Sales by Match

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

using namespace std;

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

/*
 * Complete the 'sockMerchant' function below.
 *
 * The function is expected to return an INTEGER.
 * The function accepts following parameters:
 *  1. INTEGER n
 *  2. INTEGER_ARRAY ar
 */

int sockMerchant(int n, vector<int> ar) {
    unordered_map<int, int>mp;
    int count=0;
    for(int i=0;i<n;i++){
        mp[ar[i]]++;
    }
    for(auto& i: mp){
        count+= i.second/2;
    }
    return count;
}

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

    string n_temp;
    getline(cin, n_temp);

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

    string ar_temp_temp;
    getline(cin, ar_temp_temp);

    vector<string> ar_temp = split(rtrim(ar_temp_temp));
```

```
    vector<int> ar(n);

    for (int i = 0; i < n; i++) {
        int ar_item = stoi(ar_temp[i]);

        ar[i] = ar_item;
    }

    int result = sockMerchant(n, ar);

    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;
}
```