



信奥算法

数组最值

max_element()

```
1  #include<iostream>
2  #include<algorithm>
3  using namespace std;
4  int f[5]={3,1,5,1,2};
5  int main(){
6      cout<<max_element(f, f+5)<<endl;
7      cout<<max_element(f, f+5)-f<<endl;
8      cout<<*max_element(f, f+5)<<endl;
9      return 0;
10 }
```

算法库

max
最大

element
元素

请观察
输出结果

数组的内存地址

```
int f[5]={3,1,5,1,2};
```

数组编号	0	1	2	3	4	
数组元素值	3	1	5	1	2	







内存地址	0x472010	0x472014	0x472018	0x47201C	0x472020	0x472024
------	----------	----------	----------	----------	----------	----------



0x后面为16进制 每个地址放1字节	每个int变量 占4个字节
-----------------------	------------------

数组的内存地址

```
int f[5]={3,1,5,1,2};
```

数组编号	0	1	2	3	4	
数组元素值	3	1	5	1	2	
内存地址	0x472010	0x472014	0x472018	0x47201C	0x472020	0x472024
						







快快编程
kkcoding.net

运行"数组地址"程序

数组的内存地址

```
int f[5]={3,1,5,1,2};
```

数组编号	0	1	2	3	4	
数组元素值	3	1	5	1	2	

内存地址	0x472010	0x472014	0x472018	0x47201C	0x472020	0x472024
						
	f	f+1	f+2	f+3	f+4	f+5







f为数组
开头地址

f+5正好
在数组外

数组最值的地址

```
int f[5]={3,1,5,1,2};
```

数组编号	0	1	2	3	4	
数组元素值	3	1	5	1	2	

内存地址	0x472010	0x472014	0x472018	0x47201C	0x472020	0x472024
						
	f	f+1	f+2	f+3	f+4	f+5

`max_element(f, f+5)`
寻找从地址f开始, 在地址f+5之前
最大值的第一个地址
返回值为内存地址f+2

数组最值的编号

```
int f[5]={3,1,5,1,2};
```

数组编号	0	1	2	3	4	
数组元素值	3	1	5	1	2	

内存地址	0x472010	0x472014	0x472018	0x47201C	0x472020	0x472024
	↑	↑	↑	↑	↑	↑
	f	f+1	f+2	f+3	f+4	f+5

$\text{max_element}(f, f+5) - f$
是两个地址 $f+2$ 和 f 的距离
结果为整数 2, 代表数组编号

数组最值的数值

```
int f[5]={3,1,5,1,2};
```

数组编号	0	1	2	3	4	
数组元素值	3	1	5	1	2	

内存地址	0x472010	0x472014	0x472018	0x47201C	0x472020	0x472024
	↑	↑	↑	↑	↑	↑
	f	f+1	f+2	f+3	f+4	f+5

`*max_element(f, f+5)`
对应最大数值
星号*能取出地址内存放的数值

min_element()

```
1 #include<iostream>
2 #include<algorithm>
3 using namespace std;
4 int f[5]={3,1,5,1,2};
5 int main(){
6     cout<<min_element(f,f+5)<<endl;
7     cout<<min_element(f,f+5)-f<<endl;
8     cout<<*min_element(f,f+5)<<endl;
9     return 0;
10 }
```

