Disco CodeJam Challenge 2

A. New Year Candles

2.0 s, 256 megabytes

Vasily the Programmer loves romance, so this year he decided to illuminate his room with candles.

Vasily has a candles.When Vasily lights up a new candle, it first burns for an hour and then it goes out. Vasily is smart, so he can make b went out candles into a new candle. As a result, this new candle can be used like any other new candle.

Now Vasily wonders: for how many hours can his candles light up the room if he acts optimally well? Help him find this number.

Input

The single line contains two integers, a and b $(1 \le a \le 1000; 2 \le b \le 1000)$.

Output

Print a single integer — the number of hours Vasily can light up the room for.

input
4 2
output
7

input
6 3
output
8

Consider the first sample. For the first four hours Vasily lights up new candles, then he uses four burned out candles to make two new ones and lights them up. When these candles go out (stop burning), Vasily can make another candle. Overall, Vasily can light up the room for 7 hours.

B. Laptops

3.0 s, 512 MB

One day Dima and Alex had an argument about the price and quality of laptops. Dima thinks that the more expensive a laptop is, the better it is. Alex disagrees. Alex thinks that there are two laptops, such that the price of the first laptop is less (strictly smaller) than the price of the second laptop but the quality of the first laptop is higher (strictly greater) than the quality of the second laptop.

Please, check the guess of Alex. You are given descriptions of n laptops. Determine whether two described above laptops exist.

Input

The first line contains an integer n $(1 \le n \le 10^5)$ — the number of laptops.

Next n lines contain two integers each, a_i and b_i $(1 \le a_i, b_i \le n)$, where a_i is the price of the i-th laptop, and b_i is the number that represents the quality of the i-th laptop (the larger the number is, the higher is the quality).

All a_i are distinct. All b_i are distinct.

Output

If Alex is correct, print "`Happy Alex`", otherwise print "`Poor Alex`" (without the quotes).

input
2 1 2 2 1
output
Happy Alex

C. Maze

4.5 s, 512 MB

Pavel loves grid mazes. A grid maze is an $n \times m$ rectangle maze where each cell is either empty, or is a wall. You can go from one cell to another only if both cells are empty and have a common side.

Pavel drew a grid maze with all empty cells forming a connected area. That is, you can go from any empty cell to any other one. Pavel doesn't like it when his maze has too little walls. He wants to turn exactly k empty cells into walls so that all the remaining cells still formed a connected area. Help him.

Input

The first line contains three integers n, m, k $(1 \le n, m \le 500, 0 \le k < s)$, where n and m are the maze's height and width, correspondingly, k is the number of walls Pavel wants to add and letter s represents the number of empty cells in the original maze.

Each of the next n lines contains m characters. They describe the original maze. If a character on a line equals "`.`", then the corresponding cell is empty and if the character equals "`#`", then the cell is a wall.

Output

Print n lines containing m characters each: the new maze that fits Pavel's requirements. Mark the empty cells that you transformed into walls as "`X`", the other cells must be left without changes (that is, "`.`" and "`#`").

It is guaranteed that a solution exists. If there are multiple solutions you can output any of them.

input
3 4 2 #..# ..#. #...
output
#.X# X.#. #...

input
5 4 5 #... #.#. .#.. ...# .#.#

output

```
#XXX
#X#.
X#..
...#
.#.#
```

D. Kalindrome Array

3.0 s, 512 MB

An array $[b_1, b_2, \ldots, b_m]$ is a palindrome, if $b_i = b_{m+1-i}$ for each i from 1 to m. Empty array is also a palindrome.

An array is called **kalindrome**, if the following condition holds:

- It's possible to select some integer x and delete some of the elements of the array equal to x, so that the remaining array (after gluing together the remaining parts) is a palindrome.

Note that you don't have to delete all elements equal to x, and you don't have to delete at least one element equal to x.

For example :

- $[1, 2, 1]$ is kalindrome because you can simply not delete a single element.
- $[3, 1, 2, 3, 1]$ is kalindrome because you can choose $x = 3$ and delete both elements equal to 3, obtaining array $[1, 2, 1]$, which is a palindrome.
- $[1, 2, 3]$ is not kalindrome.

You are given an array $[a_1, a_2, \ldots, a_n]$. Determine if a is kalindrome or not.

Input

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

The first line of each test case contains a single integer n ($1 \le n \le 2 \cdot 10^5$) — the length of the array.

The second line of each test case contains n integers $a_1, a_2, \ldots, a_n$ ($1 \le a_i \le n$) — elements of the array.

