Problem H

Firefighters

Input File: H.IN

Program Source File: H.PAS, H.C, H.JAVA or H.CPP

A mathematician kept all his documents in a file cabinet near his desk. One day a fire set out in his office and most of his works were badly damaged. Luckily, some of the equations he had solved so many years during his long career were partially preserved. Each equation contained an expression in left side and result in the right. The preserved expressions consisted of all the numbers and brackets, but unfortunately, some of the operators between them were lost in the fire. Another problem was that the results of the equations were scattered and the mathematician was not sure if a certain answer is the result of a certain expression. Your task is to help the mathematician determine if the expressions and the results he was able to save from the fire match one another.

In order to do this, you are given an expression, containing the integer numbers between 1 and 999, simple mathematical operators (+, -, *, /), brackets, and question marks (?), representing the lost mathematical operators. For every expression given, your only task is to state if an expression can or cannot give the required result. In order to help you, the mathematician has chosen only expressions that have the following restrictions:

1. The expressions contain no more than 100 symbols;
2. The brackets enclose no more than 1 operator with his two operands. However, every one of these operands can be an expression in brackets;
3. The constants in the expressions have no sign, i.e. there are no negative numbers in expressions;
4. The maximum number of question marks in the expressions (the lost operators) is less or equal to 10.

The calculation should be performed using the following rules:

1. The operators * and / are of higher priority than the operators + and -. Parentheses may change the priorities as usually;
2. The operators +, -, *, and / are left associative, meaning that they group from left to right. If a, b and c are numbers:
   a*b*c = (a*b)*c, a/b/c = (a/b)/c, a/b*c = (a/b)*c, a+b+c = (a+b)+c, a-b+c = (a-b)+c, etc.
3. When dividing two integers, you should ignore the decimal fraction, for example consider the following equations: 2/5=0, 9/5=1, 100/6=16.

Input: The first line of the input file contains an integer **N** – determining the number of equations. Next **2*N** lines contain the equations. One equation is defined in two lines. The first line is the expression, defining the left side of the equation; second line is an integer result, defining the right side of the equation. The input lines do not have blanks. The strings representing expressions are guaranteed to have no syntax errors.

Output: For every equation in the input file, write **yes** or **no** on separate lines on the standard output. If the expression can give the result, write yes. Otherwise, if the result cannot be achieved, write no.

Sample input and output:

Sample Input	Sample Output
3 1?((2*(3*4))+(5+6)) 35 1?2*3+4-14 0 1?3*4/5*6+12 11	yes no no