A. Lex String

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

Kuznecov likes art, poetry, and music. And strings consisting of lowercase English letters.

Recently, Kuznecov has found two strings, a and b, of lengths n and m respectively. They consist of lowercase English letters and **no character is contained in both strings**.

Let another string c be initially empty. Kuznecov can do the following two types of operations:

- Choose any character from the string a, remove it from a, and add it to the end of c.
- Choose any character from the string b, remove it from b, and add it to the end of c.

But, he can not do more than k operations of the same type in a row. He must perform operations until either a or b becomes empty. What is the lexicographically smallest possible value of c after he finishes?

A string x is lexicographically smaller than a string y if and only if one of the following holds:

- x is a prefix of y, but $x \neq y$;
- in the first position where x and y differ, the string x has a letter that appears earlier in the alphabet than the corresponding letter in y.

Input

There are several test cases in the input data. The first line contains a single integer t ($1 \leq t \leq 100$) — the number of test cases. This is followed by the test cases description.

The first line of each test case contains three integers n, m, and k ($1 \leq n, m, k \leq 100$) — parameters from the statement.

The second line of each test case contains the string a of length n.

The third line of each test case contains the string b of length m.

The strings contain only lowercase English letters. It is guaranteed that no symbol appears in a and b simultaneously.

Output

In each test case, output a single string c — the answer to the problem.

Example

input

```
3
6 4 2
aaaaaa
bbbb
5 9 3
caaca
bedededeb
7 7 1
noskill
wxhtzdy
```

output

```
aabaabaa
aaabbcc
dihktlwlxnyoz
```

Note

In the first test case, it is optimal to take two 'a's from the string a and add them to the string c. Then it is forbidden to take more characters from a, hence one character 'b' from the string b has to be taken. Following that logic, we end up with c being 'aabaabaa' when string a is emptied.

In the second test case it is optimal to take as many 'a's from string a as possible, then take as many 'b's as possible from string b. In the end, we take two 'c's from the string a emptying it.

B. Mystic Permutation

time limit per test: 2 seconds
memory limit per test: 256 megabytes

input: standard input
output: standard output

Monocarp is a little boy who lives in Byteland and he loves programming.

Recently, he found a permutation of length n. He has to come up with a *mystic* permutation. It has to be a new permutation such that it differs from the old one in each position.

More formally, if the old permutation is $p_1, p_2, \ldots, p_n$ and the new one is $q_1, q_2, \ldots, q_n$ it must hold that

$$p_1 \neq q_1, p_2 \neq q_2, \ldots, p_n \neq q_n.$$

Monocarp is afraid of lexicographically large permutations. Can you please help him to find the lexicographically minimal *mystic* permutation?

Input

There are several test cases in the input data. The first line contains a single integer t ($1 \leq t \leq 200$) — the number of test cases. This is followed by the test cases description.

The first line of each test case contains a positive integer n ($1 \leq n \leq 1000$) — the length of the permutation.

The second line of each test case contains n distinct positive integers $p_1, p_2, \ldots, p_n$ ($1 \leq p_i \leq n$). It's guaranteed that p is a permutation, i. e. $p_i \neq p_j$ for all $i \neq j$.

It is guaranteed that the sum of n does not exceed 1000 over all test cases.

Output

For each test case, output n positive integers — the lexicographically minimal mystic permutations. If such a permutation does not exist, output -1 instead.

Example

input

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

output

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

Note

In the first test case possible permutations that are mystic are $[2,3,1]$ and $[3,1,2]$. Lexicographically smaller of the two is $[2,3,1]$.

In the second test case, $[1,2,3,4,5]$ is the lexicographically minimal permutation and it is also mystic.

In third test case possible mystic permutations are $[1,2,4,3]$, $[1,4,2,3]$, $[1,4,3,2]$, $[3,1,4,2]$, $[3,2,4,1]$, $[3,4,2,1]$, $[4,1,2,3]$, $[4,1,3,2]$ and $[4,3,2,1]$. The smallest one is $[1,2,4,3]$.

