Problem 4—Bewitching Logic

If a witch weighs the same as a duck, she must be able to float like a duck. Since wood also floats, and you burn wood, you should burn the witch, if she weighs the same as a duck.

Well, logic can be confusing. The classic rule of *modus ponens* is that if you know "if P then Q" (written P==>Q) and you know P, then Q must also be true. However if you have P==>Q and Q, you *cannot* derive anything about P. Given a series of logical implications and one known true statement, you must decide whether the other statements are true or false (or unknown).

INPUT SPECIFICATION. Each case begins with two unsigned decimal integers, representing the number of logical variables and the number of statements. The integers will be separated by one space and followed by **<EOLN>**. The logical variables are just integers themselves numbered from 0 to one less than the number of variables. The individual statements will follow, each on its own line. They will follow the format as specified below. Sometimes a variable will appear alone within the statement and sometimes it will be preceded by !, signifying the negation of that variable. Following the statements will be one variable number stipulated to be true, followed by **<EOLN>**. "0 0**<EOLN>**" will follow the last data case. It is not to be processed; it just represents the end of input.

OUTPUT SPECIFICATION. Each output case should appear in the same order as the corresponding input case. The output case should begin "Case *c*:**<EOLN><EOLN>**". (*c* is the case number.) Following this is the list of true, false, and undefined variables. Follow the example below in the output. Note that the variables are listed in ascending order and each is followed by a space. Note that there may be a logical contradiction in the given statements (meaning that there is no possible way all of those statements could be true). If there is, that should be noted as well, as demonstrated in the sample below.

SAMPLE INPUT.

```
5·2<EOLN>
3==>4<EOLN>
4==>!1<EOLN>
3<EOLN>
1·1<EOLN>
0==>!0<EOLN>
0<EOLN>
1·1<EOLN>
!0==>0<EOLN>
0<EOLN>
0·0<EOLN>
<EOF>
```

SAMPLE OUTPUT.

```
Case·1:<EOLN>
<EOLN>
TRUE<EOLN>
3·4·<EOLN>
<EOLN>
FALSE<EOLN>
1·<EOLN>
<EOLN>
UNDEFINED<EOLN>
0·2·<EOLN>
<EOLN>
Case·2:<EOLN>
<EOLN>
CONTRADICTION<EOLN>
<EOLN>
Case·3:<EOLN>
<EOLN>
TRUE<EOLN>
0·<EOLN>
<EOLN>
FALSE<EOLN>
<EOLN>
<EOLN>
UNDEFINED<EOLN>
<EOLN>
<EOLN>
<EOF>
```