

NOI '22 Multi-Provincial Team Selection P1 - Preprocessor

Time limit: 1.0s **Memory limit:** 512M

Implement a macro preprocessor. Specifically,

- Code consists of a sequence of lines, terminated by newlines and otherwise only containing printable ASCII characters. A line beginning with `#` is a preprocessor directive; other lines are text.
- The preprocessor will parse each line in the order of input. A preprocessor directive will be executed and converted into an empty line, and a text line will be expanded by the defined macros.
- The first preprocessor directive is `#define`, formatted as `#define <name> <content>`, where `<name>` is one symbol corresponding to the macro to be defined and `<content>` is a string corresponding to the content this macro shall be expanded to.
- The second preprocessor directive is `#undef`, formatted as `#undef <name>`, where `<name>` is one symbol corresponding to a previously defined macro to be deleted.

A symbol is a substring **of maximal length** consisting of alphanumeric characters and underscores. For example, in the code `A_B*C`, `A_B` is a symbol but `_B` **is not**. Therefore, if `A_B` is defined as `x` and `_B` is defined as `y`, `A_B*C` is expanded as `x*C`.

Macro expansion occurs recursively but not infinitely. Specifically, the parser will attempt to parse each token of a line, and:

- When a token is not defined as a macro, it is parsed as itself.
- When a token is defined as a macro, the parser will begin to expand this token by replacing this token with the series of tokens that it is defined as.
- This in turn may expand into more tokens, but a token that is one of the tokens that are currently being expanded will not be expanded further.

For example, if `A` is defined as `B+a` and `B` is defined as `A+b`, then `A` will be expanded as `A+b+a`.

For further explanation, refer to the samples.

Constraints

There are no more than 100 lines of 100 characters each. Each expanded line has length not more than 1000 characters each. **The given code is valid.**

For 20% of test cases, there is no `#define`.

For another 20% of test cases, there is neither recursive expansion nor `#undef`.

For another 20% of test cases, there is no recursive expansion.

For another 20% of test cases, there is no infinite expansion.

Input Specification

The first line contains one integer n , the number of lines of code.

The next n lines describe the code.

Output Specification

Output n lines, representing the code after macro expansion.

Sample Input

```
5
#define BEGIN {
#define END }
#define INTEGER int
class C BEGIN INTEGER x; END;
INTEGER main() BEGIN C c; END
```

Sample Output

```
class C { int x; };
int main() { C c; }
```

Attachments
