A

Add Odd or Subtract Even

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

You are given two positive integers a and b.

In one move, you can **change** a in the following way:

- Choose any positive **odd** integer x ($x > 0$) and replace a with $a + x$;
- choose any positive **even** integer y ($y > 0$) and replace a with $a - y$.

You can perform as many such operations as you want. You can choose the same numbers x and y in different moves.

Your task is to find the minimum number of moves required to obtain b from a. It is guaranteed that you can always obtain b from a.

You have to answer t independent test cases.

Input

The first line of the input contains one integer t ($1 \le t \le 10^4$) — the number of test cases.

Then t test cases follow. Each test case is given as two space-separated integers a and b ($1 \le a, b \le 10^9$).

Output

For each test case, print the answer — the minimum number of moves required to obtain b from a if you can perform any number of moves described in the problem statement. It is guaranteed that you can always obtain b from a.

Example

input

```
5
2 3
10 10
2 4
7 4
9 3
```

output

```
1
0
2
2
1
```

Note

In the first test case, you can just add 1.

In the second test case, you don't need to do anything.

In the third test case, you can add 1 two times.

In the fourth test case, you can subtract 4 and add 1.

In the fifth test case, you can just subtract 6.

B New Year and Permutation

time limit per test: 1 second

memory limit per test: 1024 megabytes

input: standard input

output: standard output

Recall that the permutation is an array consisting of n distinct integers from 1 to n in arbitrary order. For example, $[2,3,1,5,4]$ is a permutation, but $[1,2,2]$ is not a permutation (2 appears twice in the array) and $[1,3,4]$ is also not a permutation ($n=3$ but there is 4 in the array).

A sequence a is a subsegment of a sequence b if a can be obtained from b by deletion of several (possibly, zero or all) elements from the beginning and several (possibly, zero or all) elements from the end. We will denote the subsegments as $[l, r]$, where l, r are two integers with $1 \le l \le r \le n$. This indicates the subsegment where $l-1$ elements from the beginning and $n-r$ elements from the end are deleted from the sequence.

For a permutation $p_1, p_2, \ldots, p_n$, we define a *framed segment* as a subsegment $[l,r]$ where $\max\{p_l, p_{l+1}, \dots, p_r\} - \min\{p_l, p_{l+1}, \dots, p_r\} = r - l$. For example, for the permutation $(6, 7, 1, 8, 5, 3, 2, 4)$ some of its framed segments are: $[1, 2], [5, 8], [6, 7], [3, 3], [8, 8]$. In particular, a subsegment $[i,i]$ is always a framed segments for any i between 1 and n, inclusive.

We define the *happiness* of a permutation p as the number of pairs (l, r) such that $1 \le l \le r \le n$, and $[l, r]$ is a framed segment. For example, the permutation $[3, 1, 2]$ has happiness 5: all segments except $[1, 2]$ are framed segments.

Given integers n and m, Jongwon wants to compute the sum of happiness for all permutations of length n, modulo the prime number m. Note that there exist $n!$ (factorial of n) different permutations of length n.

Input

The only line contains two integers n and m ($1 \le n \le 250\,000$, $10^8 \le m \le 10^9$, m is prime).

Output

Print r ($0 \le r < m$), the sum of happiness for all permutations of length n, modulo a prime number m.

Examples

input

```
1 993244853
```

output

```
1
```

input

```
2 993244853
```

output

```
6
```

input

```
3 993244853
```

output

```
32
```

input

```
2019 993244853
```

output

```
923958830
```

input

```
2020 437122297
```

output

```
265955509
```

Note

For sample input $n = 3$, let's consider all permutations of length 3:

- $[1, 2, 3]$, all subsegments are framed segment. Happiness is 6.
- $[1, 3, 2]$, all subsegments except $[1, 2]$ are framed segment. Happiness is 5.
- $[2, 1, 3]$, all subsegments except $[2, 3]$ are framed segment. Happiness is 5.
- $[2, 3, 1]$, all subsegments except $[2, 3]$ are framed segment. Happiness is 5.
- $[3, 1, 2]$, all subsegments except $[1, 2]$ are framed segment. Happiness is 5.
- $[3, 2, 1]$, all subsegments are framed segment. Happiness is 6.

Thus, the sum of happiness is $6 + 5 + 5 + 5 + 5 + 6 = 32$.

