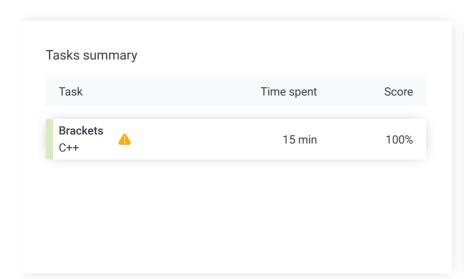
Codility_

CodeCheck Report: training8K58J8-EP6

Test Name:

Check out Codility training tasks

Summary Timeline





Tasks Details

.

1. Brackets

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



Correctness 100%

Performance

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);

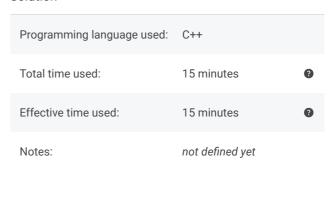
that, given a string S consisting of N characters, returns 1 if S is properly nested and 0 otherwise.

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];

Solution



Task timeline Task timeline

23:23:46

Code: 23:38:20 UTC, cpp, show code in pop-up final, score: 100

1 // you can use includes, for example: 2 // #include <algorithm>

3

Test results - Codility

• string S consists only of the following characters: "(", "{", "[", "]", "}" and/or ")".

Copyright 2009–2021 by Codility Limited. All Rights Reserved. Unauthorized copying, publication or disclosure prohibited.

```
// you can write to stdout for debugging purpo
     // cout << "this is a debug message" << endl;</pre>
 6
     #include <bits/stdc++.h>
 7
     int solution(string &S) {
 8
         vector<int> A;
 9
         int n=S.size(),cont;
10
         if(S.empty()){
11
             return 1;
12
13
         for(int i=0;i<n;i++){</pre>
14
             if(S[i]=='['){
15
                  cont=+1;
16
17
             if(S[i]==']'){
18
                  cont=-1;
19
             }
             if(S[i]=='('){
20
21
                  cont=+2;
22
             }
23
             if(S[i]==')'){
24
                  cont=-2;
25
26
             if(S[i]=='{'){
27
                  cont=+3;
28
             }
29
             if(S[i]=='}'){
30
                  cont=-3;
31
             if((i==0 && cont<0) || (cont<0 && cont
32
33
             else if(cont<0){A.pop_back();}</pre>
34
             else{A.push_back(cont);}
35
36
         if(A.empty()){return 1;}
37
         else {return 0;}
38
39
     }
```

Analysis summary

The solution obtained perfect score.

Analysis

Detected time complexity: O(N)

ехра	nd all	Example tests		
•	example1 example test 1		•	ОК
•	example2 example test 2		•	OK
expand all Correctness tests				
•	negative_match invalid structures		•	OK
•	empty empty string		•	OK
•	simple_grouped simple grouped positi test, length=22		•	OK
ехра	nd all P	erformance tes	sts	s
•	large1 simple large positive t followed by 100K)'s +	est, 100K ('s	1	OK
•	large2 simple large negative		1	OK

Test results - Codility

	followed by 10K)'s +)(+()	
•	large_full_ternary_tree tree of the form T=(TTT) and depth 11, length=177K+	✓ OK
•	multiple_full_binary_trees sequence of full trees of the form T= (TT), depths [1101], with/without some brackets at the end, length=49K+	✓ OK
•	broad_tree_with_deep_paths string of the form [TTTT] of 300 T's, each T being '{{{}}}' nested 200-fold, length=120K+	✓ OK