课程实践题目

1．试建立一个类PP，求出下列多项式的前n项的值。

$$P_n(x)=\begin{cases}1 & n=0\\ x & n=1\\ ((2n-1)xP_{n-1}(x)-(n-1)P_{n-2}(x))/n & n>1\end{cases}$$

具体要求如下：

（1）私有数据成员

- int n：前若干项的项数。
- double　x：存放x的值。
- double *p：根据n的大小动态申请存放$P_n(x)$ 前n项的数组空间。

（2）公有成员函数

- PP(int num,double x1)：构造函数，初始化数据成员n和x，使p指向动态申请的数组空间。
- ～PP（）：析构函数，释放p指向的动态内存空间。
- double fun(int n1,double x)：递归函数，用于求多项式$P_n(x)$的第n1 项。注意：将递归公式中的n用作函数参数。本函数供process函数调用。
- void process( )：完成求前n项的工作，并将它们存放到p指向的动态数组中。
- void show( )：输出n和x，并将前n项以每行 4 个数的形式输出到屏幕上。

（3） 在主函数中完成对该类的测试。先输入num和x1，并定义一个PP类的对象items，用num和x1 初始化items的成员n和x，调用items的成员函数，求出并输出多项式前num项的值。

2. 试建立一个类SP，求$f(n,k)=1^k+2^k+3^k+....+n^k$，另有辅助函数power(m,n)用于求$m^n$。

具体要求如下：

（1）私有成员数据。

- int n，k：存放公式中n和k的值；

（2）公有成员函数。

- SP(int n1,int k1)：构造函数，初始化成员数据n和k。
- int power(int m, int n)：求m^n。
- int fun( )：求公式的累加和。
- void show( )：输出求得的结果。

（3）在主程序中定义对象s，对该类进行测试。

3．建立一个类MOVE，不进行排序，将数组中小于平均值的元素放到数组的左边，大于平均值的元素放到数组的右边。

具体要求如下：

（1）私有数据成员

- float array[20]：一维实型数组。
- int n：数组中元素的个数。

（2）公有成员函数

- MOVE(float b[],int m)：构造函数，初始化成员数据。
- void average()：输出平均值，并将数组中的元素按要求重新放置。
- void print()：输出一维数组。

（3）在主程序中用数据{1.3,6.2,3,9.1,4.8,7.4,5.6,9.2,2.3}对该类进行测试。

4．建立一个类MOVE，将数组中最大元素的值与最小元素的值互换。

具体要求如下：

（1）私有数据成员

- int *array：一维整型数组。
- int n：数组中元素的个数。

（2）公有成员函数

- MOVE(int b[],int m)：构造函数，初始化成员数据。
- void exchange()：输出平均值，并将数组中的元素按要求重新放置。
- void print()：输出一维数组。
- ~MOVE()：析构函数。

（3）在主程序中用数据{21,65,43,87,12,84,44,97,32,55}对该类进行测试。

5．定义一个类 Palindrome，实现绝对回文数。设计一个算法实现对任意整型数字判断是否为绝对回文数。所谓绝对回文数，是指十进制数和二进制数均对称的数。

具体要求如下：

（1）私有数据成员

- int n：整型数字。
- int y： 标记是否为回文数。

（2）公有成员函数

- Palindrome (int x) ：构造函数，根据x参数初始化数据成员n，y初始化为 0。
- void huiwen () ：判断数n是否为绝对回文数。
- void show( ) ：若该数为回文数，则在屏幕显示。

