UVa Online Judge

625 Compression

There are many various compression method with character string compression among them. It is usually used for text compression, especially for program source codes. Source code usually consists of repeating identifiers and keywords. A simple compression method exchanges these repeating words with their codes.

Your task is to write a compression program using a below-defined method:

- keywords like: `var, end, for, then, else, case, goto, const, label, while, begin, until, repeat, downto, function, procedure` are coded as `&`X where X is equal to `0` for *var*, `1` for *end*, and so on up to `15` for `procedure`.
- identifiers are replaced by `&`X where X is a number of the identifier + 15, so the first identifier in the text is given the code &16.
- identifiers are defined as any string of digits and letters (i.e. `number`, `number33`, `20001`, `2Pi`, `h2o`). Identifiers are separated by whitespaces (not alphanumeric)
- identifiers which are shorter than 3 characters are not coded
- to make decompression possible first occurrence of an identifier is not coded but the appropriate code is stored in memory
- no identifier is longer than 39 characters
- there are at most 2000 identifiers
- character `&` does not appear in the input source code

Example: identifier '`integer`' from the source code listed below is given a code &(3+15) = &18 because '`var`' is a keyword and '`n`' is too short to be coded, so both are not counted.

Input

The first line of the input is an integer N, then a blank line followed by N datasets. There is a blank line between datasets.

Each dataset contains an unlimited number of lines of source code (the last line being '`end.`').

Output

Compressed source code.

Sample Input

```
1

program Test;
var n :integer;

function harmonic(number :integer):real;
var i :integer;
```

```
    result :real;
begin
  result := 0;
  for i := 1 to number do
  begin
    number := number + 1/i;
  end;
  harmonic := result;
end;

begin
  writeln('Get n:');
  readln(n);
  writeln('harmonic number for n: ');
  writeln(harmonic(n));
end.
```

Sample Output

```
program Test;
&0 n :integer;

&14 harmonic(number :&18):real;
&0 i :&18;
    result :&21;
&10
  &22 := 0;
  &2 i := 1 to &20 do
  &10
    &20 := &20 + 1/i;
  &1;
  &19 := &22;
&1;

&10
  writeln('Get n:');
  readln(n);
  &23('&19 &20 &2 n: ');
  &23(&19(n));
&1.
```