Tutorial attendance drama!

Input file:	`standard input`
Output file:	`standard output`
Time limit:	0.25 seconds
Memory limit:	256 megabytes

During the online semester, attendance in tutorials has been a major issue for many courses. Students are not attending the tutorials regularly which defeats the aim of practicing questions related to the theory taught during the lecture.

Even after several reminders, students are still not turning up for tutorials. It is decided that in case of low attendance the tutorials will be merged with other groups.

Now you as a student of Data structures and algorithms have to think and code an algorithm which other departments can use when they have to club the tutorial groups in case of low attendance but there is a condition that you can only combine **contiguous groups while combining the tutorial groups.**

You are given an array of integers where ith element shows the attendance of ith group and a number X which tells how many maximum tutorial groups can be formed after combining different groups according to the above mentioned condition.

Keeping in mind that groups don't get overcrowded after clubbing, you have to minimize the largest size of the tutorial group (where size refers to number of students in a group) that will result after all different groups are merged together to X number of groups at the end.

Input

The first line contains an integer n denoting the number of initial groups ($1 \leq n \leq 5 \times 10^4$)

The second line contains n space separated numbers, the attendance of each group. ($1 \leq A[i] \leq 1000$)

The third line contains X denoting the maximum tutorial groups that can be formed after combining different groups. ($1 \leq X \leq n$)

Output

Output a single integer which will be the minimized largest size of the new tutorial group.

Examples

standard input	standard output
8 1 2 3 4 5 6 7 8 3	15
6 1 1 3 1 1 1 4	3

Note

For first Testcase: After combining : 1st group: 1, 2, 3, 4, 5 2nd group: 6, 7 3rd group: 8 Largest Size 1st group = 15

For second Testcase: 1st group: 1, 1 2nd group: 3 3rd group: 1, 1 4th group: 1 Largest Size 2nd group = 3