C. Distributed Join

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

Piegirl was asked to implement two table join operation for distributed database system, minimizing the network traffic.

Suppose she wants to join two tables, A and B. Each of them has certain number of rows which are distributed on different number of partitions. Table A is distributed on the first cluster consisting of m partitions. Partition with index i has a_i rows from A. Similarly, second cluster containing table B has n partitions, i-th one having b_i rows from B.

In one network operation she can copy one row from any partition to any other partition. At the end, for each row from A and each row from B there should be a partition that has both rows. Determine the minimal number of network operations to achieve this.

Input

First line contains two integer numbers, m and n ($1 \le m, n \le 10^5$). Second line contains description of the first cluster with m space separated integers, a_i ($1 \le a_i \le 10^9$). Similarly, third line describes second cluster with n space separated integers, b_i ($1 \le b_i \le 10^9$).

Output

Print one integer — minimal number of copy operations.

Examples

input

```
2 2
2 6
3 100
```

output

```
11
```

input

```
2 3
10 10
1 1 1
```

output

```
6
```

Note

In the first example it makes sense to move all the rows to the second partition of the second cluster which is achieved in $2+6+3=11$ operations

In the second example Piegirl can copy each row from B to the both partitions of the first cluster which needs $2 \cdot 3 = 6$ copy operations.

D Restaurant Tables

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

In a small restaurant there are a tables for one person and b tables for two persons.

It it known that n groups of people come today, each consisting of one or two people.

If a group consist of one person, it is seated at a vacant one-seater table. If there are none of them, it is seated at a vacant two-seater table. If there are none of them, it is seated at a two-seater table occupied by single person. If there are still none of them, the restaurant denies service to this group.

If a group consist of two people, it is seated at a vacant two-seater table. If there are none of them, the restaurant denies service to this group.

You are given a chronological order of groups coming. You are to determine the total number of people the restaurant denies service to.

Input

The first line contains three integers n, a and b ($1 \le n \le 2 \cdot 10^5$, $1 \le a, b \le 2 \cdot 10^5$) — the number of groups coming to the restaurant, the number of one-seater and the number of two-seater tables.

The second line contains a sequence of integers $t_1, t_2, \ldots, t_n$ ($1 \le t_i \le 2$) — the description of clients in chronological order. If t_i is equal to one, then the i-th group consists of one person, otherwise the i-th group consists of two people.

Output

Print the total number of people the restaurant denies service to.

Examples

input

```
4 1 2
1 2 1 1
```

output

```
0
```

input

```
4 1 1
1 1 2 1
```

output

```
2
```

Note

In the first example the first group consists of one person, it is seated at a vacant one-seater table. The next group occupies a whole two-seater table. The third group consists of one person, it occupies one place at the remaining two-seater table. The fourth group consists of one person, he is seated at the remaining seat at the two-seater table. Thus, all clients are served.

In the second example the first group consists of one person, it is seated at the vacant one-seater table. The next group consists of one person, it occupies one place at the two-seater table. It's impossible to seat the next group of two people, so the restaurant denies service to them. The fourth group consists of one person, he is seated at the remaining seat at the two-seater table.

Thus, the restaurant denies service to 2 clients.

E

Alice, Bob and Chocolate

time limit per test: 2 seconds
memory limit per test: 64 megabytes
input: standard input
output: standard output

Alice and Bob like games. And now they are ready to start a new game. They have placed n chocolate bars in a line. Alice starts to eat chocolate bars one by one from left to right, and Bob — from right to left. For each chocololate bar the time, needed for the player to consume it, is known (Alice and Bob eat them with equal speed). When the player consumes a chocolate bar, he immediately starts with another. It is not allowed to eat two chocolate bars at the same time, to leave the bar unfinished and to make pauses. If both players start to eat the same bar simultaneously, Bob leaves it to Alice as a true gentleman.

How many bars each of the players will consume?

Input

The first line contains one integer n $(1 \le n \le 10^5)$ — the amount of bars on the table. The second line contains a sequence $t_1, t_2, ..., t_n$ $(1 \le t_i \le 1000)$, where t_i is the time (in seconds) needed to consume the i-th bar (in the order from left to right).

Output

Print two numbers a and b, where a is the amount of bars consumed by Alice, and b is the amount of bars consumed by Bob.

