NEW
PREPARE CERTIFY COMPETE Search 210554M_CSE_21

All Contests > In21-CS2023-Lab3 > Insertion Sort - Part 1

Insertion Sort - Part 1

Problem | Submissions | Leaderboard | Discussions

Submitted a minute ago • Score: 30.00

Status: Accepted

✓ Test Case #0 ✓ Test Case #1 ✓ Test Case #2
✓ Test Case #3

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 'insertionSort1' function below.
 *
 * The function accepts following parameters:
 *  1. INTEGER n
 *  2. INTEGER_ARRAY arr
 */

void insertionSort1(int n, vector<int> arr) {
    int key = arr[n-1]; // initializing right most number as key
    int x = n-2;
    while(x >= 0 && arr[x]>key){ // check whether left elements are larger than key value
        arr[x+1]= arr[x]; // shifting elements

        // printing array after each shifting
        for (int j=0; j<n; j++){
            cout << arr[j] << " ";
        }
        cout << endl;

        x--;
    }
    arr[x+1] = key; // placing key value at its correct position

    // printing array after correct placing
    for (int j=0; j<n; j++){
        cout << arr[j] << " ";
```

```cpp
    }
    cout << endl;
}

int main()
{
    string n_temp;
    getline(cin, n_temp);

    int n = stoi(ltrim(rtrim(n_temp)));

    string arr_temp_temp;
    getline(cin, arr_temp_temp);

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

    vector<int> arr(n);

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

        arr[i] = arr_item;
    }

    insertionSort1(n, arr);

    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;
```

```
}

```

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