**Task 3**

**Balanced brackets**

A bracket is considered to be any one of the following characters: (, ), {, }, [, or ].

Two brackets are considered to be a *matched pair* if the an opening bracket (i.e., (, [, or {) occurs to the left of a closing bracket (i.e., ), ], or }) *of the exact same type*. There are three types of matched pairs of brackets: [], {}, and ().

A matching pair of brackets is *not balanced* if the set of brackets it encloses are not matched. For example, {[(])} is not balanced because the contents in between { and } are not balanced. The pair of square brackets encloses a single, unbalanced opening bracket, (, and the pair of parentheses encloses a single, unbalanced closing square bracket, ].

By this logic, we say a sequence of brackets is *balanced* if the following conditions are met:

* It contains no unmatched brackets.
* The subset of brackets enclosed within the confines of a matched pair of brackets is also a matched pair of brackets.

Given  strings of brackets, determine whether each sequence of brackets is balanced. If a string is balanced, return YES. Otherwise, return NO.

**Function Description**

Complete the function *isBalanced* in the editor below.

isBalanced has the following parameter(s):

* *string s*: a string of brackets

**Returns**

* *string:* either YES or NO

**Input Format**

The first line contains a single integer , the number of strings.  
Each of the next  lines contains a single string , a sequence of brackets.

**Output Format**

For each string, return YES or NO.

**Sample Input**

STDIN Function ----- -------- 3 n = 3 {[()]} first s = '{[()]}' {[(])}

second s = '{[(])}' {{[[(())]]}} third s ='{{[[(())]]}}'

**Sample Output**

YES

NO

YES

**Explanation**

1. The string {[()]} meets both criteria for being a balanced string.
2. The string {[(])} is not balanced because the brackets enclosed by the matched pair { and } are not balanced: [(]).
3. The string {{[[(())]]}} meets both criteria for being a balanced string.