

Codility

CodeCheck Report: trainingD3UDWP-CH4

Test Name:

[Check out Codility training tasks](#)

Summary

Timeline

Tasks summary

Task	Time spent	Score
Brackets Java 8	3 min	100%

Total score



Tasks Details

Easy	1. Brackets	Task Score	Correctness	Performance	
	Determine whether a given string of parentheses (multiple types) is properly nested.				
		100%	100%	100%	

Task description

A string *S* consisting of *N* characters is considered to be *properly nested* if any of the following conditions is true:

- *S* is empty;
- *S* has the form "*(**U**)*" or "*[**U**]*" or "*{**U**}*" where *U* is a properly nested string;
- *S* has the form "*∇W*" where *V* and *W* are properly nested strings.

For example, the string "{ [() ()] }" is properly nested but "[() ()]" is not.

Write a function:

```
class Solution { public int solution(String S); }
```

that, given a string *S* consisting of *N* characters, returns 1 if *S* is properly nested and 0 otherwise.

Solution

Programming language used: Java 8

Total time used: 3 minutes ?

Effective time used: 3 minutes ?

Notes: not defined yet

Task timeline



07:37:38

07:40:20

For example, given $S = "[()]"$, the function should return 1 and given $S = "([)]"$, the function should return 0, as explained above.

Write an **efficient** algorithm for the following assumptions:

- N is an integer within the range $[0..200,000]$;
- string S consists only of the following characters: $"(", "{", "[", "]", "}"$ and/or $)"$.

Copyright 2009–2022 by Codility Limited. All Rights Reserved. Unauthorized copying, publication or disclosure prohibited.

Code: 07:40:19 UTC, java,
final, score: 100

[show code in pop-up](#)

```

1 // you can also use imports, for example:
2 // import java.util.*;
3 import java.util.ArrayList;
4 // you can write to stdout for debugging purposes
5 // System.out.println("this is a debug message")
6
7 class Solution {
8     static ArrayList<String> bracketStack = new ArrayList<>();
9
10    public static void push(ArrayList<String> stringList, String s) {
11        stringList.add(s);
12    }
13
14    public static String pop(ArrayList<String> stringList) {
15        return stringList.isEmpty() ? "" : stringList.remove(stringList.size() - 1);
16    }
17    public int solution(String S) {
18        // write your code in Java SE 8
19
20        int output = -1;
21        boolean isNestedString = true;
22        if(!S.isEmpty()) {
23            int index = 0;
24            int size = S.length();
25            boolean continueWhile = true;
26            do {
27                //char currentChar = S.charAt(index);
28                String currentChar = S.substring(index, index + 1);
29
30                switch (currentChar) {
31                    case "(":
32                        if (!bracketStack.isEmpty()) {
33                            bracketStack.add("(");
34                        }
35                        break;
36                    case ")":
37                        if (!bracketStack.isEmpty() && bracketStack.remove(bracketStack.size() - 1).equals("(")) {
38                            continueWhile = true;
39                        } else {
40                            continueWhile = false;
41                        }
42                        break;
43                    case "[":
44                        if (!bracketStack.isEmpty()) {
45                            bracketStack.add("[");
46                        }
47                        break;
48                    case "]":
49                        if (!bracketStack.isEmpty() && bracketStack.remove(bracketStack.size() - 1).equals "[")) {
50                            continueWhile = true;
51                        } else {
52                            continueWhile = false;
53                        }
54                        break;
55                    default:
56                        push(bracketStack, currentChar);
57                        break;
58                }
59                index++;
60            } while (index < size && continueWhile);
61
62            if(isNestedString && bracketStack.isEmpty()) {
63                output = 1;
64            } else {
65                output = 0;
66            }
67
68            return output;
69        }
70    }
71 }
```

70 }

Analysis summary

The solution obtained perfect score.

Analysis

Detected time complexity: **$O(N)$**

expand all	Example tests
▶ example1	✓ OK
example test 1	
▶ example2	✓ OK
example test 2	
expand all	Correctness tests
▶ negative_match	✓ OK
invalid structures	
▶ empty	✓ OK
empty string	
▶ simple_grouped	✓ OK
simple grouped positive and negative test, length=22	
expand all	Performance tests
▶ large1	✓ OK
simple large positive test, 100K '('s followed by 100K ')'s +)	
▶ large2	✓ OK
simple large negative test, 10K+1 '('s followed by 10K ')'s +)(+ ()	
▶ large_full_ternary_tree	✓ OK
tree of the form T=(TTT) and depth 11, length=177K+	
▶ multiple_full_binary_trees	✓ OK
sequence of full trees of the form T=(TT), depths [1..10..1], with/without some brackets at the end, length=49K+	
▶ broad_tree_with_deep_paths	✓ OK
string of the form [TTT...T] of 300 T's, each T being '{{{...}}}' nested 200-fold, length=120K+	