Examples

input

```
5
2 9 8 2 7
```

output

```
2 3
```

F Wet Shark and Blocks

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

There are b blocks of digits. Each one consisting of the same n digits, which are given to you in the input. Wet Shark must choose **exactly one** digit from each block and concatenate all of those digits together to form one large integer. For example, if he chooses digit 1 from the first block and digit 2 from the second block, he gets the integer 12.

Wet Shark then takes this number modulo x. Please, tell him how many ways he can choose one digit from each block so that he gets exactly k as the final result. As this number may be too large, print it modulo $10^9 + 7$.

Note, that the number of ways to choose some digit in the block is equal to the number of it's occurrences. For example, there are 3 ways to choose digit 5 from block `3 5 6 7 8 9 5 1 1 1 1 5`.

Input

The first line of the input contains four space-separated integers, n, b, k and x ($2 \le n \le 50\,000$, $1 \le b \le 10^9$, $0 \le k < x \le 100$, $x \ge 2$) — the number of digits in one block, the number of blocks, interesting remainder modulo x and modulo x itself.

The next line contains n space separated integers a_i ($1 \le a_i \le 9$), that give the digits contained in each block.

Output

Print the number of ways to pick exactly one digit from each blocks, such that the resulting integer equals k modulo x.

Examples

input
```
12 1 5 10
3 5 6 7 8 9 5 1 1 1 1 5
```
output
```
3
```

input
```
3 2 1 2
6 2 2
```
output
```
0
```

input
```
3 2 1 2
3 1 2
```
output
```
6
```

Note

In the second sample possible integers are 22, 26, 62 and 66. None of them gives the remainder 1 modulo 2.

In the third sample integers 11, 13, 21, 23, 31 and 33 have remainder 1 modulo 2. There is exactly one way to obtain each of these integers, so the total answer is 6.

G

Multihedgehog

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

Someone give a strange birthday present to Ivan. It is hedgehog — connected undirected graph in which one vertex has degree at least 3 (we will call it center) and all other vertices has degree 1. Ivan thought that hedgehog is too boring and decided to make himself k-multihedgehog.

Let us define k-multihedgehog as follows:

- 1-multihedgehog is hedgehog: it has one vertex of degree at least 3 and some vertices of degree 1.
- For all $k \geq 2$, k-multihedgehog is $(k-1)$-multihedgehog in which the following changes has been made for each vertex v with degree 1: let u be its only neighbor; remove vertex v, create a new hedgehog with center at vertex w and connect vertices u and w with an edge. New hedgehogs can differ from each other and the initial gift.

Thereby k-multihedgehog is a tree. Ivan made k-multihedgehog but he is not sure that he did not make any mistakes. That is why he asked you to check if his tree is indeed k-multihedgehog.

Input

First line of input contains 2 integers n, k $(1 \leq n \leq 10^5, 1 \leq k \leq 10^9)$ — number of vertices and hedgehog parameter.

Next $n-1$ lines contains two integers u v $(1 \leq u, \ v \leq n; \ u \neq v)$ — indices of vertices connected by edge.

It is guaranteed that given graph is a tree.

Output

Print "`Yes`" (without quotes), if given graph is k-multihedgehog, and "`No`" (without quotes) otherwise.

Examples

input

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

output

```
Yes
```

input

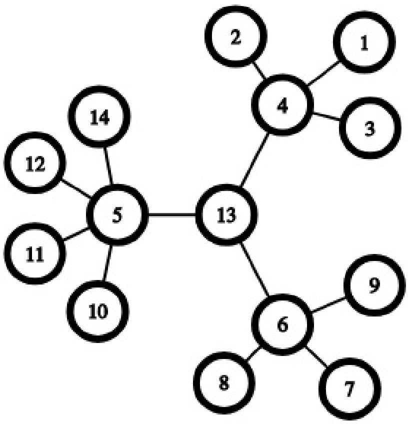
```
3 1
1 3
2 3
```

output

```
No
```

Note

2-multihedgehog from the first example looks like this:



Its center is vertex 13. Hedgehogs created on last step are: [4 (center), 1, 2, 3], [6 (center), 7, 8, 9], [5 (center), 10, 11, 12, 13].

Tree from second example is not a hedgehog because degree of center should be at least 3.

