Task: ENCODING
Encoding

day 3, source file `encoding.*`, available memory 256 MB *CPSPC 2004, June 10*

In most programming languagues, string constants can contain characters not accessible directly from the keyboard. This can be achieved by using a special *marker* symbol, usually a backslash. Sometimes there are more special symbols having different functions. Most of the string is read *verbatim*, as it is, and the rest is *interpreted* in one way or another.

Prof. P. Entropic, a famous computer scientist, has his own ideas on this. He believes that using a fixed symbol (or a fixed set of symbols) as a marker is extremely inflexible and cannot be the best solution. For what if you need to encode a string with many backslashes?

Professor has invented a *dynamic* coding method, in which verbatim and interpreted parts of the string are declared explicitly. The idea is to introduce an additional marker, which marks the beginning and the end of every verbatim part of the string. The symbol that is used for this marker is dynamic, i.e. it can change inside the string. At the beginning of the string there is no marker and the symbols are interpreted. As soon as some symbol `x` appears twice in a row, it becomes the special marker and the `xx` just read is discarded. From now on, every appearance of `x` switches between the verbatim mode and the interpreted mode. If some other symbol `y` appears twice in a row anywhere in the interpreted mode, the marker is changed to `y`, and so on.

For example the string `''cccabcabcaacbbcaac''` is read as `''abABaaCC''`, where the capitalized part of the string is interpreted and the rest is read verbatim.

Prof. Entropic needs to know how efficient dynamic coding really is in terms of space. He has gathered a representative sample of strings with parts marked “verbatim” and “interpreted”. He wants to find the shortest possible encodings of these strings using the dynamic coding method. Your task is to write a computer program that does exactly that.

Task

Write a program, that:

- reads the size of the alphabet and a string from the input file,
- finds the shortest encoding of this string, and
- writes this encoding to the output file.

Input

The first line of the input file `encoding.in` contains a single integer k, $2 \le k \le 26$ — the size of the alphabet. The alphabet consists of the first k letters of the standard english alphabet. The second line contains a string to be encoded. The part to be read verbatim is written with small letters and the part to be interpreted is written with capital letters. The length of the string is at most 100 000 characters.

Output

The first and only line of the output file `encoding.out` should contain a single string — (one of) the shortest encoding(s) of the input string using the given alphabet.

Example

For the input file `encoding.in`:

```
3
abABaaCC
```

one of possible correct answers is the file `encoding.out`:

```
cccabcabcaacbbcbbc
```