**Problem**

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 considered to be *balanced* if the following conditions are met:

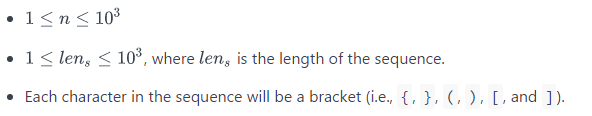
* It contains no unmatched brackets.
* The subset of brackets enclosed within the confines of a matched pair of brackets is also a matched pair of brackets.

Given n strings of brackets, determine whether each sequence of brackets is balanced. If a string is balanced, print YES on a new line; otherwise, print NO on a new line.

**Input Format**

The first line contains a single integer, n, denoting the number of strings.   
Each line i of the  n subsequent lines consists of a single string, s, denoting a sequence of brackets.

**Constraints**



**Output Format**

For each string, print whether or not the string of brackets is balanced on a new line. If the brackets are *balanced*, print YES; otherwise, print NO.

**Sample Input**

3

{[()]}

{[(])}

{{[[(())]]}}

**Sample Output**

YES

NO

YES

**Explanation**

1. The string {[()]} meets both criteria for being a balanced string, so we print YES on a new line.
2. The string {[(])} is not balanced, because the brackets enclosed by the matched pairs [(] and (]) are not balanced.
3. The string {{[[(())]]}} meets both criteria for being a balanced string, so we print YES on a new line.