Compare the Triplets ☆

Alice and Bob each created one problem for HackerRank. A reviewer rates the two challenges, awarding points on a scale from 1 to 100 for three categories: problem clarity, originality, and difficulty.

We define the rating for Alice's challenge to be the triplet $A = (a[0], a[1], a[2])$, and the rating for Bob's challenge to be the triplet $B = (b[0], b[1], b[2])$.

Your task is to find their comparison points by comparing $a[0]$ with $b[0]$, $a[1]$ with $b[1]$, and $a[2]$ with $b[2]$.

- If $a[i] > b[i]$, then Alice is awarded 1 point.
- If $a[i] < b[i]$, then Bob is awarded 1 point.
- If $a[i] = b[i]$, then neither person receives a point.

Comparison points is the total points a person earned.

Given A and B, can you compare the two challenges and print their respective comparison points?

Input Format

The first line contains 3 space-separated integers, $a[0]$, $a[1]$, and $a[2]$, describing the respective values in triplet A.
The second line contains 3 space-separated integers, $b[0]$, $b[1]$, and $b[2]$, describing the respective values in triplet B.

Constraints

- $1 \leq a[i] \leq 100$
- $1 \leq b[i] \leq 100$

Output Format

Return an array of two integers denoting the respective comparison points earned by Alice and Bob.

Sample Input 0

```
5 6 7
3 6 10
```

Sample Output 0

```
1 1
```

Explanation 0

In this example:

- $A = (a[0], a[1], a[2]) = (5, 6, 7)$
- $B = (b[0], b[1], b[2]) = (3, 6, 10)$

Now, let's compare each individual score:

- $a[0] > b[0]$, so Alice receives 1 point.
- $a[1] = b[1]$, so nobody receives a point.

Author Shafaet
Difficulty Easy
Max Score 10
Submitted By 483167

NEED HELP?

View discussions
View editorial
View top submissions

RATE THIS CHALLENGE

☆ ☆ ☆ ☆ ☆

MORE DETAILS

Download problem statement
Download sample test cases
Suggest Edits

- $a[2] < b[2]$, so Bob receives 1 point.

Alice's comparison score is 1, and Bob's comparison score is 1. Thus, we print `1 1` (Alice's comparison score followed by Bob's comparison score) on a single line.

Current Buffer (saved locally, editable) C++

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

using namespace std;

vector<string> split_string(string);

// Complete the solve function below.
vector<int> solve(vector<int> a, vector<int> b) {


}

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

    string a_temp_temp;
    getline(cin, a_temp_temp);

    vector<string> a_temp = split_string(a_temp_temp);

    vector<int> a(3);

    for (int i = 0; i < 3; i++) {
        int a_item = stoi(a_temp[i]);

        a[i] = a_item;
    }

    string b_temp_temp;
    getline(cin, b_temp_temp);

    vector<string> b_temp = split_string(b_temp_temp);

    vector<int> b(3);

    for (int i = 0; i < 3; i++) {
        int b_item = stoi(b_temp[i]);

        b[i] = b_item;
    }

    vector<int> result = solve(a, b);

    for (int i = 0; i < result.size(); i++) {
        fout << result[i];

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

    fout << "\n";

    fout.close();

    return 0;
}

```

```cpp
vector<string> split_string(string input_string) {
    string::iterator new_end = unique(input_string.begin(), input_string.end(), [] (const char &x, const char &y) {
        return x == y and x == ' ';
    });

    input_string.erase(new_end, input_string.end());

    while (input_string[input_string.length() - 1] == ' ') {
        input_string.pop_back();
    }

    vector<string> splits;
    char delimiter = ' ';

    size_t i = 0;
    size_t pos = input_string.find(delimiter);

    while (pos != string::npos) {
        splits.push_back(input_string.substr(i, pos - i));

        i = pos + 1;
        pos = input_string.find(delimiter, i);
    }

    splits.push_back(input_string.substr(i, min(pos, input_string.length()) - i + 1));

    return splits;
}

```

Line: 1 Col: 1

Upload Code as File | Test against custom input | Run Code | Submit Code