第四章课后编程练习-参考代码

- 请思考 P_{120} “问题拓展 1”：从键盘输入一组数据，存放在数组 arr 中，然后将数组 arr 中所有的偶数放在其前部，所有的奇数放在其后部，并以每行输出 5 个数据的格式输出该数组。
- 请自学 P_{137}~P_{141}，理解掌握插入和删除问题的算法思想，并熟练掌握课件(书)中相应代码。仔细阅读例 4.8、例 4.9 的代码。
- 掌握补充案例的算法及代码：只使用一个数组把一组数据中重复的数删除到只剩下一个。
- 自学 P159~P160：利用数组实现进制转换，理解并熟练掌握代码。
- 编程实现：从键盘输入整数集合 a、b 的元素个数和各个元素的值，计算并输出其交集(交集结果放在数组 c 中)。
- 编程实现：从键盘接收两个无序数列，用任一种排序法（选择法、冒泡法）对数列排序后，实现求它们的并集(并集结果放在数组 c 中) 。
- 思考主对角线和次对角线上的元素如果用下标表示？
- 如何用一个二维数组实现矩阵的转置？请编程实现。
- 自学 P154~P156 关于杨辉三角不同的输出形式，并请编程绘制如下所示的杨辉三角形。
- 自学 P156~PP158“自动按某种规则在 N 阶方阵中填数”，掌握例 4.18 的代码。
- 自学 P132：字符串比较问题，了解并熟悉代码即可。
- 自学 P135：组合国家和城市名称，了解并熟悉代码即可。
- 自学 P171：凯撒加密算法，了解并熟悉代码即可。
- 自学 P179：回文串的判定，了解并熟悉代码即可。

1、奇偶数调整

● 用两个数组实现（需掌握！）

```
#include<iostream>
#include<iomanip>
#include <ctime>
#include <cstdlib>
using namespace std;
void main()
{
    const int N=10;
    int   i,j, m,n;
    int a[N],b[N];

    cout<<"数组a:"<<endl;
    //随机产生a的元素
    srand(time(NULL));
```

```
    for(i=0;i<N;i++)
    {
        a[i]=1+rand()%20;      //产生1~20的随机数
        cout<<a[i]<<" ";      //输出时直接在每个元素后面加空格即可，也可设置宽度输出
    }

    m=0;
    n=N-1;    //设置前后下标，以记录数组b中偶数和奇数写入的位置
    for(i=0;i<N;i++)
    {
        if(a[i]%2==0)
            b[m++]=a[i];
        else
            b[n--]=a[i];
    }
    cout<<endl<<"调整后的结果为:"<<endl;
    for(i=0;i<N;i++)
        cout<<setw(4)<<b[i];
    cout<<endl;
}
```

- 用一个数组实现

```
//方法一：必须重点掌握!!!设置前后下标变量，同时进行奇偶数的交换
#include<iostream>
#include<iomanip>
using namespace std;
void main()
{
    const int N=15;
    int arr[N],i,j,n=10;
    cout<<"请输入 10 个整数："<<endl;
    for(i=0;i<n;i++)
        cin>>arr[i];
    i=0;
    j=n-1;
    while(i<j)
    {
        if(arr[i]%2!=0&&arr[j]%2==0)
        {
            arr[i]=arr[i]+arr[j];
            arr[j]=arr[i]-arr[j];
            arr[i]=arr[i]-arr[j];
        }
        else
```

```
            {
                if(arr[i]%2!=0)
                    j--;
                else
                {
                    if(arr[j]%2==0)
                        i++;
                    else
                    {
                        i++;
                        j--;
                    }
                }
            }
        }
        cout<<"调整后的数据为："<<endl;
        for(i=0;i<n;i++)
        {
            cout<<setw(6)<<arr[i];
            if((i+1)%5==0)
                cout<<endl;
        }
}
//调整方法二：通过移位实现奇偶数的调整
#include<iostream>
#include <ctime>
#include<cstdlib>
using namespace std;
void main()
{
      const int N=20;
      int a[N],i,j,n,t,m;
      cout<<"请输入数组个数"<<endl;
      cin>>n;
      cout<<"随机产生"<<n<<"个数组元素："<<endl;
     srand(time(NULL));
      for(i=0;i<n;i++)
      {
            a[i]=1+rand()%50;
            cout<<a[i]<<" ";
      }
      m=a[n-1];
      for(i=0;a[i]!=m;)
      {
```

```
        if(a[i]%2==1)
        {
            t=a[i];    //保留当前元素的值
            j=i;
            for(;j<n-1;j++)
                a[j]=a[j+1];        //其后续元素依次前移1位
            a[n-1]=t;       //将当前元素写入最后一个元素位置
            }
        else
            i++;   //如果当前位置的元素为偶数，则处理它的下一个元素
    }

    for(i=0;i<n;i++)
    {
        if(i%5==0)
        cout<<endl;
        cout<<a[i]<<" ";
    }
}
```