min
最小

element
元素

请预测
输出结果

数组求最值

`min_element(f, f+n)`

翻译为：寻找从地址f开始, 在地址f+n之前
n个数里最小值的第一个地址

`min_element(f, f+n)`
`min_element(f, f+n)-f`
`*min_element(f, f+n)`

地址
编号
数值

时间复杂度 $O(n)$

实现方法还是逐个打擂台比大小

计数器数组

counters array

集五福

一共有5个福字：数字1代表爱国福，2代表富强福，3代表和谐福，4代表友善福，5代表敬业福。你不断收集了n个福字，请问其中有几套完整的五福临门？

输入第一行是正整数n，第二行为n个1到5之间的正整数。输出一个正整数。

输入样例：

12

1 2 3 4 5 1 2 3 4 5 1 2

输出样例：

2

输入样例：

4

2 3 4 5

输出样例：

0



集五福

一共有5个福字：数字1代表爱国福，2代表富强福，3代表和谐福，4代表友善福，5代表敬业福。你不断收集了n个福字，请问其中有几套完整的五福临门？

输入第一行是正整数n，第二行为n个1到5之间的正整数。输出一个正整数。

定义计数器数组

```
int cnt[6];
```

cnt[i]记录i出现的次数

明明是集五福
为什么是6？



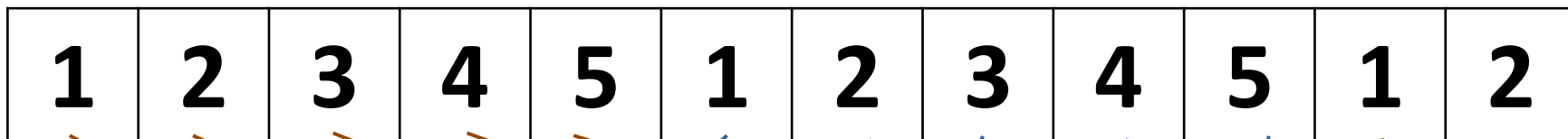
集五福

输入样例：

12

1 2 3 4 5 1 2 3 4 5 1 2

计数器数组大小
不是原数组大小



cnt数组
编号

0号

空着

1号

1的个数

2号

2的个数

3号

3的个数

4号

4的个数

5号

5的个数

```
1  #include<iostream>
2  #include<algorithm>
3  using namespace std;
4  const int R=6;
5  int cnt[R],n,x;
6  int main() {
7      cin>>n;
8      for(int i=0;i<n;i++) {
9          cin>>x;
10         cnt[x]++;
11     }
12     int ans=*min_element(cnt+1,cnt+R);
13     cout<<ans<<endl;
14     return 0;
15 }
```

注意最大编号为5

全局数组自动清零

x号福字数量加1


```
1  #include<iostream>
2  #include<algorithm>
3  using namespace std;
4  const int R=6;
5  int cnt[R],n,x;
6  int main() {
7      cin>>n;
8      for(int i=0;i<n;i++) {
9          cin>>x;
10         
11     }
12     int ans=
13     cout<<ans<<endl;
14     return 0;
15 }
```

集五福错误版

```
1  #include<iostream>
2  #include<algorithm>
3  using namespace std;
4  const int R=6;
5  int main(){
6      int cnt[R],n,x;
7      cin>>n;
8      for(int i=0;i<n;i++) {
9          cin>>x;
10         cnt[x]++;
11     }
12     int ans=*min_element(cnt+1,cnt+R);
13     cout<<ans<<endl;
14     return 0;
15 }
```

错在哪里

局部数组忘记清零

请运行程序观察

计数器最大易错点

```
1 #include<iostream>
2 using namespace std;
3 int main(){
4     int cnt[10];
5     for(int i=0;i<10;i++)
6         cout<<cnt[i]<<endl;
7     return 0;
8 }
```

能否预测输出结果

请运行程序观察

篮球统计

NBA全明星篮球比赛中双方每一次进球你都依次做了记录：可能是三分球，两分球，或者一分罚球。请统计：

三分球占总得分的百分比；

两分球占总得分的百分比；

一分罚球占总得分的百分比；

保留一位小数

输入样例：

2 2 3 1 3 3 1 1 2 2 1 1

输出样例：

40.9%

36.4%

22.7%



篮球统计

NBA全明星篮球比赛中双方每一次进球你都依次做了记录：可能是三分球，两分球，或者一分罚球。请统计：

三分球占总得分的百分比；

两分球占总得分的百分比；

一分罚球占总得分的百分比；

保留一位小数

输入样例：

2 2 3 1 3 3 1 1 2 2 1 1

明明是三分球
为什么是4？

定义计数器数组

```
int cnt[4];
```

cnt[i]记录i分球的个数



篮球统计

NBA全明星篮球比赛中双方每一次进球你都依次做了记录：可能是三分球，两分球，或者一分罚球。请统计：

三分球占总得分的百分比；

两分球占总得分的百分比；

一分罚球占总得分的百分比；

保留一位小数

输入样例：

2 2 3 1 3 3 1 1 2 2 1 1

不知道进球次数
如何输入？



篮球统计

```
1 #include<iostream>
2 #include<iomanip>
3 using namespace std;
4 const int R=4;
5 int cnt[R],x;
6 int main(){
7     while(cin>>x) cnt[x]++;
8     double sum=cnt[3]*3+cnt[2]*2+cnt[1];
9     cout<<fixed<<setprecision(1)<<cnt[3]*3/sum*100<<"%"<<endl;
10    cout<<fixed<<setprecision(1)<<cnt[2]*2/sum*100<<"%"<<endl;
11    cout<<fixed<<setprecision(1)<<cnt[1]/sum*100<<"%"<<endl;
12    return 0;
13 }
```