（3）在主程序中定义 int a，由键盘输入数字。定义一个 Palindrome 类对象 p，用 a 初始化 p，完成对该类的测试。

6．定义一个字符串类 String，实现判断该字符串是否为回文字符串。所谓回文字符串，是指该字符串左右对称。例如字符串“123321”是回文字符串。

具体要求如下：

（1）私有数据成员

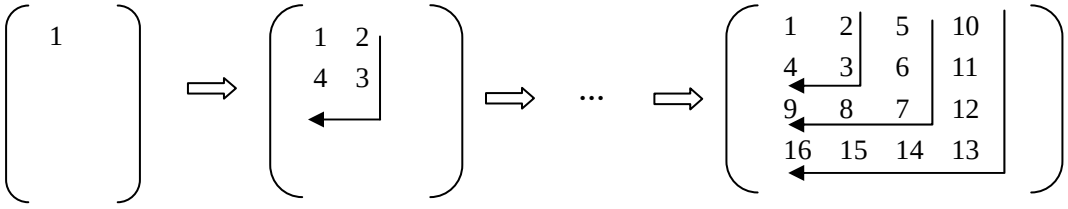
- char *str;
- int y：标记是否为回文字符串。

（2）公有成员函数

- String (char *s) ：构造函数，用给定的参数s初始化数据成员str。y初始化为 0。
- void huiwen () ：判断str所指向的字符串是否为回文字符串。
- void show( ) ：在屏幕上显示字符串。

（3）在主程序中定义字符串 char s[]=”ababcedbaba”作为原始字符串。定义一个 String 类对象 test，用 s 初始化 test，完成对该类的测试。

7．建立一个类PHALANX，生成并显示一个折叠方阵。折叠方阵如下图所示。折叠方阵的生成过程为：起始数置于方阵的左上角，然后从起始数开始递增，依次折叠构成方阵。



具体要求如下：

（1）私有数据成员

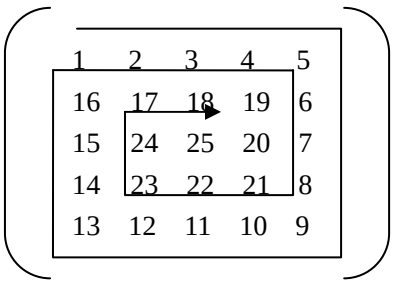
- int (*p)[20]：指向按照折叠规律存放方阵的二维整型数组。
- int startnum：折叠方阵的起始数。
- int n：存放方针的层数。

（2）公有成员函数

- PHALANX (int s, int m)：构造函数，初始化成员数据。
- void process()：生成起始数为startnum的n行方阵。
- void print()：输出折叠方阵。
- ~ PHALANX( )：析构函数。

（3）在主程序中对该类进行测试。

8．建立一个MATRIX类，生成并显示一个螺旋方阵。螺旋方阵如下图所示，起始数置于方阵的左上角，然后从起始数开始依次递增，按顺时针方向从外向里旋转填数而成。



具体要求如下：

（1）私有数据成员

- int a[20] [20]：二维整型数组存放螺旋方阵。
- int startnum：螺旋方阵的起始数。
- int n：存放方针的层数。

（2）公有成员函数

- MATRIX (int s, int m)：构造函数，初始化成员数据startnum和n。
- void process()：生成起始数为startnum的n行螺旋方阵。
- void print()：输出螺旋方阵。

（3）在主程序中定义MATRIX类的对象t对该类进行测试。

9．定义一个字符串类 CString，并设计一个算法对该串中各个不同字符出现的频率进行统计。具体要求如下：

（1）私有数据成员

- char *str：指向要统计的字符串。
- char (*p)[2]：动态分配二维空间，用以存放str所指字符串中出现的字符及其出现的次数（次数在存放时，用该数字对应的ASCII值存放；在输出次数时，输出

该ASCII字符对应的ASCII值即可）。

- int size：存放字符串中出现的所有不同的字符的个数。

（2）公有成员函数

- CString (char *s) ：根据s参数初始化数据成员str；p和size初始值为 0。
- void Count()：p根据s所指字符串长度分配空间。然后把str所指字符串中的每个字符放入p数组中，设置每个字符的出现次数为 1。根据p数组统计不同字符出现的频率，并求得size的实际大小。最后根据size的实际大小，重新分配p所指空间，并把不同字符及其出现次数重新放回p数组（提示：可以借助临时数组或指针来实现）。
- void Show()：屏幕显示字符串、字符串的每个字符和与之对应的次数。
- ~CString()：释放动态分配的空间。

（3）在主程序中定义字符串 char s[]=”abdabcdesffffd”。定义一个 CString 类对象 test，用 s 以初始化 test，完成对该类的测试。

10．定义一个字符串类 CString，并设计一个算法实现，给定关键字 str1 在字符串 str 中出现时用关键字 str2 进行替换的功能。

具体要求如下：

（1）私有数据成员

- char *str;　原始字符串。
- char *str1;　目标关键字。
- char *str2;　替换关键字。
- int flag;　　标记替换是否完成替换。