//调整方法三：利用冒泡法的思想进行调整，不仅实现调整还是实现排序!!!

```
#include<iostream>
using namespace std;
void main()
{
    const int N=100;
    int i,j,k=0,n,t;
    int a[N];
    cout<<"输入数组个数："<<endl;
    cin>>n;
    cout<<"输入数组"<<endl;
    for(i=0;i<n;i++)
        cin>>a[i];
    for(i=0;i<n;i++)
    {
        for(j=0;j<n-1-i;j++)
        {
            if(a[j]%2!=0)
            {
                t=a[j];
                a[j]=a[j+1];
                a[j+1]=t;
            }
        }
```

```
    }
    cout<<"调整后的数组："<<endl;
    for(i=0;i<n;i++)
    {
        cout<<a[i]<<"   ";
        k++;
        if(k%5==0)
            cout<<endl;
    }
    cout<<endl;
}
```

2、删除问题（两种删除方法必须熟练掌握!!!）

- 删除小于（大于）平均值的元素

```
    cout<<"删除小于平均值的元素："<<endl;
    for(i=0;i<n;i++)
    {
        if(a[i]<avg)
        {
            for(j=i;j<n;j++)
                a[j]=a[j+1];     //当前删除元素的所有后续元素依次前移 1 个位置
        n--;
        i--;    //下标回溯
        }
    }
```

- 删除重复元素

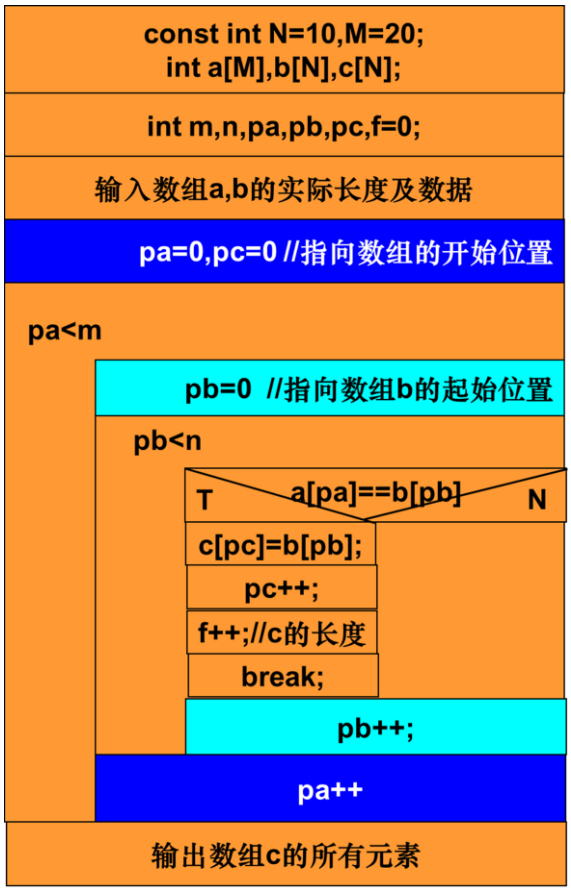
```
#include<iostream>
using namespace std;
void main(   )
{
int a[10],i,j,pos=1;   //pos 记录结果集的写入位置
for(i=0;i<10;i++)
    cin>>a[i];
for(i=1;i<10;i++)
 {
    for(j=0; j<pos;j++)
        if (a[i]==a[j])
            break;
 if(j>=pos)      //非重复元素
   {
     a[pos]=a[i];       //写入 pos 记录的位置
     pos++;           //结果集长度
```

```
    }
}
for(i=0; i<pos;i++)
      cout<<a[i]<<"   ";
cout<<endl;
 }
```

