B. Down with Brackets

time limit per test: 1 second
memory limit per test: 256 megabytes

In 2077, robots decided to get rid of balanced bracket sequences once and for all!

A bracket sequence is called *balanced* if it can be constructed by the following formal grammar.

1. The empty sequence $\varnothing$ is balanced.
2. If the bracket sequence A is balanced, then (A) is also balanced.
3. If the bracket sequences A and B are balanced, then the concatenated sequence AB is also balanced.

You are the head of the department for combating balanced bracket sequences, and your main task is to determine which brackets you can destroy and which you cannot.

You are given a balanced bracket sequence represented by a string s, consisting of the characters (and). Since the robots' capabilities are not limitless, they can remove **exactly** one opening bracket and **exactly** one closing bracket from the string.

Your task is to determine whether the robots can delete such two brackets so that the string s is no longer a balanced bracket sequence.

Input

Each test contains multiple test cases. The first line contains the number of test cases t ($1 \le t \le 10^4$). The description of the test cases follows.

Each test case consists of a single string s ($2 \le |s| \le 2 \cdot 10^5$) — a sequence of the characters (and).

It is guaranteed that s is a balanced bracket sequence.

It is also guaranteed that the sum of $|s|$ across all test cases does not exceed $2 \cdot 10^5$.

Output

For each test case, output "YES" if the robots can make the string stop being a balanced bracket sequence, and "NO" otherwise.

You may output each letter in any case (lowercase or uppercase). For example, the strings "yEs", "yes", "Yes", and "YES" will be accepted as a positive answer.

Example

input Copy

```
4
(())
(())()()
()
((()))
```

output Copy

```
NO
YES
NO
YES
```

Note

In the first test case, it can be shown that the robots will not be able to break the correct bracket sequence.

In the second test case, one of the options for removing brackets is as follows:

(())()() → ((()))(, which is not a correct bracket sequence.

In the fourth test case, one of the options for removal is as follows:

Codeforces Round 1026 (Div. 2)
Contest is running
01:41:06
Contestant

→ Submit?

Language: GNU G++23 14.2 (64 bit, ms
Choose file: Choose File No file chosen
Be careful: there is 50 points penalty for submission which fails the pretests or resubmission (except failure on the first test, denial of judgement or similar verdicts). "Passed pretests" submission verdict doesn't guarantee that the solution is absolutely correct and it will pass system tests.
Submit

→ Last submissions

Submission	Time	Verdict
321071374	May/24/2025 17:51	Pretests passed

→ Score table

	Score
Problem A	466
Problem B	699
Problem C	1398
Problem D	1864
Problem E	2097
Problem F	2796
Successful hack	100
Unsuccessful hack	-50
Unsuccessful submission	-50
Resubmission	-50

* If you solve problem on 00:17 from the first attempt

(())(()) → ())((), which is not a correct bracket sequence.

Codeforces (c) Copyright 2010-2025 Mike Mirzayanov
The only programming contests Web 2.0 platform
Server time: May/24/2025 21:52:14 UTC+7 (k1).
Desktop version, switch to mobile version.
Privacy Policy | Terms and Conditions

Supported by