Sherlock and Array

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

using namespace std;

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

/*
 * Complete the 'balancedSums' function below.
 *
 * The function is expected to return a STRING.
 * The function accepts INTEGER_ARRAY arr as parameter.
 */

string balancedSums(vector<int> arr) {
    long long totalSum = 0;
    for (int num : arr) totalSum += num;

    long long leftSum = 0;
    for (int num : arr) {
        if (leftSum == totalSum - leftSum - num) {
            return "YES";
        }
        leftSum += num;
    }
    return "NO";
}

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

    string T_temp;
    getline(cin, T_temp);

    int T = stoi(ltrim(rtrim(T_temp)));

    for (int T_itr = 0; T_itr < T; T_itr++) {
        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;
        }

        string result = balancedSums(arr);

        fout << result << "\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;
}
```