

We define the following terms:

- [Lexicographical Order](#), also known as *alphabetic* or *dictionary* order, orders characters as follows:

$$\mathbf{A < B < \dots < Y < Z < a < b < \dots < y < z}$$

For example, ball < cat, dog < dorm, Happy < happy, Zoo < ball.

- A [substring](#) of a string is a contiguous block of characters in the string. For example, the substrings of abc are a, b, c, ab, bc, and abc.

Given a string,  $s$ , and an integer,  $k$ , complete the function so that it finds the lexicographically *smallest* and *largest* substrings of length  $k$ .

### Input Format

The first line contains a string denoting  $s$ .

The second line contains an integer denoting  $k$ .

### Constraints

- $1 \leq |s| \leq 1000$
- $s$  consists of English alphabetic letters only (i.e., [a-zA-Z]).

### Output Format

Return the respective lexicographically smallest and largest substrings as a single newline-separated string.

### Sample Input 0

```
welcometojava
3
```

### Sample Output 0

```
ava
wel
```

### Explanation 0

String  $s = \text{"welcometojava"}$  has the following lexicographically-ordered substrings of length  $k = 3$ :

**["ava", "com", "elc", "eto", "jav", "lco", "met", "oja", "ome", "toj", "wel"]**

We then return the first (lexicographically smallest) substring and the last (lexicographically largest) substring as two newline-separated values (i.e., `ava\nwel`).

The stub code in the editor then prints `ava` as our first line of output and `wel` as our second line of output.