（2）公有成员函数

- CString (char *s,char s1[ ],char *s2) ：用给定的参数s、s1 和s2 相对应的初始化数据成员str、str1 和str2。flag设置缺省 0。
- void Replace () ：判断str字符串中是否出现str1，若出现就用str2 替换，否则什么都不做。若替换成功了标记flag为 1，若替换不成功则标记flag为 0。
- void Show( ) ：若替换成功，则在屏幕上显示目标关键字、替换关键字和替换后的原始字符串；若不成功则显示原始字符串。
- ~ CString() ：释放动态分配的空间。

（3）在主程序中定义字符串char s[]=”I am a student, you are student too, we are all student.” 作为原始字符串，定义char s1[]=” student”作为目标关键字，定义char s2[]=”teacher” 作为替换关键字。定义一个CString类对象test，用s，s1 和s2 初始化test，完成对该类的测试。

11．建立一个STRING类，将一个字符串交叉插入到另一个字符串中（假定两字符串等长）。例如将字符串“abcde” 交叉插入字符串“ABCDE”的结果为“aAbBcCdDeE”或“AaBbCcDdEe”。

具体要求如下：

（1）私有数据成员

- char str1[80] ：存放被插入的字符串。
- char str2[40] ：存放待插入的字符串。

（2）公有成员函数

- STRING (char *s1, char *s2)：构造函数，用s1 和s2 初始化str1 和str2。
- void process()：将str2 中的字符串插入到str1 中。
- void print()：输出插入后的字符串。

（3）在主程序中定义STRING类的对象test对该类进行测试。

12．建立一个STRING类，将一个字符串交叉插入到另一个字符串中（假定两字符串不等长）。例如将字符串“abcde”交叉插入字符串“ABCDEFG”的结果为“aAbBcCdDeEFG”或“AaBbCcDdEeFG”。

具体要求如下：

（1）私有数据成员

- char str1[60] ：存放被插入的字符串。
- char str2[40] ：存放待插入的字符串。
- char str3[100] ：存放插入后的字符串。

（2）公有成员函数

- STRING (char *s1, char *s2)：构造函数，用s1 和s2 初始化str1 和str2。
- void process()：将str2 中的字符串插入到str1 中，存放到str3 中。
- void print()：输出插入后的字符串。

（3）在主程序中定义STRING类的对象test对该类进行测试。

13．建立一个类MOVE，对数组中元素进行循环换位，即每个元素后移三位，最后三个元素移到最前面。

具体要求如下：

（1）私有数据成员

- int array[20]：一维整型数组。
- int n：数组中元素的个数。

（2）公有成员函数

- MOVE(int b[],int m)：构造函数，初始化成员数据。
- void change()：进行循环换位。
- void print()：输出一维数组。

（3）在主程序中用数据{21,65,43,87,12,84,44,97,32,55}对该类进行测试。

14．建立一个类MOVE，实现将数组中大写字母元素放在小写字母元素的左边。

具体要求如下：

（1）私有数据成员

- char *array：一维字符数组。
- int n：数组中元素的个数。

（2）公有成员函数

- MOVE(char b[],int m)：构造函数，初始化成员数据。
- void change()：进行排序换位。
- void print()：输出一维数组。
- ~MOVE()：析构函数。

（3）在主程序中用数据"fdsUFfsTjfsKFEkWC"对该类进行测试。

15．定义一个一维数组类Carray，并根据给定算法实现对原始一维数组进行线性变换。这里给定的线性变换算法为：T(bx) = bT(x) + i; 其中，b为变换常量，x为变量，i为当前类中成员数组的下标值。根据该算法，原始数组在变化后，当前数组元素的值是由常量b和i下标来决定的。

具体要求如下：

（1）私有数据成员

- int *a：指针a指向一个动态分配的原始数组。
- int n：n表示该数组的大小。

- int b：线性变换的常量。

（2）公有成员函数

- Carray (int a[ ],int n,int x) ：用给定的参数a、n和x初始化数据成员a、n和b。缺省都设置为 0。
- void Transform () ：根据上述变化算法，求解数组变换。
- void Show( ) ：在屏幕上显示数组元素。
- ~ Carray () ：释放动态分配的空间。

