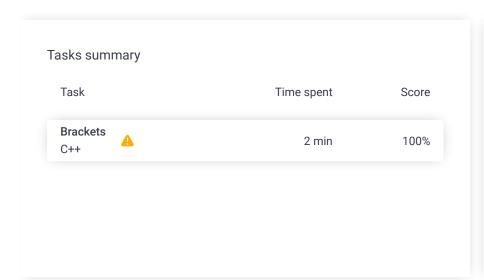
# Codility\_

### CodeCheck Report: training893927-7M3

Test Name:

Summary Timeline

Check out Codility training tasks





#### **Tasks Details**

#### 1. Brackets

Easy

Determine whether a given string of parentheses (multiple types) is properly nested.

Task Score

100%

Correctness

Performance

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 "VW" where V and W are properly nested strings.

For example, the string  $\{(())\}$  is properly nested but (()) is not.

Write a function:

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: C++

Total time used: 2 minutes

Effective time used: 2 minutes

Notes: not defined yet

Task timeline

13:48:14 13:50:10

Test results - Codility

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: 13:50:10 UTC, cpp, show code in pop-up final, score: 100

```
1
 2
     // references:
 3
     // [1] https://en.cppreference.com/w/cpp/container/
 4
     // [2] https://cp-algorithms.com/combinatorics/brac
 5
 6
     #include <iostream>
 7
     #include <stack>
8
9
     int solution(string &S) {
10
       bool balanced = true;
11
       std::stack<char> brackets;
12
13
       for(auto& ch : S)
14
15
         switch(ch)
16
           case '{':
17
18
           brackets.push(ch);
19
           break;
20
           case '(':
21
22
           brackets.push(ch);
23
           break;
24
           case '[':
25
26
           brackets.push(ch);
27
           break;
28
         }
29
30
         switch(ch)
31
         {
32
           case '}':
33
           if(brackets.empty())
34
35
              balanced = false;
36
37
           if(!brackets.empty() && brackets.top() == '{'
38
39
           {
40
             brackets.pop();
41
           break;
42
43
44
           case ')':
45
           if(brackets.empty())
46
47
             balanced = false;
48
49
50
            if(!brackets.empty() && brackets.top() == '('
51
52
             brackets.pop();
53
           }
54
           break;
55
56
           case ']':
57
            if(brackets.empty())
58
            {
59
             balanced = false;
60
61
62
           if(!brackets.empty() && brackets.top() == '['
63
64
             brackets.pop();
65
66
            break;
67
         }
68
       }
69
70
       if(! brackets.empty())
```

#### Test results - Codility

```
71  {
72    balanced = false;
73  }
74    return balanced;
75 }
```

#### Analysis summary

The solution obtained perfect score.

#### Analysis

## Detected time complexity: O(N)