C. Infected Tree

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

Byteland is a beautiful land known because of its beautiful trees.

Misha has found a binary tree with n vertices, numbered from 1 to n. A binary tree is an acyclic connected bidirectional graph containing n vertices and $n-1$ edges. Each vertex has a degree at most 3, whereas the root is the vertex with the number 1 and it has a degree at most 2.

Unfortunately, the root got infected.

The following process happens n times:

- Misha either chooses a non-infected (and not deleted) vertex and deletes it with all edges which have an end in this vertex or just does nothing.
- Then, the infection spreads to each vertex that is connected by an edge to an already infected vertex (all already infected vertices remain infected).

As Misha does not have much time to think, please tell him what is the maximum number of vertices he can save from the infection (note that deleted vertices are not counted as saved).

Input

There are several test cases in the input data. The first line contains a single integer t ($1 \le t \le 5000$) — the number of test cases. This is followed by the test cases description.

The first line of each test case contains one integer n ($2 \le n \le 3 \cdot 10^5$) — the number of vertices of the tree.

The i-th of the following $n-1$ lines in the test case contains two positive integers u_i and v_i ($1 \le u_i, v_i \le n$), meaning that there exists an edge between them in the graph.

It is guaranteed that the graph is a binary tree rooted at 1. It is also guaranteed that the sum of n over all test cases won't exceed $3 \cdot 10^5$.

Output

For each test case, output the maximum number of vertices Misha can save.

Example

input

```
4
2
1 2
4
1 2
2 3
2 4
7
1 2
1 5
2 3
2 4
5 6
5 7
15
1 2
2 3
3 4
4 5
4 6
3 7
2 8
1 9
9 10
9 11
10 12
10 13
11 14
11 15
```

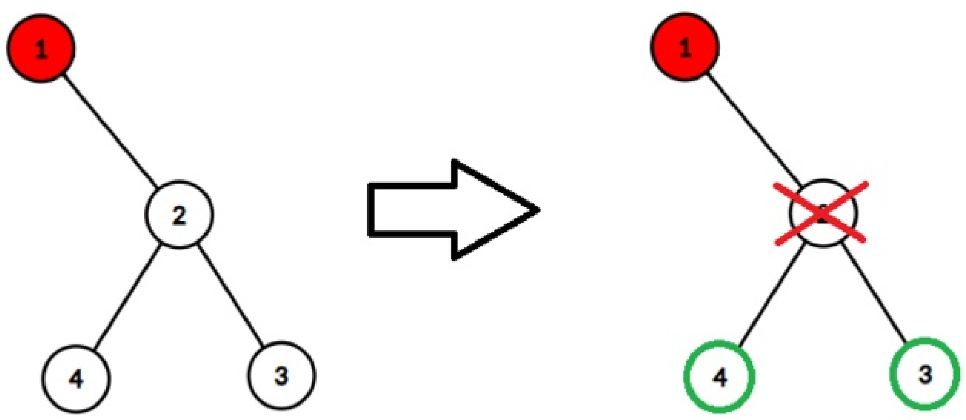
output

```
0
2
2
10
```

Note

In the first test case, the only possible action is to delete vertex 2, after which we save 0 vertices in total.

In the second test case, if we delete vertex 2, we can save vertices 3 and 4.



D. Lena and Matrix

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

output: standard output

Lena is a beautiful girl who likes logical puzzles.

As a gift for her birthday, Lena got a matrix puzzle!

The matrix consists of n rows and m columns, and each cell is either black or white. The coordinates (i, j) denote the cell which belongs to the i-th row and j-th column for every $1 \leq i \leq n$ and $1 \leq j \leq m$. To solve the puzzle, Lena has to choose a cell that minimizes the Manhattan distance to the farthest black cell from the chosen cell.