（3）在主程序中定义数组 int arr[ ]={1,2,3,4,5,6,7,8,9,10}作为原始数组，int b;由键盘输入，作为线性变换的常量。定义一个 Carray 类对象 test，用 arr 初始化 test，完成对该类的测试。

16．定义一个方阵类CMatrix，并根据给定算法实现方阵的线性变换。方阵的变换形式为：$F=W*f^T$

f为原始矩阵，f^T为原始矩阵的转置，w为变换矩阵，这里设定为

$$\begin{pmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \end{pmatrix}$$

具体要求如下：

（1）私有数据成员

- int (*a)[4]：a指向方阵数组。
- int w[4][4]：w为变换矩阵。
- int m：m表示方阵的行和列数。

（2）公有成员函数

- CMatrix (int a[][4],int m) ：用给定的参数a和m 初始化数据成员a和m；对变换矩阵w进行初始化，要求必须用循环实现。
- void Transform () ：根据上述变换算法，求出变换后的数组形式，存放在原始数组内。
- void show( ) ：在屏幕上显示数组元素。
- ~ CMatrix () ：释放动态分配的空间。

（3）在主程序中定义数组int arr[][4]={1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16}作为原始数组。定义一个CMatrix类对象test，用arr初始化test，完成对该类的测试。

17．定义一个类SIN，求 $\sin(x) = x/1 - x^3/3! + x^5/5! - x^7/7! + ... + (-1)^{n+1} x^{(2n-1)}/(2n-1)!$ 具体要求如下：

（1）私有成员数据。

- double x：输入公式中x的值，求sin(x)。
- int n：输入公式中n的值。

（2）公有成员函数。

- SIN(double x, int n)：构造函数，用于初始化x和n的值。
- double power(int q)：求q!的值。
- double mi(int m,int n)：求 m^n 的值。
- double fun( )：用于求SIN(X)的值。

- void show( )：输出求得的结果。

（3）在主程序中定义对象test，对该类进行测试。

18. 试建立一个类VAR，用于求n（$n \le 100$）个数的方差。方差的计算公式为 $d = \sum_{i=0}^{n-1} \frac{(x_i - \bar{x})^2}{n}$，

其中平均值为 $\bar{x} = \frac{\sum_{i=0}^{n-1} x_i}{n}$。

具体要求如下：

（1）私有成员数据。

- double a[100]：用于存放输入的n个数。
- int n：实际输入数的个数n。

（2）公有成员函数。

- VAR(double x[], int n1)：构造函数，初始化成员数据a和个数n。
- double average(double x[], int n)：求平均值，数组x具有n个元素。
- void variance(double x[],int n)：求均方差，数组x具有n个元素。
- void show()：输出求得的均方差。

（3）在主程序中定义一个对象test，对该类进行测试。

19．定义一个方阵类Array，实现对方阵进行顺时针 90 度旋转。如图所示。

$$\begin{pmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \\ 9 & 10 & 11 & 12 \\ 13 & 14 & 15 & 16 \end{pmatrix} \Longrightarrow \begin{pmatrix} 13 & 9 & 5 & 1 \\ 14 & 10 & 6 & 2 \\ 15 & 11 & 7 & 3 \\ 16 & 12 & 8 & 4 \end{pmatrix}$$

具体要求如下：

（1）私有数据成员

- int a[4][4]：用于存放方阵。

（2）公有成员函数

- Array (int a1[][4],int n) ：构造函数，用给定的参数a1 初始化数据成员a。
- void xuanzhuan ( ) ：实现对方阵a进行顺时针 90 度的旋转。
- void show( ) ：在屏幕上显示数组元素。

（3）在主程序中定义数组 int b[ ][4]={1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16}作为原始数组。定义一个 Array 类对象 test，用 b 初始化 test，完成对该类的测试。

20．定义一个方阵类Array，实现对方阵进行逆时针 90 度旋转。如图所示。

$$\begin{pmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \\ 9 & 10 & 11 & 12 \\ 13 & 14 & 15 & 16 \end{pmatrix} \Longrightarrow \begin{pmatrix} 4 & 8 & 12 & 16 \\ 3 & 7 & 11 & 15 \\ 2 & 6 & 10 & 14 \\ 1 & 5 & 9 & 13 \end{pmatrix}$$

