

实验八注意事项

- 本实验，每个实验任务的子任务较多，要求所有子任务均在一个主函数中完成，不要创建多个项目！！！！即8-1编写一个程序，程序里面包含三个子任务的实现；8-2编写一个程序。
- 建议同学在每个子任务前给出简单的功能注释，完成一个子任务的编码、调试、运行后再继续后面子任务的编写，这样方便代码的调试和查错，也是模块化设计的思想。

编码示例

```
#include<iostream>
#include<iomanip>
void main( )
{
    const int N=10;
    int a[N];
    int i,j,n,sum=0,avg;
    cout<<"请输入数组实际长度: "<<endl;
    cin>>n;
    //输入数组元素、求所有元素之和、求平均值
    for(i=0;i<n;i++)
    {   cin>>a[i];
        sum+=a[i];
    }
    avg=sum/n;
    //输出数组元素
    for(i=0;i<n;i++)
        cout<<setw(6)<<a[i];
}
```

实验八 任务要求

1. 设有一个double型维数组a, 长度为N (注意N值应满足以下操作需求)。建立该数组, 并完成以下任务:

- (1) 键盘输入15个实型数据存入数组a中, 并按照每行输出5个的格式, 输出该数组;
- (2) 计算a中所有元素的平均值, 并将它存储在a中有效范围的末位 (即a中已有15个有效元素, 所以, 该平均值应该被存入a[15]中。提示: 数组长度不应小于16);
- (3) 调整a数组中前15个元素的位置, 即存放平均值的元素a[15]保持不动。调整规则是: 元素值低于平均值的放在后部、高于平均值的放在前部;
- (4) 输出调整以后的a数组。要求: 以上输出格式均为每行5个, 数据域宽为10。

```
C:\Windows\system32\cmd.exe

请输入15个实型数:
12.3 34.5 64.5 23.7 1.23 78.9 50.4 88.2 34.7 5.6 90.3 24.3 78.67 44.5 6.5

    12.3      34.5      64.5      23.7      1.23
    78.9      50.4      88.2      34.7      5.6
    90.3      24.3      78.67     44.5      6.5

调整后的数据为:

    44.5      78.67     64.5      90.3      88.2
    78.9      50.4      1.23      34.7      5.6
    23.7      24.3      34.5      12.3      6.5

    42.5533
请按任意键继续. . .
```

