Mini-Max Sum ☆

Given five positive integers, find the minimum and maximum values that can be calculated by summing exactly four of the five integers. Then print the respective minimum and maximum values as a single line of two space-separated long integers.

Input Format

A single line of five space-separated integers.

Constraints

- Each integer is in the inclusive range $[1 \ldots 10^9]$.

Output Format

Print two space-separated long integers denoting the respective minimum and maximum values that can be calculated by summing exactly four of the five integers. (The output can be greater than a 32 bit integer.)

Sample Input

```
1 2 3 4 5
```

Sample Output

```
10 14
```

Explanation

Our initial numbers are 1, 2, 3, 4, and 5. We can calculate the following sums using four of the five integers:

1. If we sum everything except 1, our sum is $2+3+4+5=14$.
2. If we sum everything except 2, our sum is $1+3+4+5=13$.
3. If we sum everything except 3, our sum is $1+2+4+5=12$.
4. If we sum everything except 4, our sum is $1+2+3+5=11$.
5. If we sum everything except 5, our sum is $1+2+3+4=10$.

Hints: Beware of integer overflow! Use 64-bit Integer.

Need help to get started? Try the Solve Me First problem

Author	bishop15
Difficulty	Easy
Max Score	10
Submitted By	266090

NEED HELP?

View discussions

View editorial

View top submissions

RATE THIS CHALLENGE ☆ ☆ ☆ ☆ ☆

MORE DETAILS

Download problem statement

Download sample test cases

Suggest Edits

Current Buffer (saved locally, editable) C++

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

using namespace std;

vector<string> split_string(string);

// Complete the miniMaxSum function below.
void miniMaxSum(vector<int> arr) {
```

```
}

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

    vector<string> arr_temp = split_string(arr_temp_temp);

    vector<int> arr(5);

    for (int i = 0; i < 5; i++) {
        int arr_item = stoi(arr_temp[i]);

        arr[i] = arr_item;
    }

    miniMaxSum(arr);

    return 0;
}

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: 11 Col: 2

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