Quicksort 1 - Partition

locked

Submitted 25 minutes ago • Score: 0.00 Status: Processed

Submitted Code

nguage: C++ Open in editor

```
#include <bits/stdc++.h>

using namespace std;

string ltrim(const string &);
string rtrim(const string &);
vector<string> split(const string &);

/*
 * Complete the 'quickSort' function below.
 *
 * The function is expected to return an INTEGER_ARRAY.
 * The function accepts INTEGER_ARRAY arr as parameter.
 */

vector<int> quickSort(vector<int> arr) {
    //  initializing vectors
    vector<int> left;
    vector<int> right;
    vector<int> equal;
    vector<int> partition;

    int p = arr[0];
    equal.push_back(p);
    // iterating through the vector and differentiate the greater and lower values than the p
    for (int i=1; i<(int)arr.size(); i++){
        if(arr[i]<p){
            left.push_back(arr[i]);
        } else if (arr[i]==p){
            equal.push_back(arr[i]);
        } else {
            right.push_back(arr[i]);
        }
    }
    // combining all the vectors
    partition.insert(partition.begin(),left.begin(),left.end());
    partition.insert(partition.end(),equal.begin(),equal.end());
    partition.insert(partition.end(),right.begin(),right.end());
```

```
39
40      return is_partitioned(_IIter, _IIter, _Predicate)();
41  }
42
43  int main()
44  {
45      ofstream fout(getenv("OUTPUT_PATH"));
46
47      string n_temp;
48      getline(cin, n_temp);
49
50      int n = stoi(ltrim(rtrim(n_temp)));
51
52      string arr_temp_temp;
53      getline(cin, arr_temp_temp);
54
55      vector<string> arr_temp = split(rtrim(arr_temp_temp));
56
57      vector<int> arr(n);
58
59      for (int i = 0; i < n; i++) {
60          int arr_item = stoi(arr_temp[i]);
61
62          arr[i] = arr_item;
63      }
64
65      vector<int> result = quickSort(arr);
66
67      for (size_t i = 0; i < result.size(); i++) {
68          fout << result[i];
69
70          if (i != result.size() - 1) {
71              fout << " ";
72          }
73      }
74
75      fout << "\n";
76
77      fout.close();
78
79      return 0;
80  }
81
82  string ltrim(const string &str) {
83      string s(str);
84
85      s.erase(
86          s.begin(),
87          find_if(s.begin(), s.end(), not1(ptr_fun<int, int>(isspace)))
88      );
89
90      return s;
91  }
92
93  string rtrim(const string &str) {
94      string s(str);
95
96      s.erase(
97          find_if(s.rbegin(), s.rend(), not1(ptr_fun<int, int>(isspace))).base(),
98          s.end()
99      );
100
101     return s;
102 }
103
104 vector<string> split(const string &str) {
```

```
105     vector<string> tokens;
106 
107     string::size_type start = 0;
108     string::size_type end = 0;
109 
110     while ((end = str.find(" ", start)) != string::npos) {
111         tokens.push_back(str.substr(start, end - start));
112 
113         start = end + 1;
114     }
115 
116     tokens.push_back(str.substr(start));
117 
118     return tokens;
119 }
120 
```