具体要求如下：

（1）私有数据成员

- int a[4][4]：用于存放方阵。

（2）公有成员函数

- Array (int a1[][4],int n) ：构造函数，用给定的参数a1 初始化数据成员a。
- void xuanzhuan ( ) ：实现对方阵a进行逆时针 90 度的旋转。
- void show( ) ：在屏幕上显示数组元素。

（3） 在主程序中定义数组int b[ ][4]={1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16}作为原始数组。定义一个Array类对象test，用b初始化test，完成对该类的测试。

21．建立一个类NUM，求指定数据范围内的所有合数（非质数）。提示：合数定义是“一个数，除了 1 和它本身，还有其它约数，这样的数叫合数”。

具体要求如下：

（1）私有数据成员

- int *data：动态存放在指定范围内求出的所有合数。
- int span1,span2：存放指定范围的下限和上限。
- int num：存放span1 与span2 之间的合数个数。

（2）公有成员函数

- NUM(int n1, int n2) ：构造函数，用参数n1 和n2 初始化span1 和span2，同时初始化num。
- int isComposite (int x)：判断x是否为合数。若是合数，返回 1，否则，返回 0。
- void process( ) ：求指定范围内的所有合数，把它们依次存放在数组data中，并将求出的合数个数赋给num。
- void print( )：输出求出的素数个数及所有合数，每行输出 8 个合数。
- ~NUM( )：释放动态分配的存储空间。

（3）在主函数中完成对该类的测试。定义一个NUM类对象test，指定查找范围为 100~200，即求 100 至 200 之间的所有合数。通过test调用成员函数完成求合数及输出合数的工作。

22．建立一个类Saddle_point，求一个数组中的所有鞍点。提示：鞍点是这样的数组元素，其值在它所在行中为最大，在它所在列中为最小。

具体要求如下：

（1）私有数据成员

- int a[4][4]：存放二维数组元素。
- int b[4][4]：存放二维数组中的鞍点值。
- int num：存放鞍点个数。

（2）公有成员函数

- Saddle_point(int data[][4])：构造函数，用参数int data[][4]初始化数组a,，同时初始化数组b与num 的值均为 0。
- void process( )：求数组a所有鞍点（如果有鞍点），把它们行、列、及值相应存放在数组b中，并将求出的鞍点个数赋给num。
- void print( )：输出数组a、鞍点个数，与鞍点坐标及相应值。

（3）在主程序中定义数组int b[ ][4]={2, 6, 3, 4, 5, 6, 5, 5, 5, 7, 6, 7, 1, 9, 2, 7}作为原始数组。定义一个Saddle_point类对象fun。通过fun调用成员函数完成求鞍点及输出工作。

23．分数相加，两个分数分别是 1/5 和 7/20，它们相加后得 11/20。方法是先求出两个分数分母的最小公倍数，通分后，再求两个分子的和，最后约简结果分数的分子和分母（如果两个分数相加的结果是 4/8，则必须将其约简成最简分数的形式 1/2），即用分子分母的最大

公约数分别除分子和分母。求m、n最大公约数的一种方法为：将m、n较小的一个数赋给变量k，然后分别用{ k，k-1，k-2，…，1}中的数（递减）去除m和n，第一个能把m和n同时除尽的数就是m和n的最大公约数。假定m、n的最大公约数是v，则它们的最小公倍数就是m*n/v。试建立一个分数类Fract，完成两个分数相加的功能。

具体要求如下：

（1）私有数据成员

- int num，den ：num为分子，den为分母。

（2）公有成员函数

- Fract（int a=0,int b=1）：构造函数，用a和b分别初始化分子num、分母den。
- int ged（int m，int n）：求m、n的最大公约数。此函数供成员add()函数调用。
- Fract add（Fract f）：将参数分数f与对象自身相加，返回约简后的分数对象。
- void show( )：按照num/den的形式在屏幕上显示分数。

（3）在主程序中定义两个分数对象f1 和f2，其初值分别是 1/5 和 7/20，通过f1 调用成员函数add完成f1 和f2 的相加，将得到的分数赋给对象f3，显示分数对象f3。