3、求交集、并集（掌握相关算法思想，并记忆关键代码！！！）

- 交集：从键盘输入整数集合 a、b 的元素个数和各个元素的值，计算并输出其交集(交集结果放在数组 c 中)

```
#include <iostream>
#include <iomanip>
using namespace std;
void main(void)
 {
    const int M=20,N=10;
    int a[M],b[N],c[N];
    int m,n,f=0,pa,pb,pc;
/* m、n 为数组 a、b 的实际长度；
f 记录数组 c 实际长度；
pa、pb、pc 为数组 a、b、c 首元素的下标 */
cout<<"输入数组 a 中元素的个数："<<endl;
cin>>m;
cout<<"输入数组 a 的元素："<<endl;
for(pa=0;pa<m;pa++)
       cin>>a[pa];      //用下标变量进行数组元素的输入
cout<<"输入数组 b 中元素的个数："<<endl;
cin>>n;
cout<<"输入数组 b 的元素："<<endl;
for(pb=0;pb<n;pb++)
        cin>>b[pb];      //用下标变量进行数组元素的输入
for( pa=0,pc=0;pa<m;pa++)
  for( pb=0;pb<n;pb++)
      if(a[pa]==b[pb])
          {
            c[pc]=a[pa];      //将 a、b 中相同元素写入 c,同时下标 pc 后移
            pc++;
            f++;    //用变量 f 记录交集数组 c 的实际长度
           break;    //找到 a、b 中相同元素即退出本轮比较
           }
cout<<"交集 c 的各个元素依次为:"<<endl;
for (pc=0;pc<f;pc++)
         cout<<setw(3)<<c[pc];
```



```
}
```

- 并集：求两个数组的并集（需合并的两个数组的原始数据均无序）

  思想（把数组 a 全部付给数组 c，再用数组 a 的每个元素分别和数组 b 元素比较，得出 b 中不同的元素全部赋给 c）

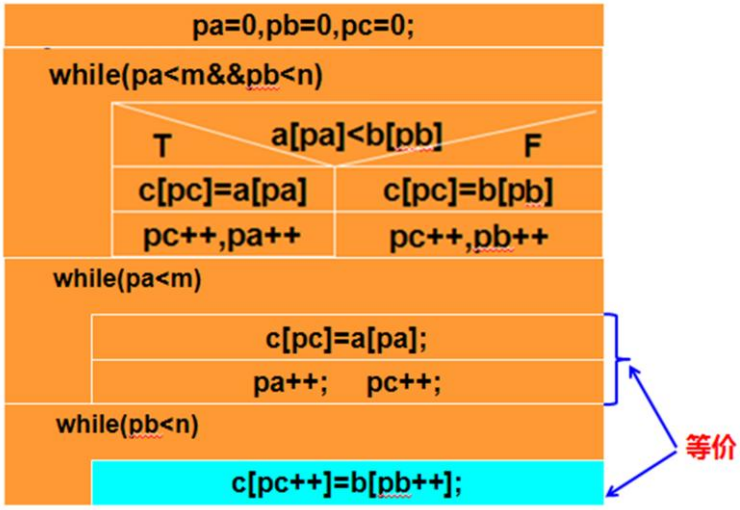
```
#include <iostream>
#include <iomanip>
using namespace std;
void main()
 {
    const int M=10,N=10;
    int a[M],b[N];
    int m,n,i=0,j=0,k,flag;

cout<<"输入数组 a 中元素的个数："<<endl;
cin>>m;
cout<<"输入数组 a 的"<<m<<"个元素(数组元素无序)："<<endl;
for(i=0;i<m;i++)
      cin>>a[i];
cout<<"输入数组 b 中元素的个数："<<endl;
   cin>>n;
cout<<"输入数组 b 的"<<n<<"个元素(数组元素无序)："<<endl;
for(j=0;j<n;j++)
    cin>>b[j];
//以数组 a 为基础，寻找 b 中不同于 a 的那些元素，然后写入 a 数组的末端。
k=m;    //指向 a 数组的末端
for(j=0;j<n;j++)
{
flag=0; //设置标志，如果 b 中有于 a 相同的元素，则 flag=1；
for(i=0;i<m;i++)
    if(a[i]==b[j])
    {
     flag=1;    //b 中若有与 a 中元素相同的，则结束本次处理，继续判断 b 的下一个元素
       break;
    }
if(flag==0)       //如果 b 的元素与 a 中所有元素均不相同，则写入 a 数组的末端
     a[k++]=b[j];
}
cout<<"并集的各个元素依次为:"<<endl;
for(i=0;i<k;i++)
cout<<setw(4)<<a[i];
}
```

