# New Year Chaos

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

using namespace std;

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

/*
 * Complete the 'minimumBribes' function below.
 *
 * The function accepts INTEGER_ARRAY q as parameter.
 */

void minimumBribes(vector<int> q) {
    int bribes=0;
    for(int i=0;i<q.size();i++){
        if(q[i]-(i+1)>2){
            cout<<"Too chaotic"<<endl;
            return;
        }
    }
    for(int i=0;i<q.size();i++){
        for(int j=max(0,q[i]-2);j<i;j++){
            if(q[j]>q[i]){
                bribes++;
            }
        }
    }
    cout<<bribes<<endl;
}

int main()
{
    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 q_temp_temp;
        getline(cin, q_temp_temp);

        vector<string> q_temp = split(rtrim(q_temp_temp));

        vector<int> q(n);

        for (int i = 0; i < n; i++) {
            int q_item = stoi(q_temp[i]);

            q[i] = q_item;
        }

        minimumBribes(q);
    }

    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;
}
```