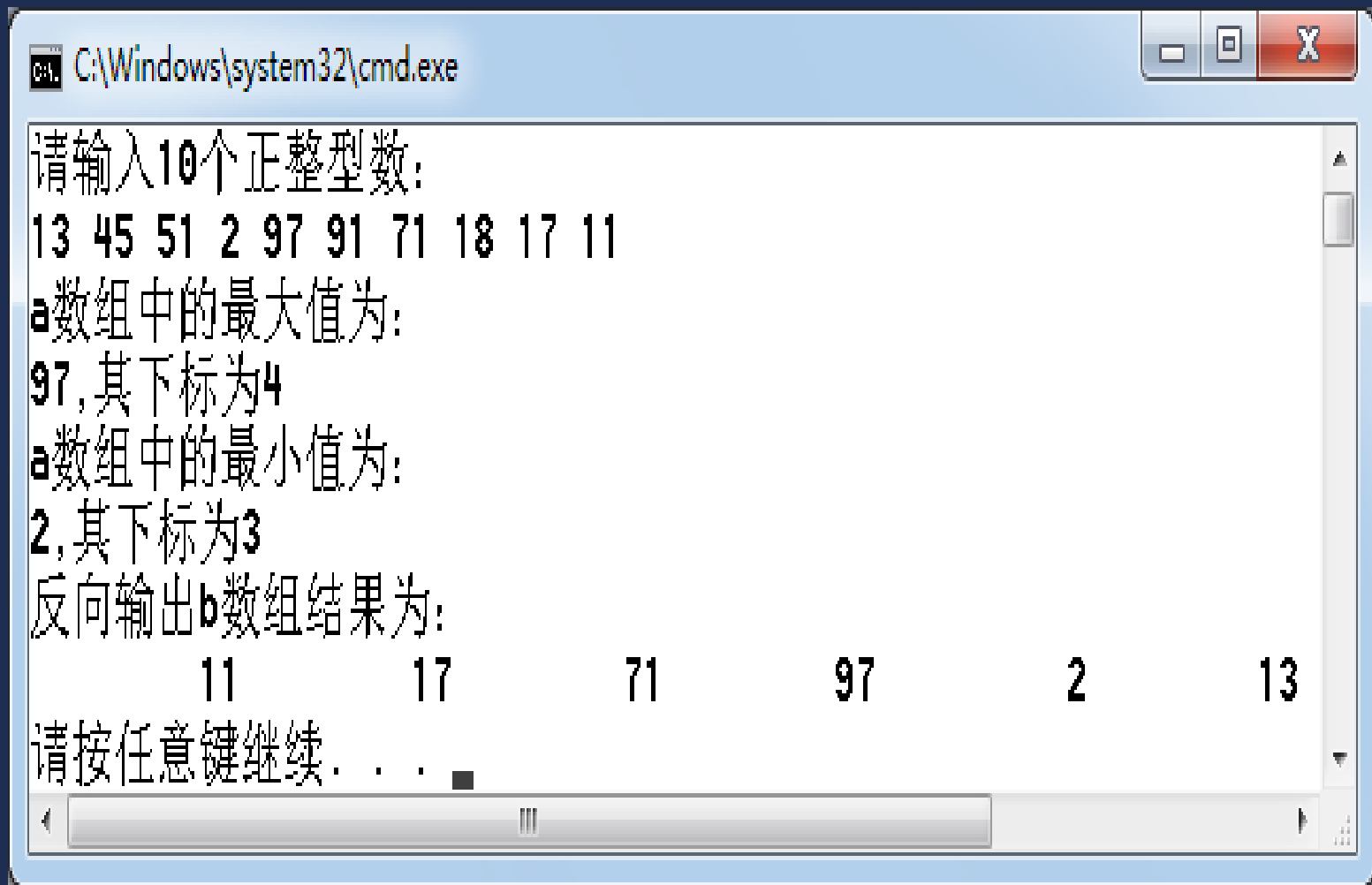
测试时，输入数据的技巧：

在数组的相关实验中，测试时通常需要输入大量的数据，为避免重复操作，可先新建一个txt文件，将需要输入的数据先写入文本文件中，每次测试只需复制这些数据，然后在程序的运行窗口中，在需要输入数据时，单击鼠标右键，然后选“粘贴”即可。

实验八 任务要求

2. 建立两个int型的一维数组，分别起名为a和b，并完成以下任务：

- (1) 编制一个判定某数是否为素数的子函数prime；
- (2) 键盘输入15个数据（这些书中有奇数、也有偶数）存入数组a中；
- (3) 输出a数组中的最大值和下标，以及输出其最小值和下标；
- (4) 通过调用子函数prime，找出数组a中所有的素数，并存入数组b中；
- (5) 反向（即从后b[n-1]到前b[0]）输出b中元素，数据域宽为10。



A screenshot of a Windows command prompt window titled "C:\Windows\system32\cmd.exe". The window contains the following text:

```
请输入10个正整型数:  
13 45 51 2 97 91 71 18 17 11  
a数组中的最大值为:  
97,其下标为4  
a数组中的最小值为:  
2,其下标为3  
反向输出b数组结果为:  
11      17      71      97      2      13  
请按任意键继续. . .
```

The output shows the maximum value 97 at index 4 and the minimum value 2 at index 3. The reverse output of array b is displayed as 11, 17, 71, 97, 2, 13.