H

Games with Rectangle

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

In this task Anna and Maria play the following game. Initially they have a checkered piece of paper with a painted $n \times m$ rectangle (only the border, no filling). Anna and Maria move in turns and Anna starts. During each move one should paint inside the last-painted rectangle a new lesser rectangle (along the grid lines). The new rectangle should have no common points with the previous one. Note that when we paint a rectangle, we always paint only the border, the rectangles aren't filled.

Nobody wins the game — Anna and Maria simply play until they have done k moves in total. Count the number of different ways to play this game.

Input

The first and only line contains three integers: n, m, k ($1 \le n, m, k \le 1000$).

Output

Print the single number — the number of the ways to play the game. As this number can be very big, print the value modulo 1000000007 ($10^9 + 7$).

Examples

input

```
3 3 1
```

output

```
1
```

input

```
4 4 1
```

output

```
9
```

input

```
6 7 2
```

output

```
75
```

Note

Two ways to play the game are considered different if the final pictures are different. In other words, if one way contains a rectangle that is not contained in the other way.

In the first sample Anna, who performs her first and only move, has only one possible action plan — insert a 1×1 square inside the given 3×3 square.

In the second sample Anna has as much as 9 variants: 4 ways to paint a 1×1 square, 2 ways to insert a 1×2 rectangle vertically, 2 more ways to insert it horizontally and one more way is to insert a 2×2 square.

I

Array with Odd Sum

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

You are given an array a consisting of n integers.

In one move, you can choose two indices $1 \le i, j \le n$ such that $i \ne j$ and set $a_i := a_j$. You can perform such moves any number of times (possibly, zero). You can choose different indices in different operations. The operation := is the operation of assignment (i.e. you choose i and j and replace a_i with a_j).

Your task is to say if it is possible to obtain an array with an odd (not divisible by 2) sum of elements.

You have to answer t independent test cases.

Input

The first line of the input contains one integer t ($1 \le t \le 2000$) — the number of test cases.

The next $2t$ lines describe test cases. The first line of the test case contains one integer n ($1 \le n \le 2000$) — the number of elements in a. The second line of the test case contains n integers $a_1, a_2, \dots, a_n$ ($1 \le a_i \le 2000$), where a_i is the i-th element of a.

It is guaranteed that the sum of n over all test cases does not exceed 2000 ($\sum n \le 2000$).

Output

For each test case, print the answer on it — "YES" (without quotes) if it is possible to obtain the array with an odd sum of elements, and "NO" otherwise.

Example

input

```
5
2
2 3
4
2 2 8 8
3
3 3 3
4
5 5 5 5
4
1 1 1 1
```

output

```
YES
NO
YES
NO
NO
```

J

Photographer

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

Valera's lifelong ambition was to be a photographer, so he bought a new camera. Every day he got more and more clients asking for photos, and one day Valera needed a program that would determine the maximum number of people he can serve.

The camera's memory is d megabytes. Valera's camera can take photos of high and low quality. One low quality photo takes a megabytes of memory, one high quality photo take b megabytes of memory. For unknown reasons, each client asks him to make several low quality photos and several high quality photos. More formally, the i-th client asks to make x_i low quality photos and y_i high quality photos.

Valera wants to serve as many clients per day as possible, provided that they will be pleased with his work. To please the i-th client, Valera needs to give him everything he wants, that is, to make x_i low quality photos and y_i high quality photos. To make one low quality photo, the camera must have at least a megabytes of free memory space. Similarly, to make one high quality photo, the camera must have at least b megabytes of free memory space. Initially the camera's memory is empty. Valera also does not delete photos from the camera so that the camera's memory gradually fills up.

Calculate the maximum number of clients Valera can successfully serve and print the numbers of these clients.

Input

The first line contains two integers n and d $(1 \le n \le 10^5, 1 \le d \le 10^9)$ — the number of clients and the camera memory size, correspondingly. The second line contains two integers a and b $(1 \le a \le b \le 10^4)$ — the size of one low quality photo and of one high quality photo, correspondingly.

Next n lines describe the clients. The i-th line contains two integers x_i and y_i $(0 \le x_i, y_i \le 10^5)$ — the number of low quality photos and high quality photos the i-th client wants, correspondingly.

All numbers on all lines are separated by single spaces.

Output

On the first line print the answer to the problem — the maximum number of clients that Valera can successfully serve. Print on the second line the numbers of the client in any order. All numbers must be distinct. If there are multiple answers, print any of them. The clients are numbered starting with 1 in the order in which they are defined in the input data.

