C. Pinkie Pie Eats Patty-cakes

2 seconds

256 megabytes

standard input

standard output

Pinkie Pie has bought a bag of patty-cakes with different fillings! But it appeared that not all patty-cakes differ from one another with filling. In other words, the bag contains some patty-cakes with the same filling.

Pinkie Pie eats the patty-cakes one-by-one. She likes having fun so she decided not to simply eat the patty-cakes but to try not to eat the patty-cakes with the same filling way too often. To achieve this she wants the minimum distance between the eaten with the same filling to be the largest possible. Herein Pinkie Pie called the distance between two patty-cakes the number of eaten patty-cakes strictly between them.

Pinkie Pie can eat the patty-cakes in any order. She is impatient about eating all the patty-cakes up so she asks you to help her to count the greatest minimum distance between the eaten patty-cakes with the same filling amongst all possible orders of eating!

Pinkie Pie is going to buy more bags of patty-cakes so she asks you to solve this problem for several bags!

**Input**

The first line contains a single integer https://assets.codeforces.com/mathjax/fonts/HTML-CSS/TeX/png/Math/Italic/400/0054.png?V=2.7.5T (https://assets.codeforces.com/mathjax/fonts/HTML-CSS/TeX/png/Main/Regular/400/0031.png?V=2.7.5https://assets.codeforces.com/mathjax/fonts/HTML-CSS/TeX/png/Main/Regular/400/2264.png?V=2.7.5https://assets.codeforces.com/mathjax/fonts/HTML-CSS/TeX/png/Math/Italic/400/0054.png?V=2.7.5https://assets.codeforces.com/mathjax/fonts/HTML-CSS/TeX/png/Main/Regular/400/2264.png?V=2.7.5https://assets.codeforces.com/mathjax/fonts/HTML-CSS/TeX/png/Main/Regular/400/0031.png?V=2.7.5https://assets.codeforces.com/mathjax/fonts/HTML-CSS/TeX/png/Main/Regular/400/0030.png?V=2.7.5https://assets.codeforces.com/mathjax/fonts/HTML-CSS/TeX/png/Main/Regular/400/0030.png?V=2.7.51≤T≤100): the number of bags for which you need to solve the problem.

The first line of each bag description contains a single integer https://assets.codeforces.com/mathjax/fonts/HTML-CSS/TeX/png/Math/Italic/400/006E.png?V=2.7.5n (https://assets.codeforces.com/mathjax/fonts/HTML-CSS/TeX/png/Main/Regular/400/0032.png?V=2.7.5https://assets.codeforces.com/mathjax/fonts/HTML-CSS/TeX/png/Main/Regular/400/2264.png?V=2.7.5https://assets.codeforces.com/mathjax/fonts/HTML-CSS/TeX/png/Math/Italic/400/006E.png?V=2.7.5https://assets.codeforces.com/mathjax/fonts/HTML-CSS/TeX/png/Main/Regular/400/2264.png?V=2.7.5https://assets.codeforces.com/mathjax/fonts/HTML-CSS/TeX/png/Main/Regular/400/0031.png?V=2.7.5https://assets.codeforces.com/mathjax/fonts/HTML-CSS/TeX/png/Main/Regular/400/0030.png?V=2.7.5https://assets.codeforces.com/mathjax/fonts/HTML-CSS/TeX/png/Main/Regular/283/0035.png?V=2.7.52≤n≤105): the number of patty-cakes in it. The second line of the bag description contains https://assets.codeforces.com/mathjax/fonts/HTML-CSS/TeX/png/Math/Italic/400/006E.png?V=2.7.5n integers https://assets.codeforces.com/mathjax/fonts/HTML-CSS/TeX/png/Math/Italic/400/0061.png?V=2.7.5https://assets.codeforces.com/mathjax/fonts/HTML-CSS/TeX/png/Main/Regular/283/0031.png?V=2.7.5https://assets.codeforces.com/mathjax/fonts/HTML-CSS/TeX/png/Main/Regular/400/002C.png?V=2.7.5https://assets.codeforces.com/mathjax/fonts/HTML-CSS/TeX/png/Math/Italic/400/0061.png?V=2.7.5https://assets.codeforces.com/mathjax/fonts/HTML-CSS/TeX/png/Main/Regular/283/0032.png?V=2.7.5https://assets.codeforces.com/mathjax/fonts/HTML-CSS/TeX/png/Main/Regular/400/002C.png?V=2.7.5https://assets.codeforces.com/mathjax/fonts/HTML-CSS/TeX/png/Main/Regular/400/2026.png?V=2.7.5https://assets.codeforces.com/mathjax/fonts/HTML-CSS/TeX/png/Main/Regular/400/002C.png?V=2.7.5https://assets.codeforces.com/mathjax/fonts/HTML-CSS/TeX/png/Math/Italic/400/0061.png?V=2.7.5https://assets.codeforces.com/mathjax/fonts/HTML-CSS/TeX/png/Math/Italic/283/006E.png?V=2.7.5a1,a2,…,an (https://assets.codeforces.com/mathjax/fonts/HTML-CSS/TeX/png/Main/Regular/400/0031.png?V=2.7.5https://assets.codeforces.com/mathjax/fonts/HTML-CSS/TeX/png/Main/Regular/400/2264.png?V=2.7.5https://assets.codeforces.com/mathjax/fonts/HTML-CSS/TeX/png/Math/Italic/400/0061.png?V=2.7.5https://assets.codeforces.com/mathjax/fonts/HTML-CSS/TeX/png/Math/Italic/283/0069.png?V=2.7.5https://assets.codeforces.com/mathjax/fonts/HTML-CSS/TeX/png/Main/Regular/400/2264.png?V=2.7.5https://assets.codeforces.com/mathjax/fonts/HTML-CSS/TeX/png/Math/Italic/400/006E.png?V=2.7.51≤ai≤n): the information of patty-cakes' fillings: same fillings are defined as same integers, different fillings are defined as different integers. It is guaranteed that each bag contains at least two patty-cakes with the same filling.