- 设有有序（值从小到大）的两组数据，将这两组数据有序合并

```
#include<iostream>
using namespace std;
void main( )
{
        const int N=20;
      int a[N],b[N],c[2*N],m,n,i;
       int    pa=0,pb=0,pc=0,end;

      cout<<"请输入数组 a 的实际长度 m:"<<endl;
      cin>>m;
      cout<<"请输入数组 b 的实际长度 n:"<<endl;
      cin>>n;
      cout<<"请输入数组 a 的所有元素"<<endl;
      for(i=0;i<m;i++)
            cin>>a[i];
      cout<<"请输入数组 b 的所有元素"<<endl;
      for(i=0;i<n;i++)
            cin>>b[i];
while(pa<m && pb<n)
{
 if(a[pa]<b[pb])
          c[pc++]=a[pa++];
else
         c[pc++]=b[pb++];
}
while(pa<m)   //处理 a 中剩余元素
        c[pc++]=a[pa++];
while(pb<n)   //处理 b 中剩余元素
          c[pc++]=b[pb++];
for(i=0; i<pc;i++)
           cout<<c[i];
}
```



- 设有有序（值从小到大）的两组数据，将这两组数据有序求交集

```
#include <iostream>
#include <iomanip>
using namespace std;
void main(void)
 {
    const int M=20,N=10;
    int a[M],b[N],c[N];
    int m,n,f=0,pa,pb,pc,i;
```

```
/* m、n为数组a、b的实际长度；f记录数组c实际长度；pa、pb、pc为数组a、b、c首元素的下标*/
    cout<<"输入数组a中元素的个数："<<endl;
cin>>m;
cout<<"输入数组a的元素："<<endl;
for(pa=0;pa<m;pa++)
       cin>>a[pa];      //用下标变量进行数组元素的输入
cout<<"输入数组b中元素的个数："<<endl;
cin>>n;
cout<<"输入数组b的元素："<<endl;
for(pb=0;pb<n;pb++)
        cin>>b[pb];      //用下标变量进行数组元素的输入
 pa=0,pb=0,pc=0;
while(pa<m&&pb<n)
 {
      if(a[pa]==b[pb])
      {
          c[pc++]=a[pa++];
           pb++;
      }     //如果a、b中的元素相同，则将a的元素写入c，同时移动a,b,c的下标
     else
     {
          if(a[pa]<b[pb])
               pa++;
          else
                pb++;
      }    //因a、b中的元素有序，如果不等，则移动对应集合的下标
}

cout<<"交集c的各个元素依次为:"<<endl;    //pc记录交集的实际长度
for (i=0;i<pc;i++)
         cout<<setw(3)<<c[i];
 }
```

