Balancing Parentheses

Given a string that consists of left and right parentheses, '(' and ')', balance the parentheses by inserting parentheses as necessary. Determine the minimum number of characters that must be inserted.

**Example**

*s = '(()))'*

Insert *1* left parenthesis at the left end of the string to get '((()))'. The string is balanced after *1* insertion.

**Constraints**

1 ≤ length of *s* ≤ 105

Input Format For Custom Testing

The first line contains a string, *s*, the initial parentheses sequence.

Sample Case 0

**Sample Input**

STDIN Function

-----    -----

()))   →  s = '()))'

**Sample Output**

2

**Explanation**

Insert a '(' *2* times at the beginning of the string to make it valid: '((()))'.

Sample Case 1

**Sample Input**

STDIN Function

-----     -----

()()   →  s = '()()'

**Sample Output**

0

**Explanation**

The sequence is already valid, so no insertions are needed.

import java.io.\*;

import java.math.\*;

import java.security.\*;

import java.text.\*;

import java.util.\*;

import java.util.concurrent.\*;

import java.util.function.\*;

import java.util.regex.\*;

import java.util.stream.\*;

import static java.util.stream.Collectors.joining;

import static java.util.stream.Collectors.toList;

class Result {

/\*

\* Complete the 'getMin' function below.

\*

\* The function is expected to return an INTEGER.

\* The function accepts STRING s as parameter.

\*/

public static int getMin(String s) {

// Write your code here

}

}

public class Solution {

public static void main(String[] args) throws IOException {

BufferedReader bufferedReader = new BufferedReader(new InputStreamReader(System.in));

BufferedWriter bufferedWriter = new BufferedWriter(new FileWriter(System.getenv("OUTPUT\_PATH")));

String s = bufferedReader.readLine();

int result = Result.getMin(s);

bufferedWriter.write(String.valueOf(result));

bufferedWriter.newLine();

bufferedReader.close();

bufferedWriter.close();

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*