A. Dinner Time

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

Given four integers n, m, p, and q, determine whether there exists an integer array $a_1, a_2, \ldots, a_n$ (elements may be negative) satisfying the following conditions:

- The sum of all elements in the array is equal to m:

$$a_1 + a_2 + \ldots + a_n = m$$

- The sum of every p consecutive elements is equal to q:

$$a_i + a_{i+1} + \ldots + a_{i+p-1} = q, \qquad \text{for all } 1 \le i \le n - p + 1$$

Input

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

The first and only line of each test case contains four integers n, m, p, and q ($1 \le p \le n \le 100$, $1 \le q, m \le 100$) — the length of the array, the sum of elements, the length of a segment, and the sum of a segment, respectively.

Output

For each test case, output "YES" (without quotes) if there exists an array satisfying the above conditions, and "NO" (without quotes) otherwise.

You can output "YES" and "NO" in any case (for example, strings "yES", "yes", and "Yes" will all be recognized as valid responses).

Standard Input	Standard Output
5 3 2 2 1 1 1 1 1 5 4 2 3 10 7 5 2 4 4 1 3	YES YES YES NO NO

Note

In the first test case, an example of an array satisfying the condition is $[1, 0, 1]$. This is because:

- $a_1 + a_2 + a_3 = 1 + 0 + 1 = 2 = m$
- $a_1 + a_2 = 1 + 0 = 1 = q$
- $a_2 + a_3 = 0 + 1 = 1 = q$

In the second test case, the only array satisfying the condition is $[1]$.

In the third test case, an example of an array satisfying the condition is $[-2, 5, -2, 5, -2]$.

In the fourth test case, it can be proven that there is no array satisfying the condition.

,

B. The Picky Cat

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

You are given an array of integers $a_1, a_2, \ldots, a_n$. You are allowed to do the following operation any number of times (possibly zero):

- Choose an index i ($1 \leq i \leq n$). Multiply a_i by -1 (i.e., update $a_i := -a_i$).

Your task is to determine whether it is possible to make the element at index 1 become the median of the array after doing the above operation any number of times. Note that operations can be applied to index 1 as well, meaning the median can be either the original value of a_1 or its negation.

The median of an array $b_1, b_2, \ldots, b_m$ is defined as the $\left\lceil \frac{m}{2} \right\rceil$-th* smallest element of array b. For example, the median of the array $[3, 1, 2]$ is 2, while the median of the array $[10, 1, 8, 3]$ is 3.

It is guaranteed that the absolute value of the elements of a are distinct. Formally, there are no pairs of indices $1 \leq i < j \leq n$ where $|a_i| = |a_j|$.

* $\lceil x \rceil$ is the ceiling function which returns the least integer greater than or equal to x.

Input

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

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

The second line of each test case contains n integers $a_1, a_2, \ldots, a_n$ ($|a_i| \leq 10^6$, $|a_i| \neq |a_j|$) — the elements of the array a.

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

Output

For each testcase, output "YES" if it is possible to make the element at index 1 become the median of the array, and "NO" 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 2 3 1 5 1 2 3 4 5 4 4 2 0 -5 4 -5 0 4 3	YES YES YES NO NO YES YES

4 -10 8 3 2 1 1 10 9 1000 -999 -13 456 -223 23 24 10 0	

Note

In the first test case, $a_1 = 2$ is already the median of the array $a = [2, 3, 1]$, so no operation is required.

In the second test case, we can do two operations: one on index 2, and one on index 5. The array becomes $[1, -2, 3, 4, -5]$, which has a median of 1.

In the third test case, if you do an operation on index 1, the array will become $[-4, 2, 0, -5]$, which has a median of -4.

In the fourth test case, it can be proven that no sequence of operations can make the median of the array become 5 or -5.

,

C. Mex in the Grid

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

You are given n^2 cards with values from 0 to $n^2 - 1$. You are to arrange them in a n by n grid such that there is **exactly** one card in each cell.

The MEX (minimum excluded value) of a subgrid* is defined as the smallest non-negative integer that does not appear in the subgrid.

