A - Comparison Expressions

Automated Compilers Manufacture, is developing a new, intelligent compiler. One of the problems its scientists are facing is how to determine whether two expressions are equivalent. An expression consists of one or more alphabetic lowercase letters, representing *variables*, separated by addition and multiplication signs, respectively + and *, totally parenthesized. Recursively, an expression is either a variable, or a string of the form (E1 + E2) or (E1 * E2), where E1 and E2 are both expressions. Examples of expressions: a, (a+b), ((a*b)*b). Examples of non-expressions: ab, a*(b+c).

Two expressions are *equivalent* if, for every integer assignment to their variables, both expressions assume the same value. For example, ((a+b)*c) and ((b*c)+(c*a)) are equivalent, whereas (a+(b*c)) and ((a+b)*c) are not equivalent. Your task is to write a program that decides whether two given expressions are equivalent.

Input

The input contains several test cases, each case consists of one line, containing two expressions separated by precisely one space. The variables of the expressions are represented by lowercase alphabetic letters. The number of occurrences of variables in an expression is at most 20 (for example, ((a+b)*a) has three occurrences of variables). Thus, an expressions consists of at most 77 characters, including variables, parentheses, plus and multiplication signs. The end of the input is indicated by a line containing only one zero.

Output

For each test case in the input, your program must print a single line, containing one single character, a Y if the two expressions are equivalent, an N otherwise.

Sample Input

(a+(b*c)) ((a+b)*c)

((a+b)*c) ((c*b)+(a*c))

0

Sample Output

N

Y