- 编程实现：随机产生两个无序数列，用任一种排序法（选择法、冒泡法）对数列排序后，实现求它们的并集(并集结果放在数组 c 中) 。

```
#include<iostream>
#include<iomanip>
#include <ctime>
#include <cstdlib>
using namespace std;
void main()
{
     void sort(int a[],int n);     //冒泡排序法
```

```
void del(int a[], int &n);     //删重

const int N=5;
int    i,j, m=0,n1,n2;
int a[N],b[N],c[2*N];

cout<<"数组a:"<<endl;
//随机产生a的元素
srand(time(NULL));
for(i=0;i<N;i++)
{
      a[i]=1+rand()%10;      //产生1~10的随机数
      cout<<a[i]<<" ";
}
//调用函数排序
 sort(a,N);
//去重处理
 n1=N;
del(a,n1);      //n1为去重后结果集的实际长度!



//随机产生b的元素
cout<<endl<<"数组b:"<<endl;
srand(time(NULL));
for(i=0;i<N;i++)
{
      b[i]=3+rand()%8;      //产生3~10的随机数
      cout<<b[i]<<" ";
}
//排序
sort(b,N);
//去重处理
 n2=N;
del(b,n2);       //n2为去重后结果集的实际长度!

//输出a,b的数据
cout<<endl<<"The new a is:"<<endl;
for(i=0;i<n1;i++)
      cout<<setw(4)<<a[i];
cout<<endl;
cout<<"The new b is:"<<endl;
for(i=0;i<n2;i++)
      cout<<setw(4)<<b[i];
```

```
    cout<<endl;

    //合并
    for(i=0;i<n1;i++)
        c[m++]=a[i];        //先将a中所有元素写入c中

    for(i=0;i<n2;i++)     //处理b中所有元素
        {
            j=0;      //将a中所有元素依次和b的当前元素做比较，如果b的当前元素均不
                        等于a的元素，则写入c
            while(b[i]!=a[j]&&j<n1)      //逐个比较元素
                j++;
            if(j==n1)
                c[m++]=b[i];
      //如果b的当前元素与a中所有元素均不等，则a中元素会处理到最后一个，循环结束
       时，看判断j==n1？以明确b的当前元素是否和a中所有元素均不等，否则当b[i]==a[j]
       退出循环，此时b[i]是重复元素，不应写入c中
    }
    cout<<"The intersection is:"<<endl;
    for(i=0;i<m;i++)
        cout<<setw(4)<<c[i];
    cout<<endl;
}

void sort(int a[],int n)
{
    int i,j;
    int temp;
    for(i=0;i<n-1;i++)          //N 个数要进行N-1轮的比较，i从0~N-2，刚好N-1次
    for(j=0;j<n-i-1;j++)          //每一轮中两两比较的次数为N-i-1次
            if(a[j]<a[j+1])
            {
                temp=a[j];
                a[j]=a[j+1];
                 a[j+1]=temp;
            }
}

void del(int a[], int &n)          //去重后，需要返回结果集的实际长度，而函数为void型，故
将n设置为引用变量(地址)，则形参n的改变会反馈到实参，实现数组实际长度值的修改！
{
        int i,j;
        int pos=1;
          for(i=1;i<n;i++)
```

```
        {
            for(j=0;   j<pos    ;j++)
                if (a[i]==a[j])
                    break;
            if(j>=pos)
            { a[pos]=a[i];   pos++; }
    }
    n=pos;
}
```

