A. Array Divisibility

```
Input file:     standard input
Output file:    standard output
Time limit:     1 second
Memory limit:   256 megabytes
```

An array of integers $a_1, a_2, \cdots, a_n$ is beautiful subject to an integer k if it satisfies the following:

- The sum of a_j over all j such that j is a multiple of k and $1 \le j \le n$, itself, is a multiple of k.
- More formally, if $\sum_{k|j} a_j$ is divisible by k for all $1 \le j \le n$ then the array a is beautiful subject to k. Here, the notation $k|j$ means k divides j, that is, j is a multiple of k.

Given n, find an array of positive nonzero integers, with each element less than or equal to 10^5 that is beautiful subject to all $1 \le k \le n$.
It can be shown that an answer always exists.

Input

Each test contains multiple test cases. The first line contains the number of test cases t ($1 \le t \le 100$). The description of the test cases follows.

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

Output

For each test case, print the required array as described in the problem statement.

Standard Input	Standard Output

3
3
6
7
	4 22 18
	10 6 15 32 125 54
	23 18 27 36 5 66 7

Note

In the second test case, when $n = 6$, for all integers k such that $1 \le k \le 6$, let S be the set of all indices of the array that are divisible by k.

- When $k = 1$, $S = \{1, 2, 3, 4, 5, 6\}$ meaning $a_1 + a_2 + a_3 + a_4 + a_5 + a_6 = 242$ must be divisible by 1.
- When $k = 2$, $S = \{2, 4, 6\}$ meaning $a_2 + a_4 + a_6 = 92$ must be divisible by 2.
- When $k = 3$, $S = \{3, 6\}$ meaning $a_3 + a_6 = 69$ must divisible by 3.
- When $k = 4$, $S = \{4\}$ meaning $a_4 = 32$ must divisible by 4.
- When $k = 5$, $S = \{5\}$ meaning $a_5 = 125$ must divisible by 5.
- When $k = 6$, $S = \{6\}$ meaning $a_6 = 54$ must divisible by 6.

The array $a = [10, 6, 15, 32, 125, 54]$ satisfies all of the above conditions. Hence, a is a valid array.

,

B. Corner Twist

Input file:	standard input
Output file:	standard output
Time limit:	1 second
Memory limit:	256 megabytes

You are given two grids of numbers a and b, with n rows and m columns. All the values in the grid are 0, 1 or 2.

You can perform the following operation on a any number of times:

- Pick any subrectangle in the grid with length and width ≥ 2. You are allowed to choose the entire grid as a subrectangle.
- The subrectangle has four corners. Take any pair of diagonally opposite corners of the chosen subrectangle and add 1 to their values modulo 3.
- For the pair of corners not picked, add 2 to their values modulo 3.

Note that the operation only changes the values of the corners of the picked subrectangle.

Is it possible to convert the grid a into grid b by applying the above operation any number of times (possibly zero)?

Input

The first line contains an integer t, the number of testcases ($1 \leq t \leq 250$).

For each testcase:

The first line contains two integers n and m, the number of rows and columns in the grid ($2 \leq n, m \leq 500$).

Each of the next n lines contain m characters — the j-th character of the i-th line represents $a_{i,j}$.

Each of the next n lines contain m characters — the j-th character of the i-th line represents $b_{i,j}$ ($0 \leq a_{i,j}, b_{i,j} \leq 2$).

It is guaranteed that the sum of n over all test cases and the sum of m over all test cases do not exceed 500.

Output

For each test case print "YES" (without quotes) if it is possible to convert grid a into grid b and "NO" (without quotes) otherwise.

You can output the answer in any case (upper or lower). For example, the strings "yEs", "yes", "Yes", and "YES" will be recognized as positive responses.

Standard Input	Standard Output

7
3 3
000
000
000
111
111
111
	YES
	YES
	YES
	NO
	YES
	NO
	YES

```
4 4
0000
0000
0000
0000
2100
1200
0012
0021
4 4
1020
1200
1210
0000
0000
1200
2200
0000
3 3
012
012
012
010
111
011
8 8
00000000
00000000
00000000
00000000
00000000
00000000
00000000
10000000
00000000
01200000
02010000
00102000
00020100
00001020
00000210
10000000
2 7
0000000
0000000
2220111
0111222
2 7
0000000
```