Your task is to arrange the cards such that the sum of MEX values over all $\left(\frac{n(n+1)}{2}\right)^2$ subgrids is maximized.

*A subgrid of a n by n grid is specified by four numbers l_1, r_1, l_2, r_2 satisfying $1 \le l_1 \le r_1 \le n$ and $1 \le l_2 \le r_2 \le n$. The element in the i-th row and the j-th column of the grid is part of the subgrid if and only if $l_1 \le i \le r_1$ and $l_2 \le j \le r_2$.

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 line of each test case contains a single integer n ($1 \le n \le 500$) — the side length of the grid.

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

Output

For each test case, output n lines, each containing n integers representing the elements of the grid.

If there are multiple answers, you can output any of them.

Standard Input	Standard Output
2 2 3	0 1 2 3 8 4 5 6 0 1 7 2 3

Note

In the first test case, one valid arrangement is:

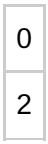
0	1
2	3

There are 9 subgrids in total, and the 4 of them with non-zero MEX are shown below:

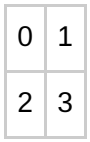
0

values: $[0]$ — MEX: 1

values:$[0, 1]$ — MEX: 2



values:$[0, 2]$ — MEX: 1



values:$[0, 1, 2, 3]$ — MEX: 4

The sum of MEX over all subgrids would be $1 + 2 + 1 + 4 = 8$. It can be proven that no other arrangements have a larger sum of MEX values.

,

D. Quartet Swapping

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

You are given a permutation a of length n*. You are allowed to do the following operation any number of times (possibly zero):

- Choose an index $1 \leq i \leq n-3$. Then, swap a_i with a_{i+2}, and a_{i+1} with a_{i+3} simultaneously. In other words, permutation a will be transformed from $[\ldots, a_i, a_{i+1}, a_{i+2}, a_{i+3}, \ldots]$ to $[\ldots, a_{i+2}, a_{i+3}, a_i, a_{i+1}, \ldots]$.

Determine the lexicographically smallest permutation† that can be obtained by applying the above operation any number of times.

---

*A permutation of length n 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).

†An array x is lexicographically smaller than an array y of the same size if and only if the following holds:

- in the first position where x and y differ, the array x has a smaller element than the corresponding element in y.

Input

Each test contains multiple test cases. The first line contains the number of test cases t ($1 \leq t \leq 1000$). The description of the test cases follows.

The first line of each test case contains a single integer n ($4 \leq n \leq 2 \cdot 10^5$) — the length of permutation a.

The second line contains n integers $a_1, a_2, \ldots, a_n$ ($1 \leq a_i \leq n$) — the elements of permutation a.

It is guaranteed that the sum of n over all test cases does not exceed $2 \cdot 10^5$.

Output

For each test case, output the lexicographically smallest permutation that can be obtained by applying the above operation any number of times.

Standard Input	Standard Output
3 4 3 4 1 2 5 5 4 3 1 2 10 10 9 8 7 6 5 4 3 2 1	1 2 3 4 2 1 3 4 5 2 1 4 3 6 5 8 7 10 9

Note

In the first test case, an operation can be done on index $i=1$, and the permutation will become $[1,2,3,4]$, which is the lexicographically smallest permutation achievable.

In the second test case, we can do the following sequence of operations:

- Do an operation on index $i = 2$. The permutation becomes $[5, 1, 2, 4, 3]$.
- Do an operation on index $i = 1$. The permutation becomes $[2, 4, 5, 1, 3]$.
- Do an operation on index $i = 2$. The permutation becomes $[2, 1, 3, 4, 5]$.

,

E. 23 Kingdom

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

The *distance* of a value x in an array c, denoted as $d_x(c)$, is defined as the largest gap between any two occurrences of x in c.

Formally, $d_x(c) = \max(j - i)$ over all pairs $i < j$ where $c_i = c_j = x$. If x appears only once or not at all in c, then $d_x(c) = 0$.