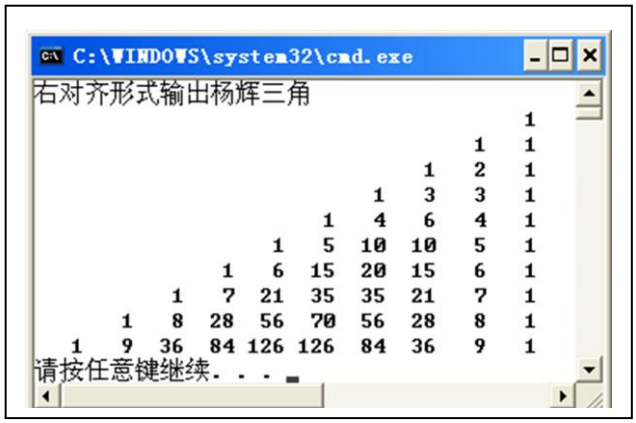
●

4、矩阵转置

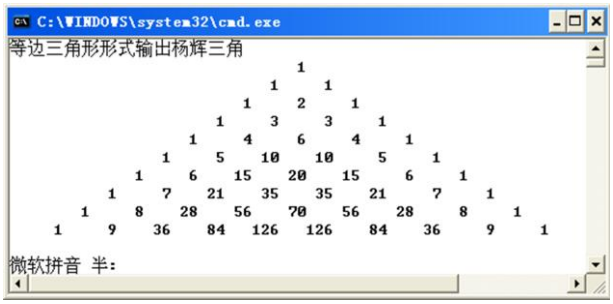
```
#include<iostream>
#include<ctime>
using namespace std;
void main()
{
const int N=4;
int a[N][N];
int i,j,t;
srand(time(NULL));
for(i=0;i<N;i++)
   for(j=0;j<N;j++)
      a[i][j]=10+rand()%(50-10+1);
cout<<"产生的二维数组为："<<endl;
for(i=0;i<N;i++)
{
   for(j=0;j<N;j++)
       cout<<a[i][j]<<"   ";      //输出一行元素
 cout<<endl;          //   一行输出完后输出一个回车换行
}
for(i=0;i<N;i++)
 for(j=0;j<i;j++)     //j=i，表示主对角线，把下三角的所有元素和上三角的元素对换
  {
     t=a[i][j];  a[i][j]=a[j][i];   a[j][i]=t;
  }
cout<<"转置之后的数组为："<<endl;
for(i=0;i<N;i++)
 {
    for(j=0;j<N;j++)
       cout<<a[i][j]<<"   ";
   cout<<endl;
}
}
```

```
for(i=0;i<N;i++)
    for(j=i+1;j<N;j++)
//把上三角的所有元素和下三角的元素对换
     {
          t=a[i][j];
          a[i][j]=a[j][i];
          a[j][i]=t;
     }
```

5、杨辉三角的不同输出形式




```
#include<iostream>
#include<iomanip>
using namespace std;
void main()
{
    const int N=10;
    int i,j,a[N][N];
    for(i=0;i<N;i++)
    {
        a[i][i]=1;          a[i][0]=1;
    }
    for(i=2;i<N;i++)   //杨辉三角放置成下三角的形式，然后找元素行列下标的关系
    {
        for(j=1;j<i;j++)
            a[i][j]=a[i-1][j]+a[i-1][j-1];
    } //杨辉三角形的数值处理
    cout<<"右对齐形式输出杨辉三角"<<endl;
    for(i=0;i<N;i++)
    {
        for(j=0;j<N-1-i;j++)          //次对角线
        cout<<setw(4)<<" ";
        for(j=0;j<=i;j ++)                  //数据部分，按产生规律（下三角）
            cout<<setw(4)<<a[i][j];
        cout<<endl;
    }//右对齐形式输出
}
```




```
#include<iostream>
#include<iomanip>
using namespace std;
void main()
{
    const int N=10;    int i,j,a[N][N];
    for(i=0;i<N;i++)
    {
        a[i][i]=1;          a[i][0]=1;
    }
    for(i=2;i<N;i++)      //杨辉三角放置成下三角的形式，然后找元素行列下标的关系
    {
        for(j=1;j<i;j++)
```

```
            a[i][j]=a[i-1][j]+a[i-1][j-1];
    } //杨辉三角形的数值处理
    cout<<"等边三角形形式输出杨辉三角"<<endl;
    for(i=0;i<N;i++)
    {
        for(j=0;j<N-1-i;j++)        //次对角线
            cout<<setw(3)<<" ";     //利用宽度控制输出格式
        for(j=0;j<=i;j++)           //数据部分，按产生规律（下三角）
            cout<<setw(6)<<a[i][j];          //利用宽度控制输出格式
        cout<<endl;
    }//等边三角形输出
    cout<<endl;
}
```

```
for(j=0;j<3*(N-1-i);j++)        //次对角线
  cout<<<" ";       //直接输出空格
```