It's guaranteed that the sum of n over all test cases won't exceed $2 \cdot 10^5$.

Output

For each test case, print `YES` if a is kalindrome and `NO` otherwise. You can print each letter in any case.

input

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

output

```
YES
YES
NO
YES
```

In the first test case, array $[1]$ is already a palindrome, so it's a kalindrome as well.

In the second test case, we can choose $x = 2$, delete the second element, and obtain array $[1]$, which is a palindrome.

In the third test case, it's impossible to obtain a palindrome.

In the fourth test case, you can choose $x = 4$ and delete the fifth element, obtaining $[1, 4, 4, 1]$. You also can choose $x = 1$, delete the first and the fourth elements, and obtain $[4, 4, 4]$.

E. Fox and Minimal path

2.0 s, 512 MB

Fox Ciel wants to write a task for a programming contest. The task is: "You are given a simple undirected graph with n vertexes. Each its edge has unit length. You should calculate the number of shortest paths between vertex 1 and vertex 2."

Same with some writers, she wants to make an example with some certain output: for example, her birthday or the number of her boyfriend. Can you help her to make a test case with answer equal exactly to k?

Input

The first line contains a single integer k ($1 \le k \le 10^9$).

Output

You should output a graph G with n vertexes ($2 \le n \le 1000$). There must be exactly k shortest paths between vertex 1 and vertex 2 of the graph.

The first line must contain an integer n. Then adjacency matrix G with n rows and n columns must follow. Each element of the matrix must be '`N`' or '`Y`'. If G_{ij} is '`Y`', then graph G has a edge connecting vertex i and vertex j. Consider the graph vertexes are numbered from 1 to n.

The graph must be undirected and simple: G_{ii} = '`N`' and $G_{ij} = G_{ji}$ must hold. And there must be at least one path between vertex 1 and vertex 2. It's guaranteed that the answer exists. If there multiple correct answers, you can output any of them.

input

```
2
```

output

```
4
NNYY
NNYY
YYNN
YYNN
```

input

```
9
```

output

```
8
NNYYYNNN
NNNNNYYY
YNNNNYYY
YNNNNYYY
YNNNNYYY
NYYYYNNN
NYYYYNNN
NYYYYNNN
```

input

```
1
```

output
2 NY YN

In first example, there are 2 shortest paths: 1-3-2 and 1-4-2.

In second example, there are 9 shortest paths: 1-3-6-2, 1-3-7-2, 1-3-8-2, 1-4-6-2, 1-4-7-2, 1-4-8-2, 1-5-6-2, 1-5-7-2, 1-5-8-2.

F. Yet Another Minimization Problem

4.0 s, 512 MB

You are given two arrays a and b, both of length n.

You can perform the following operation any number of times (possibly zero): select an index i ($1 \le i \le n$) and swap a_i and b_i.

Let's define the *cost* of the array a as $\sum_{i=1}^{n} \sum_{j=i+1}^{n} (a_i + a_j)^2$. Similarly, the *cost* of the array b is $\sum_{i=1}^{n} \sum_{j=i+1}^{n} (b_i + b_j)^2$.

Your task is to minimize the total *cost* of two arrays.

Input

Each test case consists of several test cases. The first line contains a single integer t ($1 \le t \le 40$) — the number of test cases. The following is a description of the input data sets.

The first line of each test case contains an integer n ($1 \le n \le 100$) — the length of both arrays.

The second line of each test case contains n integers $a_1, a_2, \ldots, a_n$ ($1 \le a_i \le 100$) — elements of the first array.

The third line of each test case contains n integers $b_1, b_2, \ldots, b_n$ ($1 \le b_i \le 100$) — elements of the second array.

It is guaranteed that the sum of n over all test cases does not exceed 100.

Output

For each test case, print the minimum possible total cost.

input
3 1 3 6 4 3 6 6 6 2 7 4 1 4 6 7 2 4 2 5 3 5

output
0 987 914

In the second test case, in one of the optimal answers after all operations $a = [2, 6, 4, 6]$, $b = [3, 7, 6, 1]$.

The cost of the array a equals to $(2 + 6)^2 + (2 + 4)^2 + (2 + 6)^2 + (6 + 4)^2 + (6 + 6)^2 + (4 + 6)^2 = 508$.

The cost of the array b equals to $(3 + 7)^2 + (3 + 6)^2 + (3 + 1)^2 + (7 + 6)^2 + (7 + 1)^2 + (6 + 1)^2 = 479$.

The total cost of two arrays equals to $508 + 479 = 987$.