Alice and Bob went to a pet store. There are *N* animals in the store where the *ith* animal is of type *Ai*.

Alice decides to buy some of these *N* animals. Bob decides that he will buy **all** the animals **left** in the store after Alice has made the purchase.

Find out whether it is possible that Alice and Bob end up with **exactly same multiset** of animals.

Input Format

- The first line of input will contain a single integer *T*, denoting the number of test cases.
- Each test case consists of multiple lines of input.
    - The first line of each test case contains an integer *N* — the number of animals in the store.
    - The next line contains *N* space separated integers, denoting the type of each animal.

Output Format

For each test case, output on a new line, YES, if it is possible that Alice and Bob end up with **exactly same** multiset of animals and NO otherwise.

You may print each character in uppercase or lowercase. For example, the strings YES, yes, Yes, and yES are considered identical.

TalentBattle

Sample Input

4

3

4 4 4

4

2 3 3 2

4

1 2 2 3

6

5 5 1 5 1 5

Sample Output

NO

YES

NO

YES

C

```
#include <stdio.h>


int main(void) {
        int t;
        scanf("%d",&t);
        while(t--)
        {
           int n,i,p=0;
           scanf("%d",&n);
           int A[n];
           int ans[101];
           for(i=0;i<n;i++)
           {
              scanf("%d",&A[i]);
           }
           for(i=0;i<101;i++)
           {
              ans[i]=0;
           }
           if(n%2==0)
           {
              for(i=0;i<n;i++)
              {
                 ans[A[i]]++;
              }
```

TalentBattle

```
        for(i=0;i<101;i++)
        {
          if(ans[i]%2!=0)
          {
            p=1;
            break;
          }
        }
        if(p==0)
        {
          printf("YES\n");
        }
        else
        {
          printf("NO\n");
        }
      }
      else
      {
        printf("NO\n");
      }
    }
    return 0;
}
```

TalentBattle

C++

```
#include<bits/stdc++.h>
using namespace std;
using ll = long long ;
#define endl "\n"
int main(){
ios_base::sync_with_stdio(false);
cin.tie(NULL);
int T=1; cin>>T;
while(T--){
ll n; cin>>n;
ll a[n]; map<ll,ll>mp; set<ll>s;
bool ok = true;
for(int i = 0 ; i < n ; i++){
   cin>>a[i];
   mp[a[i]]++;
   s.insert(a[i]);
}
for(auto it = s.begin(); it!=s.end(); it++){
   ll temp = mp[*it];
   if(temp%2){ok=false;}
}
if(ok){cout<<"YES"<<endl;}
else{cout<<"NO"<<endl;}

}
return 0;
}
```

TalentBattle

Java

```
import java.util.*;
import java.util.ArrayList;
import java.io.*;

class TestClass {

    static class FastReader {
        BufferedReader br;
        StringTokenizer st;

        public FastReader() {
            br = new BufferedReader(new InputStreamReader(System.in));
        }

        String next() {
            while (st == null || !st.hasMoreElements()) {
                try {
                    st = new StringTokenizer(br.readLine());
                } catch (IOException e) {
                    e.printStackTrace();
                }
            }
            return st.nextToken();
        }

        int nextInt() {
            return Integer.parseInt(next());
        }
```

TalentBattle

```
        long nextLong() {
                return Long.parseLong(next());
        }

        double nextDouble() {
                return Double.parseDouble(next());
        }

        double nextFloat() {
                return Float.parseFloat(next());
        }

        String nextLine() {
                String str = "";
                try {
                        str = br.readLine();
                } catch (IOException e) {
                        e.printStackTrace();
                }
                return str;
        }
}

public static void main(String[] sadf) {
        FastReader fr = new FastReader();
        int t = fr.nextInt();
        while (t-- > 0) {
                solve(fr);
```

TalentBattle

```
            }
        }

        public static void solve(FastReader fr) {
            int n = fr.nextInt();
            HashMap<Integer, Integer> map = new HashMap<Integer, Integer>();
            for (int i = 0; i < n; i++) {
                int num = fr.nextInt();
                map.put(num, map.getOrDefault(num, 0) + 1);
            }
            for (Map.Entry<Integer, Integer> e : map.entrySet()) {
                if (e.getValue() % 2 != 0) {
                    System.out.println("NO");
                    return;
                }
            }
            System.out.println("YES");
        }
        private static int log(int N) {
            return 31 - Integer.numberOfLeadingZeros(N);
        }

}
```

TalentBattle

Python

```
from collections import Counter
t = int(input())
for _ in range(t):
    n = int(input())
```

```
nums = [int(x) for x in input().split()]
b = Counter(nums)
c = True
for i in b:
    if b[i]%2==1:
        c = False
if c:
    print("YES")
else:
    print("NO")
```


TalentBattle