Valid Substring (ST2)

locked

Submitted a few seconds ago • Score: 1.00 Status: Accepted

✔ Test Case #0	✔ Test Case #1	✔ Test Case #2
✔ Test Case #3	✔ Test Case #4	✔ Test Case #5

Submitted Code

Language: C++ Open in editor

```
1 /*
2     Link: http://www.geeksforgeeks.org/length-of-the-longest-valid-substring/
3     Length of the longest valid substring
```

```
    Given a string consisting of opening and closing parenthesis, find length of the longest valid parenthesis substring.
    Examples:
    Input : (((()
    Output : 2
    Explanation : ()
    Input: )()())
    Output : 4
    Explanation: ()()
    Input:  ()((((()))
    Output: 6
    Explanation:  ()((()))
*/









#include <cmath>
#include <cstdio>
#include <vector>
#include <iostream>
#include <algorithm>
#include <stack>
using namespace std;

// Shortcuts for "common" data types in contests
typedef long long ll;
#define inf 0x3f3f3f3f
#define Inf 0x3FFFFFFFFFFFFFFFLL

```

```
// To simplify repetitions/loops, Note: define your loop style and stick with it!
#define rep(i, n) for (int i = 0; i < (n); ++i)
#define Rep(i, n) for (int i = 1; i <= (n); ++i)
#define REP(i, j, n) for (int i = (j); i<= (n); ++i)
#define clr(x) memset((x), 0, sizeof (x))

using namespace std;


int findMaxLen(string str)
{
    int n = str.length();

    // Create a stack and push -1 as initial index to it.
    stack<int> stk;
    stk.push(-1);

    // Initialize result
    int result = 0;

    // Traverse all characters of given string
    for (int i=0; i<n; i++)
    {
        // If opening bracket, push index of it
        if (str[i] == '(')
          stk.push(i);

        else // If closing bracket, i.e.,str[i] = ')'
        {
            // Pop the previous opening bracket's index
            stk.pop();

            // Check if this length formed with base of
            // current valid substring is more than max
            // so far
```

```
            if (!stk.empty())
                result = max(result, i - stk.top());

            // If stack is empty. push current index as
            // base for next valid substring (if any)
            else stk.push(i);
        }
    }

    return result;
}

// Driver program
int main()
{
    string str ;
    cin>>str;
    cout << findMaxLen(str) << endl;



    return 0;
}
```