24．建立一个类NUM，并统计特定序列中相同的数字的个数。

具体要求如下：

（1）私有数据成员

- int data[25]：随机生成 25 个在 0-9 之间的数字。
- int num[10]：储存每个数字出现的个数。

（2）公有数据成员

- NUM(int data)：构造函数，初始化数组data。
- void process( )：统计数组data中每个数字出现的个数，并保存到数组num中。
- void print( )：输出每个数字出现的个数，每行输出 5 个

（3）在主程序中定义一个对象，对该类进行测试。

25．建立一个类NUM，并统计特定序列中相同的字符的个数。

具体要求如下：

（1）私有数据成员

- char data[25]：随机生成 25 个字符。
- int num[128]：储存每个字符出现的个数。

（2）公有数据成员

- NUM(int data)：构造函数，同时初始化数组data。
- void process( )：统计数组data中每个字符出现的个数，并保存到数组num中。
- void print( )：输出每个出现过的字符及其出现的个数，每行输出 5 个，没有出现过的字符不显示。

（3）在主程序中定义一个对象，对该类进行测试。

26．建立一个类NUM，随机生成 25 个字符序列，并为特定序列进行排序。

具体要求如下：

（1）私有数据成员

- int data[25]：随机生成 25 个字符。

（2）公有数据成员

- NUM(int data)：构造函数，初始化数组data。
- void process( )：为数组data进行排序，要求按照ASCII码进行升序排列。
- void print( )：输出数组data，每行输出 5 个字符。

（3）在主程序中定义一个对象，对该类进行测试。

27．建立一个类NUM，求指定数据范围内的所有素数（质数）。提示：素数定义是“只能被 1 和它本身整除的整数”，即质数。

具体要求如下：

（1）私有数据成员

- int data[10]：依次存放原始数据。
- int prime[10]：存放指定数据内所求出的所有素数。
- int num：存放素数个数。

（2）公有数据成员

- NUM(int n[ ])：构造函数，用参数n初始化data，同时初始化num为 0。
- int isprime (int x)：判断x是否为素数。若是素数，返回 1，否则，返回 0。
- void process( )：求指定data数组中的所有素数，把它们依次存放在数组prime中，并将求出的素数个数赋给num。
- void print( )：输出求出的素数个数及所有素数，每行输出 4 个素数。

（3）在主函数中完成对该类的测试。定义NUM类对象test，通过test调用成员函数完成求素数及输出素数的工作。原始数据为{4, 5, 9, 11, 36, 29, 31, 101, 56, 199}。

28．编程实现对大于 1 的整数进行质因数分解，并求出其和。所谓整数的质因子分解是指将整数分解为其所有质数（素数）因数的积，例如，60=2*2*3*5，则整数 60 的质因数之和为 12。定义一个类Decompose实现上述功能。

具体要求如下：

（1）私有数据成员

- int *a：指向待分解质因数整数的动态存储空间。
- int *num：指向存放对应整数的质因数之和的动态存储空间。
- int n：整数的个数。

（2）公有数据成员

- Decompose(int m，int b[ ])： 用m初始化n，并用n初始化为动态申请空间的指针a与num。用参数b给数组a赋值。
- void print( )：输出数组a以及num所指向的存储空间中的内容。
- void primenum( )：求整数a[i]的所有质因数（保留重复部分，例如 60 的质因数为 2,2,3,5，之和为 12），并将这些质因数之和存放到指针num所指向的存储空间中。
- ~Decompose( )：释放动态分配的存储空间。

（3）在主函数中完成对该类的测试。从键盘输入一组大于 1 的整数，存放在number数组中，定义类Decompose的对象d ，并用 number初始化d ，调用函数primenum( ) 求 number 的所有质因数，最后输出测试结果。

29．建立一个类SUM，输入 5×5 的二维数组，编写程序实现：求出两对角线上各元素的和，求出对角线上行、列下标均为偶数的各元素的积，找出对角线上其值最大的元素以及它在数组中的位置。

具体要求如下：

（1）私有数据成员

- int array[5][5]：二维整型数组。
- int s：数组array两对角线元素的和。
- int a：数组array对角线上行、列下标均为偶数的各元素的积
- int b,m,n：数组array对角线上其值最大的元素以及它在数组中的位置。

