Arrays - DS

Problem | Submissions | Leaderboard | Discussions

Submitted a few seconds ago • Score: 30.00 Status: Accepted

✓	Test Case #0	✓	Test Case #1	✓	Test Case #2
✓	Test Case #3	✓	Test Case #4	✓	Test Case #5
✓	Test Case #6	✓	Test Case #7	✓	Test Case #8

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 'reverseArray' function below.
 *
 * The function is expected to return an INTEGER_ARRAY.
 * The function accepts INTEGER_ARRAY a as parameter.
 */

vector<int> reverseArray(vector<int> a) {
    int arr_length = a.size(); // getting the size of array a
    vector<int> reverse_arr(arr_length); // initializing the reverse_arr vector<int>

    // looping through array a
    for (int i=0; i<arr_length; i++){
        reverse_arr[i] = a[arr_length-i-1]; // assigning the last element for first element i reverse_arr
    }
    return reverse_arr;
}

int main()
{
    ofstream fout(getenv("OUTPUT_PATH"));

    string arr_count_temp;
    getline(cin, arr_count_temp);

    int arr_count = stoi(ltrim(rtrim(arr_count_temp)));

    string arr_temp_temp;
    getline(cin, arr_temp_temp);

    vector<string> arr_temp = split(rtrim(arr_temp_temp));

    vector<int> arr(arr_count);

    for (int i = 0; i < arr_count; i++) {
        int arr_item = stoi(arr_temp[i]);

        arr[i] = arr_item;
    }

    vector<int> res = reverseArray(arr);

    for (size_t i = 0; i < res.size(); i++) {
        fout << res[i];

        if (i != res.size() - 1) {
            fout << " ";
        }
    }

    fout << "\n";

    fout.close();

    return 0;
}

string ltrim(const string &str) {
    string s(str);

    s.erase(
        s.begin(),
        find_if(s.begin(), s.end(), not1(ptr_fun<int, int>(isspace)))
    );

    return s;
}

string rtrim(const string &str) {
    string s(str);

    s.erase(
        find_if(s.rbegin(), s.rend(), not1(ptr_fun<int, int>(isspace))).base(),
        s.end()
    );

    return s;
}

vector<string> split(const string &str) {
    vector<string> tokens;

    string::size_type start = 0;
    string::size_type end = 0;

    while ((end = str.find(" ", start)) != string::npos) {
        tokens.push_back(str.substr(start, end - start));

        start = end + 1;
    }

    tokens.push_back(str.substr(start));

    return tokens;
}

```