数组的规范定义

```
#include <iostream>
void main()
```

```
{
    const int N=30;
```

```
    int a[N],n,i;
```

```
    cout<<"请输入数组的实际长度："<<endl;
```

```
    cin>>n;    //数组实际长度
```

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

```
        cin>>a[i];
```

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

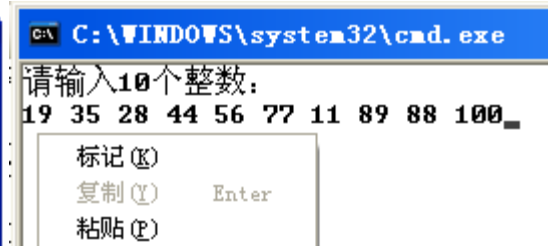
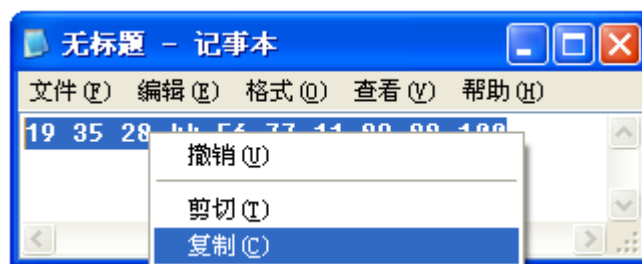
```
        cout<<a[i];
```

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

```
    {
        sum=sum+a[i];
    }
```

```
    .....
```

```
}
```



如何处理大量数据的输入：

1、在txt文档中按要求写入多个数据并保存，数据之间用空格间隔，选中所有数据，并“复制”。

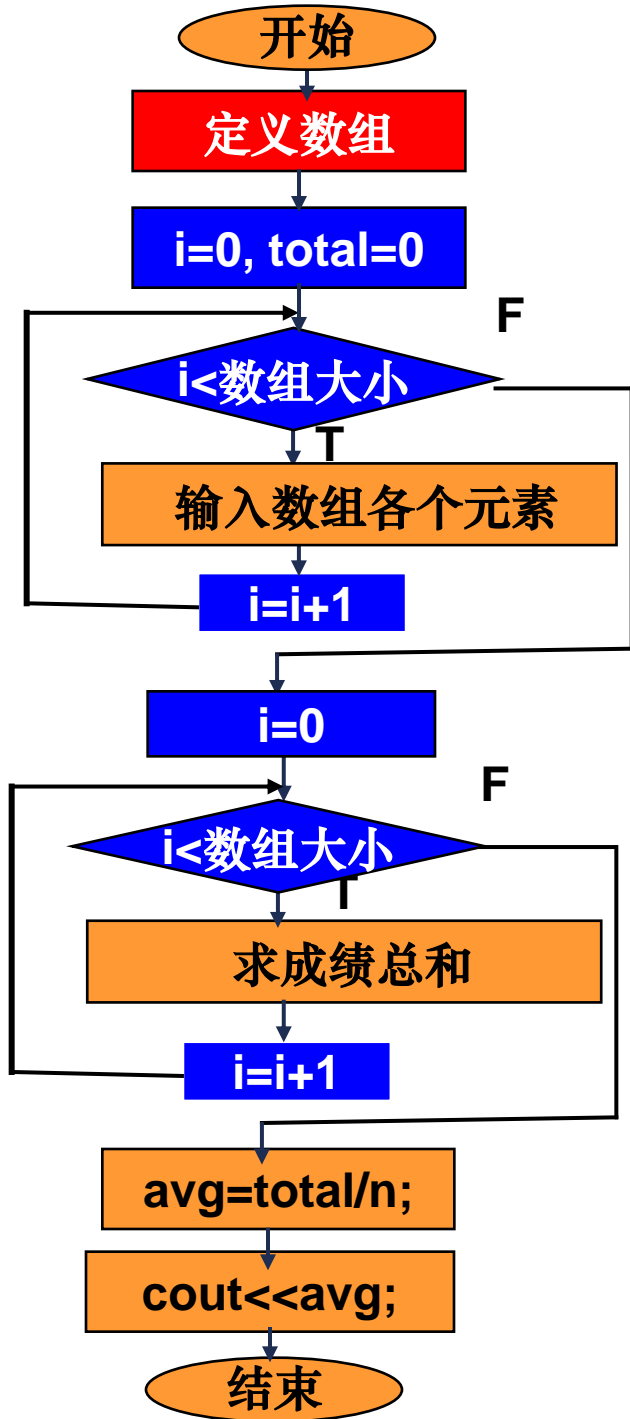
2、然后在VS中，编译、运行程序，在运行窗口的输入位置，单击鼠标右键，然后在弹出的快捷菜单中选择“粘贴”即可。