0100010 2220111 1210202	

Note

In the first testcase, grid a can be converted into b in the following manner:

$$\begin{matrix} \boxed{0} & 0 & \boxed{0} \\ 0 & 0 & 0 \\ \boxed{0} & 0 & \boxed{0} \end{matrix} \Rightarrow \begin{matrix} 1 & 0 & 2 \\ 0 & \boxed{0} & \boxed{0} \\ 2 & \boxed{0} & \boxed{1} \end{matrix} \Rightarrow \begin{matrix} 1 & 0 & 2 \\ \boxed{0} & \boxed{1} & 2 \\ \boxed{2} & \boxed{2} & 2 \end{matrix} \Rightarrow \begin{matrix} 1 & \boxed{0} & \boxed{2} \\ 1 & 0 & 2 \\ 1 & \boxed{0} & \boxed{2} \end{matrix} \Rightarrow \begin{matrix} 1 & 1 & 1 \\ 1 & \boxed{0} & \boxed{2} \\ 1 & \boxed{2} & \boxed{0} \end{matrix} \Rightarrow \begin{matrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{matrix}$$

Here, in each operation, the top-right and bottom-left corners highlighted by a box are incremented by 2 modulo 3, while the top-left and bottom-right corners are incremented by 1 modulo 3.

In the fourth testcase, it can be proven that it is not possible to convert grid a into grid b using the above-mentioned operations any number of times.

,

C. Have Your Cake and Eat It Too

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

Alice, Bob and Charlie want to share a rectangular cake cut into n pieces. Each person considers every piece to be worth a different value. The i-th piece is considered to be of value a_i by Alice, b_i by Bob and c_i by Charlie.

The sum over all a_i, all b_i and all c_i individually is the same, equal to tot.

Given the values of each piece of the cake for each person, you need to give each person a contiguous slice of cake. In other words, the indices at the left and right ends of these subarrays (the slices given to each person) can be represented as (l_a, r_a), (l_b, r_b) and (l_c, r_c) respectively for Alice, Bob and Charlie. The division needs to satisfy the following constraints:

- No piece is assigned to more than one person, i.e., no two subarrays among $[l_a, \ldots, r_a]$, $[l_b, \ldots, r_b]$ and $[l_c, \ldots, r_c]$ intersect.
- $\sum_{i=l_a}^{r_a} a_i, \sum_{i=l_b}^{r_b} b_i, \sum_{i=l_c}^{r_c} c_i \geq \lceil \frac{tot}{3} \rceil$.

Here, the notation $\lceil \frac{a}{b} \rceil$ represents ceiling division. It is defined as the smallest integer greater than or equal to the exact division of a by b. In other words, it rounds up the division result to the nearest integer. For instance $\lceil \frac{10}{3} \rceil = 4$ and $\lceil \frac{15}{3} \rceil = 5$.

Input

The first line contains an integer t, the number of testcases, ($1 \leq t \leq 10^4$)

For each testcase:

The first line contains the integer n ($3 \leq n \leq 2 \cdot 10^5$).

The following three lines contain n integers each:

One line with n integers $a_1, a_2, \ldots, a_n$ represents the values for Alice ($1 \leq a_i \leq 10^6$).

The next line with n integers $b_1, b_2, \ldots, b_n$ represents the values for Bob ($1 \leq b_i \leq 10^6$).

The next line with n integers $c_1, c_2, \ldots, c_n$ represents the values for Charlie ($1 \leq c_i \leq 10^6$).

It is guaranteed that $\sum_{i=1}^{n} a_i = \sum_{i=1}^{n} b_i = \sum_{i=1}^{n} c_i$.

The sum of n over all testcases does not exceed $2 \cdot 10^5$.

Output

For each testcase, output -1 if the required condition is impossible.

Otherwise, output six numbers – $l_a, r_a, l_b, r_b, l_c, r_c$, the respective starting and ending indices (1-indexed) of the subarrays for Alice, Bob and Charlie, respectively.

Standard Input	Standard Output

```
10
5
5 1 1 1 1
1 1 5 1 1
1 1 1 1 5
6
1 2 3 4 5 6
5 6 1 2 3 4
3 4 5 6 1 2
4
4 4 4 4
4 4 4 4
4 4 4 4
5
5 10 5 2 10
9 6 9 7 1
10 7 10 2 3
3
4 5 2
6 1 4
1 8 2
3
10 4 10
8 7 9
10 4 10
7
57113 65383 19795 53580 74452 3879 23255
12917 16782 89147 93107 27365 15044 43095
33518 63581 33565 34112 46774 44151 41756
6
6 3 1 8 7 1
10 2 6 2 2 4
10 9 2 1 2 2
5
5 5 4 5 5
1 6 3 8 6
2 4 1 9 8
10
1 1 1 1 1001 1 1 1001 1 1
1 1 1 1 1 1 2001 1 1 1
1 1 1 1 1 1001 1 1 1 1001
```

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

Note

In the first testcase, the sum of either of the three arrays is 9. Each person needs a cake slice corresponding to a subarray with a total value of at least $\lceil \frac{9}{3} \rceil = 3$.