It is guaranteed that the sum of https://assets.codeforces.com/mathjax/fonts/HTML-CSS/TeX/png/Math/Italic/400/006E.png?V=2.7.5n over all bags does not exceed https://assets.codeforces.com/mathjax/fonts/HTML-CSS/TeX/png/Main/Regular/400/0031.png?V=2.7.5https://assets.codeforces.com/mathjax/fonts/HTML-CSS/TeX/png/Main/Regular/400/0030.png?V=2.7.5https://assets.codeforces.com/mathjax/fonts/HTML-CSS/TeX/png/Main/Regular/283/0035.png?V=2.7.5105.

**Output**

For each bag print in separate line one single integer: the largest minimum distance between the eaten patty-cakes with the same filling amongst all possible orders of eating for that bag.

**Example**

**input**

**Copy**

4

7

1 7 1 6 4 4 6

8

1 1 4 6 4 6 4 7

3

3 3 3

6

2 5 2 3 1 4

**output**

**Copy**

3

2

0

4

**Note**

For the first bag Pinkie Pie can eat the patty-cakes in the following order (by fillings): https://assets.codeforces.com/mathjax/fonts/HTML-CSS/TeX/png/Main/Regular/400/0031.png?V=2.7.51, https://assets.codeforces.com/mathjax/fonts/HTML-CSS/TeX/png/Main/Regular/400/0036.png?V=2.7.56, https://assets.codeforces.com/mathjax/fonts/HTML-CSS/TeX/png/Main/Regular/400/0034.png?V=2.7.54, https://assets.codeforces.com/mathjax/fonts/HTML-CSS/TeX/png/Main/Regular/400/0037.png?V=2.7.57, https://assets.codeforces.com/mathjax/fonts/HTML-CSS/TeX/png/Main/Regular/400/0031.png?V=2.7.51, https://assets.codeforces.com/mathjax/fonts/HTML-CSS/TeX/png/Main/Regular/400/0036.png?V=2.7.56, https://assets.codeforces.com/mathjax/fonts/HTML-CSS/TeX/png/Main/Regular/400/0034.png?V=2.7.54 (in this way, the minimum distance is equal to https://assets.codeforces.com/mathjax/fonts/HTML-CSS/TeX/png/Main/Regular/400/0033.png?V=2.7.53).

For the second bag Pinkie Pie can eat the patty-cakes in the following order (by fillings): https://assets.codeforces.com/mathjax/fonts/HTML-CSS/TeX/png/Main/Regular/400/0031.png?V=2.7.51, https://assets.codeforces.com/mathjax/fonts/HTML-CSS/TeX/png/Main/Regular/400/0034.png?V=2.7.54, https://assets.codeforces.com/mathjax/fonts/HTML-CSS/TeX/png/Main/Regular/400/0036.png?V=2.7.56, https://assets.codeforces.com/mathjax/fonts/HTML-CSS/TeX/png/Main/Regular/400/0037.png?V=2.7.57, https://assets.codeforces.com/mathjax/fonts/HTML-CSS/TeX/png/Main/Regular/400/0034.png?V=2.7.54, https://assets.codeforces.com/mathjax/fonts/HTML-CSS/TeX/png/Main/Regular/400/0031.png?V=2.7.51, https://assets.codeforces.com/mathjax/fonts/HTML-CSS/TeX/png/Main/Regular/400/0036.png?V=2.7.56, https://assets.codeforces.com/mathjax/fonts/HTML-CSS/TeX/png/Main/Regular/400/0034.png?V=2.7.54 (in this way, the minimum distance is equal to https://assets.codeforces.com/mathjax/fonts/HTML-CSS/TeX/png/Main/Regular/400/0032.png?V=2.7.52).

O(nlog^2(n))

Binary search +constructive algorithms

#include<bits/stdc++.h>

#define pb push\_back

#define pii pair<int,int>

#define int long long int

#define vec vector<int>

#define inf 1e18

using namespace std;

int n,a[100001];

int check(int x)

{

map<int,int> h;

set<pii> s;

for(int i=0;i<n;i++)

h[a[i]]++;

for(auto it: h)

s.insert({it.second,it.first});

for(auto it=s.begin();it!=s.end();it++)

h[it->second]--;

bool flag=true;

int b[n+1];

for(int i=0;i<n && flag;i++)

{

if(s.empty())

{

flag=false;

break;

}

auto it=s.end();

it--;

b[i]=it->second;

s.erase(it);

if(i>=x && h[b[i-x]]>0)

s.insert({h[b[i-x]],b[i-x]}),h[b[i-x]]--;

}

return flag;

}

int32\_t main()

{

ios\_base::sync\_with\_stdio(false);

cin.tie(NULL);

cout.tie(NULL);

int tt=1;

cin>>tt;

while(tt--)

{

cin>>n;

int i;

for(i=0;i<n;i++)

cin>>a[i];

int l=0,r=n,ans=0;

while(l<=r)

{

int mid=l+(r-l)/2;

if(check(mid))

ans=max(ans,mid),l=mid+1;

else

r=mid-1;

}

cout<<ans<<"\n";

}

}