Given a string s containing only three types of characters: '(', ')' and '\*', return true if s is valid.

The following rules define a valid string:

Any left parenthesis '(' must have a corresponding right parenthesis ')'.

Any right parenthesis ')' must have a corresponding left parenthesis '('.

Left parenthesis '(' must go before the corresponding right parenthesis ')'.

'\*' could be treated as a single right parenthesis ')' or a single left parenthesis '(' or an empty string "".

Example 1:

Input: s = "()"

Output: true

Example 2:

Input: s = "(\*)"

Output: true

Example 3:

Input: s = "(\*))"

Output: true

Constraints:

1 <= s.length <= 100

s[i] is '(', ')' or '\*'.

Solution

class Solution {

public:

    bool checkValidString(string s) {

        int low = 0; // minimum number of open parenthesis

        int high = 0; // maximum number of open parenthesis

        for (char c : s) {

            low += (c == '(') ? 1 : -1;

            high += (c != ')') ? 1 : -1;

            if (high < 0) break; // We have more closing parenthesis than open

            low = max(low, 0); // We cannot have less than 0 open parenthesis

        }

        return low == 0; // Valid string if we can have exactly 0 open parenthesis

    }

};