Czech ACM Student Chapter
Charles University in Prague
Slovak University of Technology
University of Žilina
Matej Bel University in Banská Bystrica

Czech Technical University in Prague
Technical University of Ostrava
Pavol Jozef Šafárik University in Košice
Masaryk University
University of West Bohemia

CTU Open Contest 2012 — Practice Session

Software Bugs

`bugs.c`, `bugs.cpp`, `Bugs.java`

Not all bugs are insects. There are also certain kinds of bugs that may become a real nightmare for software developers. You definitely know the situation when a user calls to say "Hey, I've found a bug in your program." Once you have located and removed the bug, another one appears immediately. It is a hard and never-ending process.

Recently, there appeared a promising open-source initiative called the *Bug Preprocessor*. The preprocessor is a program able to find all bugs in your source code and mark them, so they are relatively easy to remove. Your task is to write a program that will remove all marked bugs from the preprocessed source code.

Input Specification

The input contains several test cases. Each test case starts with a line containing one integer number T ($0 \leq T \leq 1000$), one space and a string B used by the preprocessor to mark all bugs. The next T lines then contain the preprocessed source code. All bugs are represented by a case-sensitive string B.

Each line of the input will be between 0 and 200 characters long. The bug marker B consists of at least 1 and at most 10 uppercase letters ("`A`" through "`Z`").

Output Specification

Your program must remove all of the bugs from the input and print a text that does not contain any occurrence of B. Nothing else than bugs may be removed, not even spaces.

Sample Input

```
7 BUG
print "No bugs here..."

void hello() {
BUGBUG
  printfBUG("Hello, world!\n");
}

1 ERR
wriERRERRtelERRn("Hello E-R-R");
```

Output for Sample Input

```
print "No bugs here..."

void hello() {

  printf("Hello, world!\n");
}

writeln("Hello E-R-R");
```

(This problem has been adapted from SWERC archives.)