The *beauty* of an array is the sum of the distances of each distinct value in the array. Formally, the beauty of an array c is equal to $\sum\limits_{1 \le x \le n} d_x(c)$.

Given an array a of length n, an array b is *nice* if it also has length n and its elements satisfy $1 \le b_i \le a_i$ for all $1 \le i \le n$. Your task is to find the maximum possible beauty of any nice array.

Input

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

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

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

It is guaranteed that the sum of n over all test cases does not exceed $2 \cdot 10^5$.

Output

For each test case, output a single integer representing the maximum possible beauty among all nice arrays.

Standard Input	Standard Output
4 4 1 2 1 2 2 2 2 10 1 2 1 5 1 2 2 1 1 2 8 1 5 2 8 4 1 4 2	4 1 16 16

Note

In the first test case, if $b = [1, 2, 1, 2]$, then $d_1(b) = 3 - 1 = 2$ and $d_2(b) = 4 - 2 = 2$, resulting in a beauty of $2 + 2 = 4$. It can be proven that there are no nice arrays with a beauty greater than 4.

In the second test case, both $b = [1, 1]$ and $b = [2, 2]$ are valid solutions with a beauty of 1.

In the third test case, if $b = [1, 2, 1, 4, 1, 2, 1, 1, 1, 2]$ with $d_1(b) = 9 - 1 = 8$, $d_2(b) = 10 - 2 = 8$, and $d_4(b) = 0$, resulting in a beauty of 16.

,

F. Mani and Segments

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

An array b of length $|b|$ is *cute* if the sum of the length of its Longest Increasing Subsequence (LIS) and the length of its Longest Decreasing Subsequence (LDS)* is **exactly** one more than the length of the array. More formally, the array b is cute if $\mathrm{LIS}(b) + \mathrm{LDS}(b) = |b| + 1$.

You are given a permutation a of length n†. Your task is to count the number of non-empty subarrays‡ of permutation a that are cute.

---

*A sequence x is a subsequence of a sequence y if x can be obtained from y by the deletion of several (possibly, zero or all) element from arbitrary positions.

The longest increasing (decreasing) subsequence of an array is the longest subsequence such that its elements are ordered in strictly increasing (decreasing) order.

†A permutation of length n 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).

‡An array x is a subarray of an array y if x can be obtained from y by the deletion of several (possibly, zero or all) elements from the beginning and several (possibly, zero or all) elements from the end.

Input

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

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

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

It is guaranteed that the sum of n over all test cases does not exceed $2 \cdot 10^5$.

Output

For each test case, output the number of cute non-empty subarrays of permutation a.

Standard Input	Standard Output
5 3 3 1 2 5 2 3 4 5 1 4 3 4 1 2 7 1 2 3 4 5 6 7 10 7 8 2 4 5 10 1 3 6 9	6 15 9 28 36

Note

In the first test case, all of the 6 non-empty subarrays are cute:

- $[3]$: $\operatorname{LIS}([3]) + \operatorname{LDS}([3]) = 1 + 1 = 2$.
- $[1]$: $\operatorname{LIS}([1]) + \operatorname{LDS}([1]) = 1 + 1 = 2$.
- $[2]$: $\operatorname{LIS}([2]) + \operatorname{LDS}([2]) = 1 + 1 = 2$.
- $[3, 1]$: $\operatorname{LIS}([3, 1]) + \operatorname{LDS}([3, 1]) = 1 + 2 = 3$.
- $[1, 2]$: $\operatorname{LIS}([1, 2]) + \operatorname{LDS}([1, 2]) = 2 + 1 = 3$.
- $[3, 1, 2]$: $\operatorname{LIS}([3, 1, 2]) + \operatorname{LDS}([3, 1, 2]) = 2 + 2 = 4$.

In the second test case, one of the cute subarrays is $[2, 3, 4, 5, 1]$ as $\operatorname{LIS}([2, 3, 4, 5, 1]) = 4$ and $\operatorname{LDS}([2, 3, 4, 5, 1]) = 2$, which satisfies $4 + 2 = 5 + 1$.