1. Java Stack

Given a list of strings of bracket characters: *{}()*, the string of brackets is *balanced* under the following conditions:

1. It is the empty string.
2. If strings *a* and *b* are balanced, then *ab* is balanced.
3. If string *a* is balanced, then *(a)* and *{a}* are balanced.

Write a class that determines whether the brackets in each string are balanced and returns *true* if the string is balanced, or *false* if it is not.

Example 0

s = ["*{}()*", "*{()}*", "*({()})*"]

s[0] exhibits condition *2* above. "{}" and "()" are balanced, so "{}()" is balanced. Return *true*.
s[1] exhibits condition 3 above. "()" is balanced, so "{()}" is balanced. Return *true*.
s[2] exhibits condition 3 above. "()" is balanced, so "{()}" is balanced and "({()})" is balanced. Return *true*.

Example 1

s = ["*{}(*", "*(()}*", "*((*", "*}{*"]

s[0] rarr *2*. "*{}(*" is an unbalanced string due to the open "*(*". Return *false*.
s[1] rarr 2. "*(()}*" is an unbalanced string due to ")" before "{" has been closed. Return *false*.
s[2] rarr 2. "*((*", is an unbalanced string because neither "(" is closed. Return *false*.
s[2] rarr 2. "*}{*" is an unbalanced string because "}" comes before a "{" and because the final "{" is not closed. Return *false*.

Function Description

The provided code contains the declaration for a class named *Solution* with a *main* method that does the following:

- Creates a *Parser* object.
- Reads an unknown number of strings from stdin.
- Passes each string as an argument to the *Parser* object's *isBalanced* method and prints value returned by the method on a new line.

Complete the function an *isBalanced* in the editor below.

isBalanced has the following parameter(s):
string s: a string of characters to check for balance
Returns :
bool : a boolean that denotes whether the string is balanced: *true* if the string is balanced, or *false* if it is not

Constraints

- Each string consists only of the characters *{*, *}*, *(*, and *)*.
- Each string has fewer than 50 characters.

▼ Input Format for Custom Testing

Input from stdin will be processed as follows and passed to your Parser.isBalanced method.

Each line contains a string to parse.

▼ Sample Case 0

Sample Input 0

```
STDIN    Function
-----    --------
{}()  →  s = ['{}())', '({()})', '{}(']
({()})
{}(
```

Sample Output 0

```
true
true
false
```

Explanation 0

2. '{}()' contains two adjacent balanced strings, '*{}* 'and '*()*', so return *true*.

3. '({()})' contains a balanced string, '*()*', nested inside another balanced string, '*{}*', nested inside another balanced string, '*()*'. Return *true*.

2. '{}(' contains a balanced string '*{}*', followed by an unbalanced string '*(*'. Return *false*.