一维数组注意事项

- C++数组第一个元素的下标为0，而不是1，且下标表达方式是固定的。
- 数组是一种复合类型，是不能作为一个整体进行访问和处理的，只能按元素进行个别的访问和处理（需用循环处理）。

score[0]	88
score[1]	91
score[2]	81
score[3]	84
	84
	88
score[8]	...
score[n-1]

数组在内存中的存储



利用一维数组计算平均值

```
#include <iostream>
using namespace std;
void main( )
{
    const int N=10;
    int scores[N];           // 定义数组
    int n,i,total=0;         // n为数组实际长度
    float avg=0.0;
    cout<<"请输入数组长度: ";
    cin>>n;                  // 输入数组实际长度
    for(i=0;i<n;i++)
    { cin>>scores[i];        // 利用循环逐个输入数组元素
      total=total+scores[i]; // 对数组元素求和
    }
    avg=float(total)/n;      // 计算平均成绩
    cout<<"Average:"<<avg<<endl;
}
```

算法

定义数组，定义变量

从键盘接收数组元素的值

`high=dollars[0], j=0;`

`i=1`

`i<数组长度`

`dollars[i]>high`

T

F

`high=dollars[i]`

`j=i;`

`i=i+1`

最大值为第j+1个数，值为high

求一维数组中最大值及其位置

```
#include <iostream>
using namespace std;
void main( )
{
    const int N=10;
    int dollars[N];           //声明数组
    int n,i,j,high;
    cout<<"请输入数组实际长度：";
    cin>>n;
    for(i=0;i<n;i++) cin>>dollars[i];
    high=dollars[0]; //默认第一个元素为最大值
    j=0;             //记录最大元素的下标
    for(i=1;i<n;i++) //从第二个元素开始比较
    {
        if(dollars[i]>high)
        {
            high=dollars[i]; //记录最大值
            j=i;             //记录最大元素的下标
        }
    }
    cout<<"最大元素为第"<<j+1<<"个元素，
    它的值为："<< dollars[j] <<endl;
}
```

调整元素位置

将大于平均值的元素放在数组前面，小于平均值的放在后面。

a[0]	a[1]	a[2]	a[3]	a[4]	a[5]	a[6]	a[7]	a[8]	a[9]
13.5	20	11	67	3.3	41.2	3	5.2	29	21.47
29	41.2	67	11	3.3	20	3	5.2	13.5	21.47

a[0]	a[1]	a[2]	a[3]	a[4]	a[5]	a[6]	a[7]	a[8]	a[9]
13.5	20	11	67	3.3	41.2	3	5.2	29	21.47

↑
i

↑
j

29	20	11	67	3.3	41.2	3	5.2	13.5	21.47
----	----	----	----	-----	------	---	-----	------	-------

↑
i

↑
j

29	20	11	67	3.3	41.2	3	5.2	13.5	21.47
----	----	----	----	-----	------	---	-----	------	-------

↑
i

↑
j

29	20	11	67	3.3	41.2	3	5.2	13.5	21.47
----	----	----	----	-----	------	---	-----	------	-------

↑
i

↑
j

调整元素位置

将大于平均值的元素放在数组前面，小于平均值的放在后面。

a[0]	a[1]	a[2]	a[3]	a[4]	a[5]	a[6]	a[7]	a[8]	a[9]
13.5	20	11	67	3.3	41.2	3	5.2	29	21.47
29	41.2	67	11	3.3	20	3	5.2	13.5	21.47

a[0]	a[1]	a[2]	a[3]	a[4]	a[5]	a[6]	a[7]	a[8]	a[9]
29	41.2	11	67	3.3	20	3	5.2	13.5	21.47
	↑ i				↑ j				
29	41.2	11	67	3.3	20	3	5.2	13.5	21.47
		↑ i		↑ j					
29	41.2	11	67	3.3	20	3	5.2	13.5	21.47
		↑ i	↑ j						
29	41.2	67	11	3.3	20	3	5.2	13.5	21.47
		↑ i	↑ i	↑ j					

