to queries of type 1.

Input
The first line contains a single integer n ($1 \le n \le 2 \cdot 10^5$) — the size of the array and permutation.

The second line contains n integers $a_1, a_2, \ldots, a_n$ ($-10^8 \le a_i \le 10^8$).

The third line contains n distinct integers $p_1, p_2, \ldots, p_n$ ($1 \le p_i \le n$).

The fourth line contains a single integer q — the number of queries ($1 \le q \le 2 \cdot 10^5$).

Next q lines contain description of queries. The i-th of them starts with an integer t_i ($1 \le t_i \le 3$) — the query type.

- If $t_i = 1$, then the i-th line also contains two integers l, r ($1 \le l \le r \le n$).

- If $t_i = 2$, then the i-th line also contains two integers v, x ($1 \le v \le n$, $-10^8 \le x \le 10^8$).
- If $t_i = 3$, then the i-th line also contains also two integers i, j ($1 \le i, j \le n$).

Output

For every first type query, print a single integer — the answer to this query.

Examples

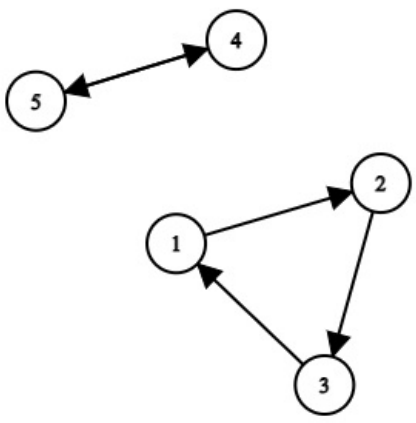
input
5 6 9 -5 3 0 2 3 1 5 4 6 1 1 5 2 1 1 1 1 5 3 1 5 2 1 -1 1 1 5
output
13 16 11

input
8 -15 52 -4 3 5 9 0 5 2 4 6 8 1 3 5 7 10 2 2 2 2 5 -1 1 1 8 1 1 5 1 5 8 3 1 6 2 1 50 1 1 8 2 6 -20 1 1 8
output
61 45 22 461 301

input
1 1 1 1 1 1 1
output
1

Note

In the first example:

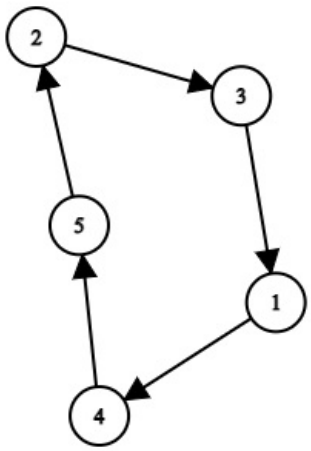


The graph corresponding to the initial permutation.

There are 6 queries.

1. The sum on the segment from 1 to 5 is $a_1 + a_2 + a_3 + a_4 + a_5 = 6 + 9 + (-5) + 3 + 0 = 13$.

2. If we start from 1, we can reach $\{1, 2, 3\}$. After this query a is: $[7, 10, -4, 3, 0]$.
3. The sum on the segment from 1 to 5 is $a_1 + a_2 + a_3 + a_4 + a_5 = 6 + 9 + (-5) + 3 + 0 = 16$.
4. After this query $p = [4, 3, 1, 5, 2]$.



The graph corresponding to the new permutation.

5. If we start from 2, we can reach $\{1, 2, 3, 4, 5\}$. After this query a is: $[6, 9, -5, 2, -1]$.
6. The sum on the segment from 1 to 5 is $a_1 + a_2 + a_3 + a_4 + a_5 = 6 + 9 + (-5) + 2 + (-1) = 11$.