More formally, let there be $k \geq 1$ black cells in the matrix with coordinates (x_i, y_i) for every $1 \leq i \leq k$. Lena should choose a cell (a, b) that minimizes

$$\max_{i=1}^{k}(|a - x_i| + |b - y_i|).$$

As Lena has no skill, she asked you for help. Will you tell her the optimal cell to choose?

Input

There are several test cases in the input data. The first line contains a single integer t ($1 \leq t \leq 10\,000$) — the number of test cases. This is followed by the test cases description.

The first line of each test case contains two integers n and m ($2 \leq n, m \leq 1000$) — the dimensions of the matrix.

The following n lines contain m characters each, each character is either 'W' or 'B'. The j-th character in the i-th of these lines is 'W' if the cell (i, j) is white, and 'B' if the cell (i, j) is black.

It is guaranteed that at least one black cell exists.

It is guaranteed that the sum of $n \cdot m$ does not exceed 10^6.

Output

For each test case, output the optimal cell (a, b) to choose. If multiple answers exist, output any.

Example

input

```
5
3 2
BW
WW
WB
3 3
WWB
WBW
BWW
2 3
BBB
BBB
5 5
BWBWB
WBWBW
BWBWB
WBWBW
BWBWB
9 9
WWWWWWWWW
WWWWWWWWW
BWWWWWWWW
WWWWWWWWW
WWWWBWWWW
WWWWWWWWW
WWWWWWWWW
WWWWWWWWW
WWWWWWWWB
```

output

```
2 1
2 2
1 2
3 3
6 5
```

Note

In the first test case the two black cells have coordinates $(1, 1)$ and $(3, 2)$. The four optimal cells are $(1, 2)$, $(2, 1)$, $(2, 2)$ and $(3, 1)$. It can be shown that no other cell minimizes the maximum Manhattan distance to every black cell.

In the second test case it is optimal to choose the black cell $(2, 2)$ with maximum Manhattan distance being 2.

E. ANDfinity

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

Bit Lightyear, to the ANDfinity and beyond!

After graduating from computer sciences, Vlad has been awarded an array $a_1, a_2, \ldots, a_n$ of n non-negative integers. As it is natural, he wanted to construct a graph consisting of n vertices, numbered $1, 2, \ldots, n$. He decided to add an edge between i and j if and only if $a_i \& a_j > 0$, where $\&$ denotes the bitwise AND operation.

Vlad also wants the graph to be connected, which might not be the case initially. In order to satisfy that, he can do the following two types of operations on the array:

1. Choose some element a_i and increment it by 1.
2. Choose some element a_i and decrement it by 1 (possible only if $a_i > 0$).

It can be proven that there exists a finite sequence of operations such that the graph will be connected. So, can you please help Vlad find the minimum possible number of operations to do that and also provide the way how to do that?

Input

There are several test cases in the input data. The first line contains a single integer t ($1 \le t \le 1000$) — the number of test cases. This is followed by the test cases description.

The first line of each test case contains an integer n ($2 \le n \le 2000$).

The second line of each test case contains n integers $a_1, a_2, \ldots, a_n$ ($0 \le a_i < 2^{30}$) — the elements of the array.

It is guaranteed that the sum of n over all test cases does not exceed 2000.

Output

For each test case, print a single integer m in the first line — the minimum number of operations. In the second line print the array after a valid sequence of operations that have been done such that the graph from the task becomes connected.

If there are multiple solutions, output any.

Example

input

```
4
5
1 2 3 4 5
2
0 2
2
3 12
4
3 0 0 0
```

output

```
0
1 2 3 4 5
2
2 2
1
3 11
3
3 1 1 1
```

Note

In the first test case, the graph is already connected.

In the second test case, we can increment 0 twice and end up with the array $[2, 2]$. Since $2 \& 2 = 2 > 0$, the graph is connected. It can be shown that one operation is not enough.

In the third test case, we can decrement 12 once and we end up with an array $[3, 11]$. $3 \& 11 = 3 > 0$ hence the graph is connected. One operation has to be done since the graph is not connected at the beginning.