（2）公有成员函数

- SUM(int d[5][5])：构造函数，初始化成员数据。
- void process1()：求二维数组两对角线元素的和。
- void process2()：求二维数组两对角线上行、列下标均为偶数的各元素的积。
- void process3()：求二维数组两对角线上其值最大的元素和它在数组中的位置。
- void print()：输出二维数组（每行输出 5 个元素）及其它所求的值。

（3）在主程序中对该类进行测试。

30．建立一个矩阵类Array，对二维数组中左下三角的全部元素（包括对角线上的元素）作如下变换：（1）若该数不是素数则保持不变；（2）若该数是素数，则用大于它的最小素数替换该数。并统计二维数组中左下三角的全部元素（包括对角线上的元素）中的素数个数。具体要求如下：

（1）私有数据成员

- int x[4][4]：存储需要处理的二维数组的各元素值。
- int count：存储左下三角元素中素数的个数。

（2）公有成员函数

- 构造函数：进行初始化x数组和count的值。
- int fun(int)；判断一个数是否为素数的函数。
- int encode( )：对x数组中左下三角的全部元素（包括对角线上的元素）逐一进行判断，若该数不是素数则保持不变，若该数是素数，则用大于它的最小素数替换该数。
- void print()：按行输出矩阵的值。

（3）编写一个程序测试该类，说明（声明）Array对象A，将一个矩阵存入对象A中，并输出矩阵的值，使用以下测试数据：

$$\begin{pmatrix} 3 & 6 & 4 & 17 \\ 8 & 5 & 9 & 10 \\ 12 & 19 & 7 & 20 \\ 4 & 14 & 21 & 23 \end{pmatrix}$$ 变换后的矩阵为 $$\begin{pmatrix} 5 & 6 & 4 & 17 \\ 8 & 7 & 9 & 10 \\ 12 & 23 & 11 & 20 \\ 4 & 14 & 21 & 29 \end{pmatrix}$$

31．建立一个类SUM，实现m行k列矩阵与k行n列矩阵的乘积。设A为m行k列的矩阵，B为k行n列的矩阵，则C=A×B。

具体要求如下：

const int m=3;

const int k=4;

const int n=3;

（1）私有数据成员

- int A[m][k]：存放m行k列矩阵。
- int B[k][n]：存放k行n列矩阵
- int (*C)[n]：指向乘积矩阵

（2）公有成员函数

- 构造函数：初始化成员数据。
- 析构函数：收回行指针。
- void process()：求矩阵的乘积。
- void print()：输出各二维数组（按行列形式）。

（3）在主程序中对该类进行测试。

32．建立一个类SUM，使用二维数组输入“Follow me”，“BASIC”，“Great wall”，“Fortran”，

“Pascal”，将它们按从小到大的顺序排列后输出。

具体要求如下：

（1）私有数据成员

- char *p[5]：存放二维数组每行的字符串的地址。

（2）公有成员函数

- SUM(char d[5][5])：构造函数，初始化成员数据。
- void process()：对二维数组中存放的字符串进行排序。
- void print()：输出二维数组中排好序的字符串。

（3）在主程序中对该类进行测试。

33．建立一个类Integer_String，把一个正整数转换为字符串。

具体要求如下：

（1）私有数据成员

- int num：要转换的正整数。
- char *s：用动态空间存储转换得到的字符串。

（2）公有成员函数

- Integer_String(int n)：用参数n初始化数据成员num。
- int f()：求数据成员num的位数。
- void fun()：把正整数num转换为字符串s。
  - void show()：输出数据成员num和s；
  - ~Integer_String()：释放动态空间。

（3）在主函数中对定义的类进行测试。用正整数 12345 初始化类Integer_String的对象test，调用相关成员函数后输出转换结果。

34．建立一个类String_Integer，把一个字符串中的数字字符转换为正整数。

具体要求如下：

（1）私有数据成员

- char *s：用动态空间存放字符串。

（2）公有成员函数

- String_Integer(char *str)：用参数str初始化数据成员s。
- operator int()：转换函数，数据成员s转换整数并返回该数。
- void show()：输出数据成员s。
- ~String_Integer()：释放动态空间。

