# Academy Tasks

As you know in our Academy we give you some problems to solve. You must first solve problem 0. After solving each problem **i**, you must either move on to problem **i+1** or skip ahead to problem **i+2**. You are not allowed to skip more than one problem. For example, {0, 2, 3, 5} is a valid order, but {0, 2, 4, 7} is not because the skip from 4 to 7 is too long.

You are given an array **pleasantness** (0-based), where **pleasantness[i]** indicates how much you like problem **i**. We will let you stop solving problems once the range of pleasantness you've encountered reaches a certain threshold. Specifically, you may stop once the difference between the maximum and minimum pleasantness of the problems you've solved is greater than or equal to the integer **variety**. If this never happens, you must solve all the problems. Return the minimum number of problems you must solve to satisfy our requirements.

## Input

The input data should be read from the console.

On the first input line you will be given the list of numbers in **pleasantness** separated by a comma and a space (see the examples below).

On the second input line you will be given the integer **variety**.

The input data will always be valid and in the format described. There is no need to check it explicitly.

## Output

The output data should be printed on the console.

On the only output line you must print the minimum number of problems you must solve to satisfy our requirements.

## Constraints

- **pleasantness** will contain between 1 and 50 elements, inclusive.
- Each element of **pleasantness** will be between 0 and 1000, inclusive.
- **variety** will be between 1 and 1000, inclusive.
- Allowed working time for your program: 0.1 seconds. Allowed memory: 16 MB.

## Examples

| Input | Output | Explanation |
|---|---|---|
| `1, 2, 3`<br>`2` | 2 | Solve the 0-th problem and the 2-nd after it. |
| `1, 2, 3, 4, 5`<br>`3` | 3 | Obviously, the first and the last problems should be solved. Skip a problem ahead twice in a row. |
| `6, 2, 6, 2, 6, 3, 3, 3, 7`<br>`4` | 2 | You can stop after solving the first 2 problems. |