Bottom Coder (Easy)

Some of you may be familiar with the TopCoder (TM) contest. Its exact rules are not important for this problem, but know that the most important part of it is writing a program according to the given specification. Many times the contestant ends up with a program which would work perfectly – if only he could change a couple of characters (like, replacing "=" by "==" in C, etc.). Unfortunately, even the best programmers sometimes aren't able to spot these tiny but necessary changes until it's too late... and that's why we developed a brand-new BottomCoder training for them!

The idea is very simple – you're given a problem specification, a source code, and a list of permitted modifications. Your task is to find a modification which would cause the program to behave according to the specification.

Specification: "Write a program which outputs EXACTLY 42 asterisks and NOTHING more (e.g. NO end-of-line markers, like "\n", ...)"

The code you are supposed to modify:

```
int i, n=42;
main() {
  for(i=0; i<n; i--) {
    printf("*");
  }
}
```

As this is a really, really simple problem, you are only permitted to make exactly ONE of these modifications to the source: 1) Add one character to the source. 2) Delete one character from the source. 3) Replace one character in the source by a different one.

Moreover, it would be definitely too easy if we asked you to find just one solution, so you'll need to find TWO DIFFERENT solutions in order to obtain credit for this problem. (There are exactly three different solutions, so don't worry, it can be done!)

Input

There is no input for given problem.

Output

Your submission should consist of two parts. The first part should contain the first of your solutions. A single line with the letter "Q" follows. (Note that the letter Q is used as a separator. You will have to do without inserting the letter Q in at least one of your solutions :) After this line you should add your second solution.

You don't need to worry much about the exact formatting of your submission. The exact judging procedure will look as follows:

The first occurrence of the letter Q is found, the input is split into two parts. Any whitespace in each of the parts is removed. It is checked whether the two submissions differ and whether each of them was obtained from the original program by an allowed change. Each of your two submissions is compared to each of the three correct solutions.

Example

Output:

```
int i, n=42; main(
){ for(i=0; i<n; i--)   { printf("?"); } }
Q
int i, n=41; main() { for(i=0; i<n;i--) { printf("*"); } }
```

(syntactically valid (but incorrect) submission)