Examples

input

```
3 10
2 3
1 4
2 1
1 0
```

output

```
2
3 2
```

input

```
3 6
```

```
6 6
1 1
1 0
1 0
```

output

```
1
2
```

K

Short Substrings

time limit per test: 2 seconds

memory limit per test: 256 megabytes

input: standard input

output: standard output

Alice guesses the strings that Bob made for her.

At first, Bob came up with the secret string a consisting of lowercase English letters. The string a has a length of 2 or more characters. Then, from string a he builds a new string b and offers Alice the string b so that she can guess the string a.

Bob builds b from a as follows: he writes all the substrings of length 2 of the string a in the order from left to right, and then joins them in the same order into the string b.

For example, if Bob came up with the string a="abac", then all the substrings of length 2 of the string a are: "ab", "ba", "ac". Therefore, the string b="abbaac".

You are given the string b. Help Alice to guess the string a that Bob came up with. It is guaranteed that b was built according to the algorithm given above. It can be proved that the answer to the problem is unique.

Input

The first line contains a single positive integer t $(1 \le t \le 1000)$ — the number of test cases in the test. Then t test cases follow.

Each test case consists of one line in which the string b is written, consisting of lowercase English letters $(2 \le |b| \le 100)$ — the string Bob came up with, where $|b|$ is the length of the string b. It is guaranteed that b was built according to the algorithm given above.

Output

Output t answers to test cases. Each answer is the secret string a, consisting of lowercase English letters, that Bob came up with.

Example

input

```
4
abbaac
ac
bccddaaf
zzzzzzzzzz
```

output

```
abac
ac
bcdaf
zzzzzz
```

Note

The first test case is explained in the statement.

In the second test case, Bob came up with the string a="ac", the string a has a length 2, so the string b is equal to the string a.

In the third test case, Bob came up with the string a="bcdaf", substrings of length 2 of string a are: "bc", "cd", "da", "af", so the string b="bccddaaf".

L

Relay Race

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

Furik and Rubik take part in a relay race. The race will be set up on a large square with the side of n meters. The given square is split into $n \times n$ cells (represented as unit squares), each cell has some number.

At the beginning of the race Furik stands in a cell with coordinates $(1, 1)$, and Rubik stands in a cell with coordinates (n, n). Right after the start Furik runs towards Rubik, besides, if Furik stands at a cell with coordinates (i, j), then he can move to cell $(i + 1, j)$ or $(i, j + 1)$. After Furik reaches Rubik, Rubik starts running from cell with coordinates (n, n) to cell with coordinates $(1, 1)$. If Rubik stands in cell (i, j), then he can move to cell $(i - 1, j)$ or $(i, j - 1)$. Neither Furik, nor Rubik are allowed to go beyond the boundaries of the field; if a player goes beyond the boundaries, he will be disqualified.

To win the race, Furik and Rubik must earn as many points as possible. The number of points is the sum of numbers from the cells Furik and Rubik visited. **Each cell counts only once in the sum**.

Print the maximum number of points Furik and Rubik can earn on the relay race.

Input

The first line contains a single integer $(1 \le n \le 300)$. The next n lines contain n integers each: the j-th number on the i-th line $a_{i,j}$ $(-1000 \le a_{i,j} \le 1000)$ is the number written in the cell with coordinates (i, j).

Output

On a single line print a single number — the answer to the problem.

Examples

input

```
1
5
```

output

```
5
```

input

```
2
11 14
16 12
```

output

```
53
```

input

```
3
25 16 25
12 18 19
11 13 8
```

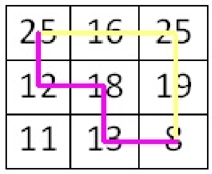
output

```
136
```

Note

Comments to the second sample: The profitable path for Furik is: $(1, 1)$, $(1, 2)$, $(2, 2)$, and for Rubik: $(2, 2)$, $(2, 1)$, $(1, 1)$.

Comments to the third sample: The optimal path for Furik is: $(1, 1)$, $(1, 2)$, $(1, 3)$, $(2, 3)$, $(3, 3)$, and for Rubik: $(3, 3)$, $(3, 2)$, $(2, 2)$, $(2, 1)$, $(1, 1)$. The figure to the sample:



Furik's path is marked with yellow, and Rubik's path is marked with pink.