Question 2: *Rules*

Some computer programs try to prevent easily guessable passwords being used, by rejecting those that do not meet some simple rule. Your task for this question is to determine if a password is accepted by a rule.

The only characters that can occur in a password are the digits 0 to 9.

A rule is made up from a sequence of symbols, each with a special meaning. Symbols can be combined to make more complicated rules, and are written in order. Rules are described as follows:

- The symbol `x` means any digit is acceptable.

  For example: The rule `x` means the password has to be a single digit. The rule `xxx` means the password can be any combination of three digits.

- The symbol `u` means any digit that is higher than the previous digit.
- The symbol `d` means any digit that is lower than the previous digit.

  For example: The rule `xu` means the password has exactly two digits, the second of which is higher than the first; 46 would be accepted, 64 or 66 would be rejected. The rule `xud` means the password has exactly three digits, the second of which is higher than the first, and the third of which is lower than the second; 040 would be accepted.

Combinations of the `x`, `u` and `d` symbols (and only these symbols) can be placed inside brackets without changing their meaning; brackets must contain some symbols. The following two symbols can only follow an `x`, `u`, `d` or bracketed expression. If it is an `x`, `u` or `d` they change the meaning of the previous symbol. If they follow a bracketed expression, they change the meaning of everything inside the brackets.

- The symbol `?` means ignore or apply once.

  For example: The rule `xxx?` means the password must match either the rule `xx` or the rule `xxx`. The rule `x(xx)?` means the password must match either the rule `x` or the rule `xxx`.

- The symbol `*` means ignore, or repeat any number of times.

  For example: The rule `x(xu)*` means the password must match one of the rules `x`, `xxu`, `xxuxu`, `xxuxuxu`, etc...

Some combinations of these symbols are invalid, such as those that break any of the above conditions or, if for some interpretation of their `*` and `?` symbols, are equivalent to an invalid combination. For example, the rule `u` is invalid since there is no previous digit. The rule `x?u` is invalid, since it means match either the rule `xu` or the rule `u`, and the rule `u` is invalid. The rule `(x(ud)*)*` is invalid because the only valid symbols within a pair of brackets are `x`, `u` and `d`.

You will not be given any invalid rules.

2(a) [24 marks]

Write a program to test passwords.

Your program should first read in a single line, containing the rule; this rule will have between 1 and 12 symbols (inclusive, and including any brackets). You should then read in two more lines, each of which will contain a single password; these passwords will contain between 1 and 12 digits (inclusive).

No rule will contain more than two `?` or `*` symbols (combined).

You should output two lines, the first line indicating whether the first password is accepted by the rule, and the second line indicating whether the second password is accepted by the rule. Your should output **`Yes`** if a password is accepted and **`No`** if the password is rejected.

Sample run

```
xu*
02468
4688

Yes
No
```

2(b) [3 marks]

Consider the rule `x(xd)?(xx)?`. How many different passwords will be accepted this rule?

2(c) [3 marks]

A zig-zag password is one where the digits alternatively rise and fall, always rising initially if the password has more than one digit. For example, 5 and 08362 are zig-zag passwords. Find the rule that only matches zig-zag passwords and uses the smallest number of symbols. (No justification is necessary.)

2(d) [5 marks]

Is there a rule, containing exactly 11 symbols, that can only match a single password? Justify your answer.