If we assign the subarray $(1,1)$ to Alice, its total value to her is 5, which is ≥ 3; the subarray $(2,3)$ to Bob, its total value to him is $1 + 5 = 6$, which is ≥ 3; and the subarray $(4,5)$ to Charlie, its total value to him

$1 + 5 = 6$, which is also ≥ 3. Each person gets their own separate pieces of the cake, and no piece is common to two or more people.

It can be shown that for the third test case, it is not possible to give slices of the cake in a way that satisfies the given constraints.

D. Swap Dilemma

```
Input file:     standard input
Output file:    standard output
Time limit:     1 second
Memory limit:   256 megabytes
```

Given two arrays of distinct positive integers a and b of length n, we would like to make both the arrays the same. Two arrays x and y of length k are said to be the same when for all $1 \le i \le k$, $x_i = y_i$.

Now in one move, you can choose some index l and r in a ($l \le r$) and swap a_l and a_r, then choose some p and q ($p \le q$) in b such that $r - l = q - p$ and swap b_p and b_q.

Is it possible to make both arrays the same?

Input

Each test contains multiple test cases. The first line contains the number of test cases t ($1 \le t \le 2 \cdot 10^4$). The description of the test cases follows.

The first line of each test case contains a single integer n ($1 \le n \le 10^5$) — the length of the arrays a and b.

The second line of each test case contains n distinct integers $a_1, a_2, a_3, \ldots, a_n$ ($1 \le a_i \le 2 \cdot 10^5$) — the integers in the array a.

The third line of each test case contains n distinct integers $b_1, b_2, b_3, \ldots, b_n$ ($1 \le b_i \le 2 \cdot 10^5$) — the integers in the array b.

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

Output

For each testcase, print "YES" if the arrays a and b can be made the same. Otherwise, print "NO". can output the answer in any case (upper or lower). For example, the strings "yEs", "yes", "Yes", and "YES" will be recognized as positive responses.

Standard Input	Standard Output
6	YES
4	NO
1 2 3 4	YES
1 2 3 4	NO
5	NO
1 3 4 2 5	NO
7 1 2 5 4	
4	
1 2 3 4	
4 3 2 1	
3	
1 2 3	
1 3 2	
5	
1 5 7 1000 4	
4 1 7 5 1000	

3
1 4 2
1 3 2	

Note

In the first testcase, you don't need to perform any operations since the arrays are same.

In the second testcase, it can be proven there exists no way to make the arrays same.

In the third testcase, one of the ways to make the arrays same is to first choose $l = 1$, $r = 3$, $p = 1$, $q = 3$ then choose $l = 1$, $r = 2$, $p = 3$, $q = 4$.

,

E. I Love Balls

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

Alice and Bob are playing a game. There are n balls, out of which k are special. Each ball has a value associated with it.

The players play turn by turn. In each turn, the player randomly picks a ball and adds the value of the ball to their score, which is 0 at the beginning of the game. The selected ball is removed from the game. If the ball was special, the same player takes the next turn if at least one ball is remaining. If the ball picked was not special, the next player plays their turn.

They play this game until no balls are remaining in the game. Alice plays first.

Find the expected score that both the players have at the end of the game modulo $10^9 + 7$.

Formally, let $M = 10^9 + 7$. It can be shown that the answer can be expressed as an irreducible fraction $\frac{p}{q}$, where p and q are integers and $q \not\equiv 0 \pmod{M}$. Output the integer equal to $p \cdot q^{-1} \bmod M$. In other words, output such an integer x that $0 \le x < M$ and $x \cdot q \equiv p \pmod{M}$.

Input

There are multiple test cases. The first line of the input contains an integer t, the number of test cases ($1 \le t \le 2 \cdot 10^5$).

Each test case description is on a new line. The first line of the test case contains two integers n and k in the respective order separated by a space ($1 \le k \le n \le 4 \cdot 10^5$).

The second line of the test case contains n integers: $v_1, v_2, \ldots, v_n$, the value for each ball separated by spaces. The first k balls are special ($1 \le v_i \le 10^7$).