调整元素位置

将大于平均值的元素放在数组前面，小于平均值的放在后面。

a[0]	a[1]	a[2]	a[3]	a[4]	a[5]	a[6]	a[7]	a[8]	a[9]
13.5	20	11	67	3.3	41.2	3	5.2	29	21.47
29	41.2	67	11	3.3	20	3	5.2	13.5	21.47

处理流程：

设置两个下标变量 i (初值为0), j (初值为 $n-2$), 分别“指向”数组的第1个元素和最后一个元素 (不包括平均数), 比较两个元素与平均数 ($a[n-1]$) 的大小关系, 并进行下列处理, 直到 $i \geq j$ 时停止:

如果 $a[i] < a[n-1] \&\& a[j] > a[n-1]$, 则交换 $a[i]$ 和 $a[j]$, 同时 $i++$, $j--$, 即位置“指针”分别往数组后方移动和往前方移动。

否则, 判断: 如果 $a[i] < a[n-1]$, 则 $j--$ 。 (下标 j 前移, 在数组后部寻找大于平均值的元素)
否则, 判断: 如果 $a[j] > a[n-1]$, 则 $i++$ (下标 i 后移, 在前部寻找小于平均数的元素)

否则, $i++$ 、 $j--$ (此时, 数据均处于正确位置, 不需要交换, 直接移动前、后“指针”)。

调整元素位置 (二)

将大于平均值的元素放在数组前面，小于平均值的放在后面。

a[0]	a[1]	a[2]	a[3]	a[4]	a[5]	a[6]	a[7]	a[n-2]	a[n-1]
13.5	20	11	67	3.3	41.2	3	5.2	29	21.47

a[0]	a[1]	a[2]	a[3]	a[4]	a[5]	a[6]	a[7]	a[8]	a[9]
13.5	20	11	67	3.3	41.2	3	5.2	29	21.47

↑ $i=0$ $a[i] < a[n-1]$? 保留 $a[i]$ 的值 $t=a[i]$ ，将其后方所有元素依次前移1位

20	11	67	3.3	41.2	3	5.2	29	13.5	21.47
----	----	----	-----	------	---	-----	----	------	-------

将 t 的值 (保留的 $a[i]$ 的值) 写入 $a[n-2]$

继续考察 $a[i]$ 的值是否小于平均值，如果小于继续上述操作，否则处理下一个元素

20	11	67	3.3	41.2	3	5.2	29	13.5	21.47
----	----	----	-----	------	---	-----	----	------	-------

↑ $i=0$

11	67	3.3	41.2	3	5.2	29	13.5	20	21.47
----	----	-----	------	---	-----	----	------	----	-------

调整元素位置 (二)

将大于平均值的元素放在数组前面，小于平均值的放在后面。

a[0]	a[1]	a[2]	a[3]	a[4]	a[5]	a[6]	a[7]	a[n-2]	a[n-1]
13.5	20	11	67	3.3	41.2	3	5.2	29	21.47

a[0]	a[1]	a[2]	a[3]	a[4]	a[5]	a[6]	a[7]	a[8]	a[9]
11	67	3.3	41.2	3	5.2	29	13.5	20	21.47

↑ $i=0$ $a[i] < a[n-1]$? 保留 $a[i]$ 的值 $t=a[i]$ ，将其后方所有元素依次前移1位

67	3.3	41.2	3	5.2	29	13.5	20	11	21.47
----	-----	------	---	-----	----	------	----	----	-------

继续考察 $a[i]$ 的值是否小于平均值，如果小于继续上述操作，否则处理下一个元素

67	3.3	41.2	3	5.2	29	13.5	20	11	21.47
----	-----	------	---	-----	----	------	----	----	-------

