2-1：

1. 众数唯一时：

#include<iostream>

#include<vector>

#include<algorithm>

using namespace std;

//众数唯一的分治法 ,mid众数，maxcnt重数

void function(int s[],int n,int &mid,int &maxcnt)

{

int le,ri;

int num=n/2;

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

if(s[le]==s[num])

break;

for(ri=le+1;ri<n;ri++)

if(s[ri]!=s[num])

break;

int cnt=ri-le;

if(cnt>maxcnt)

{

maxcnt=cnt;

mid=s[num];

}

if(le+1>maxcnt)

{

function(s,le+1,mid,maxcnt);

}

if(n-ri>maxcnt)

{

function(s+ri,n-ri,mid,maxcnt);

}

}

int main()

{

int n,x;

int num=0,maxcnt=0;

int a[1000];

cin>>n;

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

{

cin>>a[i];

}

sort(a,a+n-1);

function(a,n,num,maxcnt);

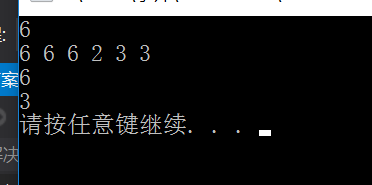
cout<<num<<endl;

cout<<maxcnt<<endl;

system("pause");

return 0;

}



1. 众数不唯一时：

STL解法：

#include<iostream>

#include<algorithm>

#include<map>

using namespace std;

bool cmp (const pair<int,int> p1,const pair<int,int> p2)

{

return p1.second<p2.second;

}

int main()

{

int n;

map<int,int> m;

cin>>n;

int x;

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

{

cin>>x;

m[x]++;//map中的int 不是类类型，初始值为0；

}

map<int,int>::iterator it=max\_element(m.begin(),m.end(),cmp);

map<int,int>::iterator i=m.begin();

cout<<"众数是：";

for(;i!=m.end();i++)

{

if((\*i).second==(\*it).second)

cout<<i->first<<" ";

}

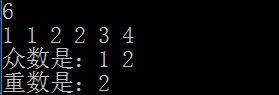
cout<<endl;

cout<<"重数是："<<it->second<<endl;

system("pause");

return 0;

}



2-3

#include<iostream>

using namespace std;

void function(int n,int &count)

{

for(int i=1;i<=n/2;i++)

function(i,++count);

}

int main()

{

int count=1;

int n;

cin>>n;

function(n,count);

cout<<count<<endl;

system("pause");

return 0;

}

