Problem E

Shortest Superstring

The shortest superstring problem belongs to the NP-Hard class and is defined as follows: given a set of strings S where no element is substring of another element, find the shortest string s that contains each string in S as substring. Formally, let $S = \{s_1, s_2, \ldots, s_n\}$ be a set of strings such that

$$(\forall s_i, s_j \in S)s_i \not\subseteq s_j.$$

Thus, the shortest substring problem is to find a string

$$s \in S' = \{s' | (\forall s_i \in S)s_i \subseteq s'\} \text{ such that } (\forall s' \in S')|s| \leq |s'|.$$

For instance, consider the following set of strings:

$$S = \{\texttt{CATGC, CTAAGT, GCTA, TTCA, ATGCATC}\}.$$

The shortest string that contains all the above strings as a substring is

$$s = \texttt{GCTAAGTTCATGCATC}.$$

The proposed solution employs a greedy strategy over an operation called "overlap", described next.

The *overlap* operation over strings a and b is their concatenation ab where the matching parts in the suffix of a and the prefix of b are merged. For instance, if $a = \{\texttt{ABC}\}$ and $b = \{\texttt{BCDE}\}$, the left overlap of a and b is $\{\texttt{ABCDE}\}$. The overlap value of the operation is the size of the corresponding suffixes/prefixes of both strings. In the example, the overlap value is 2 ($\texttt{BC}$). Order matters; the overlap operation and its respective overlap value are not commutative.

With the overlap operation defined we can now show the greedy algorithm that solves the problem at hand. Its idea is simple; at each step, we replace the two strings whose resulting overlap value would be the highest by the result of their overlap. At the end, we have the shortest superstring:

1. Let T be a copy of S.
2. While $|T| > 1$ do:
   a. let a and b be the two strings that yield the highest overlap value;

b. pop *a* and *b* from T and insert the string obtained by overlapping *a* and *b*.

Now the only element of *T* is the shortest superstring containing as substrings all strings of *S*. It is important to notice the overlap of strings *x* and *y* is probably not the same as the overlap of strings *y* and *x* because of the its aforementioned lack of commutativity. Thus, both configurations must be evaluated for each pair of strings taken into account and its order preserved.

You shall submit a parallel implementation of an algorithm that solves the shortest superstring problem.

Input

The first line contains the number *n* of strings to be read and processed. Each of the following *n* lines of the input contains a string at most 256 ASCII characters. All the characters are readable and there are no blanks.

The input must be read from the standard input.

Output

It is a single line containing the shortest superstring that contains all strings from the input.

The output must be written to the standard output.

Example

Input	Output for the input
5 CATG CTAAGT GCTA TTCA ATGCATC	GCTAAGTTCATGCATC