The sum of n over all test cases does not exceed $5 \cdot 10^5$.

Output

Output two integers per test case in a new line, the expected score of Alice and the expected score of Bob modulo $10^9 + 7$.

Standard Input	Standard Output

1
5 2
10 20 5 15 25
	45 30

5
1 1
732507
2 2
5817860 5398510
5 1
2122894 4951549 2750585 7821535 3214167
	732507 0
	11216370 0
	810642660 210218077
	722402997 318336932
	349086489 678010430

8 4
1405323 5069867 6883092 6972029 328406
2478975 7628890 9973340
4 2
9662050 3566134 3996473 9872255	

5
3 3
1095611 8219204 7773462
2 1
8176490 2774103
3 1
9178636 5138057 3367761
12 9
7597698 6843019 2298534 1522386 4969588
1340345 3967362 9152890 6689668 9986080
4745473 7407325
10 5
6986368 2397882 5804127 6980694 3740836
3215836 5195724 3179261 4136769 4544231
	17088277 0
	6862348 4088245
	677038671 340645790
	36949997 29570371
	725118051 321063684

Note

In the first test case, Alice's expected score is 45, and Bob's is 30 at the end of the game.

,

F. array-value

```
Input file:     standard input
Output file:    standard output
Time limit:     4 seconds
Memory limit:   256 megabytes
```

You have an array of non-negative integers $a_1, a_2, \ldots, a_n$.

The value of a sub-array of length ≥ 2, $a[l, r] = [a_l, a_{l+1}, \ldots, a_r]$ is the minimum value of $a_i \oplus a_j$ such that $l \leq i < j \leq r$, where $\oplus$ is the xor (exclusive-or) operator.

You have to find the k-th smallest value over all sub-arrays of length ≥ 2.

Input

The first line of the input contains multiple test cases t $(1 \leq t \leq 2 \cdot 10^4)$.

The first line of each test case contains integer numbers n and k $(2 \leq n \leq 10^5$, $1 \leq k \leq \frac{n\cdot(n-1)}{2})$.

The second line of the input contains n non-negative integer numbers $a_1, a_2, \ldots, a_n$ $(0 \leq a_i \leq 10^9)$ — the array itself.

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

Output

Print the k-th smallest value obtained over all subarrays of length at least 2.

Standard Input	Standard Output

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

Note

In the first testcase, we have subarrays with their smallest exclusive-or pair as:

$[1, 2] : 3$

$[2, 3] : 1$

$[3, 4] : 7$

$[4, 5] : 1$

$[1, 2, 3] : 1$

$[2, 3, 4] : 1$

$[3, 4, 5]$: 1

$[1, 2, 3, 4]$: 1

$[2, 3, 4, 5]$: 1

$[1, 2, 3, 4, 5]$: 1

The sorted order would be: $1, 1, 1, 1, 1, 1, 1, 1, 3, 7$. Therefore, the second smallest element would be 1.

,

G. Your Loss

```
Input file:     standard input
Output file:    standard output
Time limit:     3 seconds
Memory limit:   256 megabytes
```

You are given a tree with n nodes numbered from 1 to n, along with an array of size n. The value of i-th node is a_i. There are q queries. In each query, you are given 2 nodes numbered as x and y.

Consider the path from the node numbered as x to the node numbered as y. Let the path be represented by $x = p_0, p_1, p_2, \ldots, p_r = y$, where p_i are the intermediate nodes. Compute the sum of $a_{p_i} \oplus i$ for each i such that $0 \le i \le r$ where $\oplus$ is the XOR operator.

More formally, compute

$$\sum_{i=0}^{r} a_{p_i} \oplus i$$

.

Input

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

The first line of each set of input data contains a single integer n ($1 \le n \le 5 \cdot 10^5$) — the number of nodes.

The next $n-1$ lines of each set of input data contain 2 integers, u and v representing an edge between the node numbered u and the node numbered v. It is guaranteed that $u \neq v$ and that the edges form a tree.

The next line of each set of input data contains n integers, $a_1, a_2, \ldots, a_n$ ($1 \le a_i \le 5 \cdot 10^5$) — values of the nodes.

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

The next q lines describe the queries. The i-th query contains 2 integers x and y ($1 \le x, y \le n$) denoting the starting and the ending node of the path.

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

Output

For each query, output a single number — the sum from the problem statement.

Standard Input	Standard Output

1
4
1 2
2 3
3 4
2 3 6 5
3
	14
	10
	2

1 4
3 4
1 1	