当成功输入x时
不断循环

输出三行类似
可以复制黏贴

每次复制黏贴
易错点:忘修改

现场挑战 快快编程410

快快编程
kkcoding.net

'a'到'z'的字母如何统计出现次数

'a'对应0

'b'对应1

'c'对应2

...

'z'对应25

```
char x;  
cin>>x;  
int i=x-'a';  
cnt[i]++;
```

cnt[i]代表什么含义?

代表('a'+i)这个字母出现的次数

cnt[]数组定义一共多少个元素?

至少26个元素

```
3  const int R=30;  
4  int cnt[R];
```

```
8      char ch;  
9      while(cin>>ch) {  
10         if(ch>='a'&&ch<='z'){  
11             int i=ch-'a';  
12               
13         }  
14     }  
15     int cl=cnt['l'-'a'];  
16     int co=cnt['o'-'a'];  
17     int cv=  
18     int ce=cnt['e'-'a'];  
19     int ans=  
20     cout<<ans<<endl;
```

计数器数组 + 前缀和数组

现场挑战 快快编程409

快快编程
kkcoding.net

得x分的
人的名次

=

分数大于
x的人数

+

1

=

总共
人数

-

分数小于等于
x的人数

+

1

0分的
人数

1分的
人数

2分的
人数

.....

x分的
人数


计数器数组+前缀和

```
1  #include<iostream>
2  using namespace std;
3  const int R=109;
4  const int N=10009;
5  int x[N],cnt[R],s[R],n;
6  int main(){
7      cin>>n;
8      for(int i=1;i<=n;i++){
9          cin>>x[i];
10         cnt[x[i]]++;
11     }
12     s[0]=cnt[0];
13     for(int j=1;j<=100;j++) s[j]=s[j-1]+cnt[j];
14     for(int i=1;i<=n;i++)
15         cout<<n-s[x[i]]+1<<" ";
16     return 0;
17 }
```

cnt[i]代表得分为i有几人

s[i]代表得分小于等于i有几人

这句不可以省略
因为可能得0分



计数排序

counting sort

老师播放计数排序演示

请学生总结算法步骤

算法步骤可视化网站：

<https://visualgo.net/en/sorting>

计数器数组大小
不是原数组大小

请将以下10个正整数从小到大排序，每个数的范围是{1,2,3}

算法核心：利用计数器数组，统计{1,2,3}出现的次数

2	3	1	1	2	3	3	2	2	2
---	---	---	---	---	---	---	---	---	---

2	5	3
---	---	---

cnt[1]	cnt[2]	cnt[3]
1的个数	2的个数	3的个数

定义计数器数组

```
int cnt[4];  
cnt[i]记录i的个数
```

根据计数器的结果
依次输出2个1,5个2,3个3

计数排序

```
1  #include<iostream>
2  using namespace std;
3  const int R=4; ←————
4  const int N=10; ←————
5  int cnt[R],x;
6  int main(){
7      for(int i=0;i<N;i++) {
8          cin>>x;
9          cnt[x]++; ←————
10     }
11     for(int i=1;i<=R-1;i++) ←————
12         for(int j=1;j<=cnt[i];j++) ←————
13             cout<<i<<" "; ←————
14     return 0;
15 }
```

轮流翻译
每一行

cout输出共几次

时间复杂度?

$O(N+R)$

计数排序算法：
时间复杂度 $O(N+R)$
 N 代表输入数据个数
 R 代表输入数值的大小范围

基于比较的排序算法：
如快速排序, `sort()`
时间复杂度最快 $O(N\log(N))$
 N 代表输入数据个数

当 R 比较小的时候，
计数排序算法更快

当 R 比较大的时候，
基于比较的排序算法更快

现场挑战

快快编程368

快快编程
kkcoding.net

设计计数器数组

`cnt[i]`代表什么含义

`cnt[]`数组定义一共多少个元素?

答案ans变量如何求得?

快快编程作业

368

410

409

拓展题

411, 1467, 412