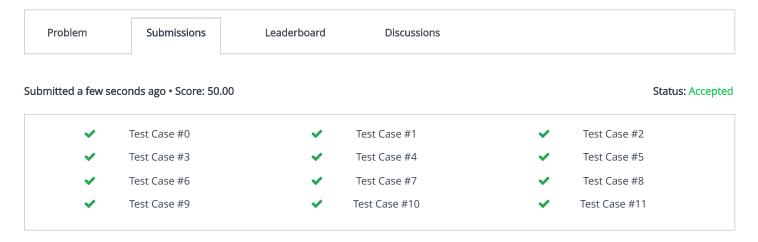
All Contests > In21-CS2023-Lab4 > Recursive Digit Sum

Recursive Digit Sum



Submitted Code

```
Language: C++
                                                                                            Open in editor
 1 #include <bits/stdc++.h>
 3 using namespace std;
 4
 5 string ltrim(const string &);
 6 string rtrim(const string &);
 7 vector<string> split(const string &);
 8
9 /*
10 * Complete the 'superDigit' function below.
11
12 * The function is expected to return an INTEGER.
   * The function accepts following parameters:
13
   * 1. STRING n
15 * 2. INTEGER k
16 */
17
18 int superDigit(string n, int k) {
19
       // calculate the sum of digits in the string
20
       long int sum = 0;
21
       for (char c: n){
22
           sum += c - '0';
23
24
       // multiplying the sum by k
25
       sum *= k;
      // checking if the sum is one digit
26
27
      if (sum < 10) {
28
           return sum;
29
       // calculate the sum of dsigits recursively until it's single digit
30
31
       return superDigit(to_string(sum), 1);
32 }
```

```
34 int main()
35 {
36
       ofstream fout(getenv("OUTPUT_PATH"));
37
38
       string first_multiple_input_temp;
39
       getline(cin, first_multiple_input_temp);
40
       vector<string> first_multiple_input = split(rtrim(first_multiple_input_temp));
41
42
       string n = first_multiple_input[0];
43
44
45
       int k = stoi(first_multiple_input[1]);
46
47
       int result = superDigit(n, k);
48
       fout << result << "\n";</pre>
49
50
51
       fout.close();
52
       return 0;
53
54 }
55
56 string ltrim(const string &str) {
57
       string s(str);
58
59
       s.erase(
60
           s.begin(),
           find_if(s.begin(), s.end(), not1(ptr_fun<int, int>(isspace)))
61
62
       );
63
64
       return s;
65 }
66
67 string rtrim(const string &str) {
68
       string s(str);
69
70
71
           find_if(s.rbegin(), s.rend(), not1(ptr_fun<int, int>(isspace))).base(),
72
           s.end()
73
       );
74
75
       return s;
76 }
77
78 vector<string> split(const string &str) {
79
       vector<string> tokens;
80
81
       string::size_type start = 0;
82
       string::size_type end = 0;
83
       while ((end = str.find(" ", start)) != string::npos) {
84
85
           tokens.push_back(str.substr(start, end - start));
86
87
           start = end + 1;
88
       }
89
90
       tokens.push_back(str.substr(start));
91
92
       return tokens;
93 }
94
```

Interview Prep | Blog | Scoring | Environment | FAQ | About Us | Support | Careers | Terms Of Service | Privacy Policy |