A sequence is called an increasing sequence if each character in the sequence is strictly greater than the one before it (if there is one). For example, "abcdgz" is an increasing sequence and "abccd" and "abcda" are not increasing sequences. The comparison of the character is by the ASCII values. Similarly a sequence is called a decreasing sequence if each character in the sequence is strictly smaller than the one before it (if there is one). Now, a sequence is called a "V" sequence is if it is composed of a decreasing sequence followed by an increasing sequence with one character of overlap. For example, "zyxabcd" is a "V" sequence because it is made up of a decreasing sequence "zyxa" followed by an increasing sequence "abcd" where the two sequences overlap at the character "a".

Your goal is to find one of the longest “V” sequences which are substrings from the input string:

Your program should be able to read an input file and run as follows. The output should be written to **console**.

Java: java LongestSequenceV <input file name>

C++: ./LongestSequenceV <input file name>

More examples (the input file only has 1 line in it):

**Input File1**

abcdabc

**Output1**:

abcd

(The left side is "a", the right side "abcd", and the overlap is "a". Another solution is “dabc”. Note you only need to print one of the longest v-sequence)

**Input File2**:

addebbbc

**Output2**:

eb

(The left side is “e”, and the right side is “eb”. Other solutions: “ad”, “de”, “eb”, “bc”)

**Input File3**:

abababab

**Output3**:

bab

**Input File4**:

aaaa

**Output4**:

a

(The left side is “a”, and the right side is “a”)