Problem C
Reversible Compression

Time Limit: 2 seconds

Data compression is an essential technology in our information society. It is to encode a given string into a (preferably) more compact code string so that it can be stored and/or transferred efficiently.

You are asked to design a novel compression algorithm such that even a code string is *reversed* it can be decoded into the given string.

Currently, you are considering the following specification as a candidate.

- A given string is a sequence of decimal digits, namely, `0`, `1`, `2`, `3`, `4`, `5`, `6`, `7`, `8`, and `9`.
- A code string is a sequence of code words, and a code word consists of two decimal digits `A` and `L`. So, a code string is a sequence of even number of decimal digits.
- A code string $\mathtt{A}_1\mathtt{L}_1 \cdots \mathtt{A}_k\mathtt{L}_k$ is decoded into a string by the following procedure. Here, for brevity, a decimal digit ($\mathtt{A}_i$ or $\mathtt{L}_i$) is also treated as the single digit decimal integer it represents.

  1. $i \leftarrow 1$
  2. while (i is not greater than k) {
  3. if ($\mathtt{A}_i$ is zero) { output $\mathtt{L}_i$ }
  4. else if ($\mathtt{L}_i$ is zero) { do nothing }
  5. else if ($\mathtt{A}_i$ is larger than the number of digits output so far) { raise an error }
  6. else {
  7. repeat $\mathtt{L}_i$ times {
  8. output the $\mathtt{A}_i$-th of the already output digits counted backwards
  9. }
  10. }
  11. $i \leftarrow i + 1$
  12. }

For example, a code string `000125` is decoded into a string `0101010` as follows:

1. The first code word `00` outputs `0`.
2. The second code word `01` outputs `1`.

3. The first digit `2` of the last code word `25` means that the second digit in the already decoded digits, counted backwards, should be output. This should be repeated five times. For the first repetition, the decoded digits so far are `0` and `1`, and thus the second last digit is `0`, which is output. For the second repetition, digits decoded so far are `0`, `1`, and `0`, and the second last is `1`, which is output. Repeating this three more times outputs `0`, `1`, and `0`.

A sequence of code words that raises no error is a valid code string. A valid code string is said to be *reversible* when its reverse is also valid and both the original and its reverse are decoded into the same string.

For example, the code string `000125` is not reversible, because its reverse, `521000`, raises an error and thus is not valid. The code string `0010` is not reversible though its reverse is valid. It is decoded into `0`, but its reverse `0100` is decoded into `10`. On the other hand, `0015599100` is reversible, because this and its reverse, `0019955100`, are both decoded into `00000000000000000`.

You want to evaluate the performance of this compression method when applied to a variety of cases. Your task is to write a program that, for an arbitrary given digit string, finds the shortest reversible code string that is decoded into the given string.

Input

The input consists of a single line containing a non-empty string s of decimal digits. The length of s does not exceed 500.

Output

Output the shortest reversible code string that is decoded into s. If there are multiple solutions with the same shortest length, output the earliest in the lexicographic order. Note that it is easily shown that a reversible code string always exists for any input string (e.g., see Sample Output 4).

Sample Input 1	**Sample Output 1**
00000000000000000	0015599100

Sample Input 2	**Sample Output 2**
0101010	000122221000

Sample Input 3	**Sample Output 3**
123123123123123123123123123123	01020336699993302010

Sample Input 4	**Sample Output 4**
123456789	0102030405060708099080706050403020 10