↑ $i=1$

什么时候结束处理？ 处理到最后一个元素29结束，即 $a[i]=29$ 。

观察发现，在处理过程中，29的位置在变化，故在处理开始，需要保存它的值，即 $temp=a[n-2]$ ，`while(a[i]!=temp){ }`

const int N=30;

int a[N],i,j,k,n,m,t; *//n为数组实际长度*

输入数据a[0]至a[n-2] *//a[n-1]用于保存平均数*

temp=a[n-2] *//保存最后一个元素的值*

i=0;

while(a[i]!=temp)

a[i]<a[n-1]

//当前元素是否小于平均数

T

F

t=a[i]; *//保存原值*

for(j=i; j<n-2; j++)

a[j]=a[j+1]; *//元素前*

a[n-2]=t;

a[i]!=temp&&a[i]>a[n-1]

//未处理到最后一个元素且大于平均值

T

F

i++;

输出数组a的值

const int N=30;

int a[N],i,j,k,n,m,t; //n为数组实际长度

输入数据a[0]至a[n-2] //a[n-1]用于保存平均数

temp=a[n-2] //保存最后一个元素的值

i=0;

while(a[i]!=temp)

a[i]<a[n-1]

T

//当前元素是否小于平均数

F

t=a[i]; //保存原值

for(j=i; j<n-2; j++)

a[j]=a[j+1];//元素前

a[n-2]=t;

a[i]!=temp&&a[i]>a[n-1]

T

//未处理到最后一个元素且大于平均值

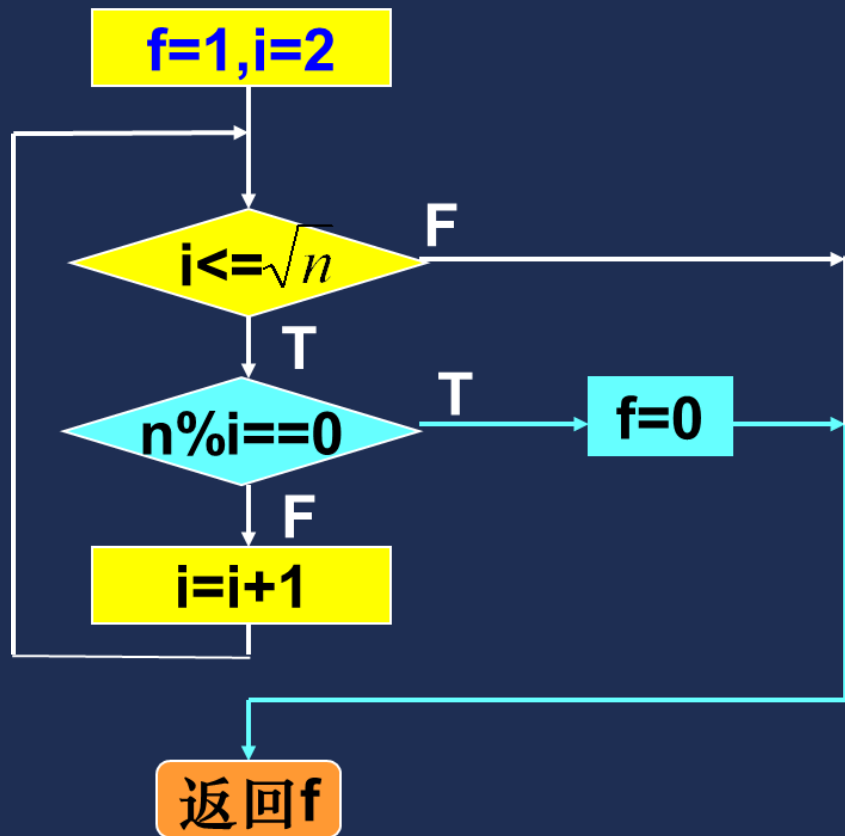
F

i++;

输出数组a的值

prime函数

bool prime(**int** n);



```
bool prime(int n)
{
    bool f=1;
    int i;
    for(i=2; i<=sqrt((double)n); i++)
        if(n%i==0)
        {
            f=0;
            break;
        }
    return f;
}
```

函数调用： prime(**a[i]**); //数组元素作为函数的实参