（3）在主函数中对定义的类进行测试。定义字符数组，把由键盘输入的字符串“ab12 3c00d45ef”存入数组，并用该数组初始化类String_Integer的对象test，调用show函数输出test的数据成员s，然后把对象test赋值给整型变量n并输出，转换结果如下所示（下划线部分是从键盘输入的内容）：

请输入字符串ab12　3c00d45ef ：<u>ab12　3c00d45ef</u>

字符串为：ab12　3c00d45ef

转换得到的整数为：1230045

35．建立一个类Union求两个整数集合的并集。

具体要求如下：

（1）私有数据成员

- int *set1,len1：用动态空间set1 存储集合 1，len1 表示其元素的个数。
- int *set2,len2：用动态空间set2 存储集合 2，len2 表示其元素的个数。
- int set[20],len：用数组空间set存储并集，len表示其元素的个数

（2）公有成员函数

- Union(int *s1,int l1,int *s2,int l2)：用变量s1 和l1 初始化集合 1 及其长度，用变量s2 和l2 初始化集合 2 及其长度，并把并集的长度置为 0；
- int f(int num)：判断整数num是否属于集合 1，是返回 1，否则返回 0；
- void fun()：求集合 1 和集合 2 的并集，方法是先把集合 1 中的所有元素复制给并集，然后调用f函数把集合 2 中不属于集合 1 的元素复制给并集；
- void show()：输出集合 1、集合 2 和并集；
- ~Union()：释放动态空间。

（3） 在主函数中对定义的类进行测试。定义数组s1：{1,2,3,4,5,6,7,8}、s2：{1,3,5,7,9,11}，并用它们初始化类Union的对象obj，然后调用相关的成员函数，求并集，输出集合 1、集合 2 和并集。

36．建立一个类Intersection求两个整数集合的交集。

具体要求如下：

（1）私有数据成员

- int set[20]：用数组空间set存储集合。
- int len：表示该集合中元素的个数

（2）公有成员函数

- Intersection(int *s,int l)：用s初始化集合，用变量l初始化其长度。
- Intersection( )：把set中各元素和长度初始化为 0。
- int f(int num)：判断整数num是否属于集合，是返回 1，否则返回 0；
- Intersection operator&&(Intersection t)：重载&&，求当前对象的集合和参数对象t的集合的交集，方法是用对象t的集合中的每个元素作为参数调用f函数，若该元素属于当前对象的集合，则把它复制给交集。
- void show()：输出集合。

（3）在主函数中对定义的类进行测试。定义数组s1：{1,3,4,5,7,8}、s2：{1,2,3,5,7,9,11}，并用它们初始化类Intersection的对象obj1 和obj2，然后调用相关的成员函数输出集合；定义对象obj3，并用obj1 和obj2 的与运算符结果（交集）初始化该对象，并输出交集。

37．建立一个类NUM，为特定序列进行排序，并多次重复以统计每个数字出现的概率。

具体要求如下：

（1）私有数据成员

- int data[25]：随机生成 25 个 10000 以内的整数，不能出现重复的数字。

（2）公有数据成员

- NUM()：构造函数，初始化数组data。
- void process( )：为数组data进行排序，要求升序排列，使用排序算法不限。
- void times( )：充分多（自行输入，大于 100）次调用构造函数，统计每个数字出现的概率，每行输出一个数字的出现概率与它出现的数学期望，并验证大数定理。
- void print( )：输出数组data，每行输出 5 个数字。

（3）在主程序中定义一个对象，对该类进行测试。

提示：

① 可能需要的库函数及其用法：

n*rand()/(RAND_MAX+1.0) 生成一个 0-n之间的伪随机数，需要头文件stdlib.h支持。

srand((int)time(0)); 不断重置某些函数（例如rand()），需要头文件time.h支持）。

② 大数定理的验证：只要所有数字的出现概率除以数学期望的商在 0.95-1.05 之间，便

可以认为在 5%误差范围内验证了大数定理的正确性。

38．建立一个类Sample，对数组中元素用选择法进行升序排序。排序函数定义到Sample类的友元类Process中。

具体要求如下：

类Sample

#define Max 100;

（1）私有数据成员

- int A [MAX]：一维整型数组，存放需要排序的数。
- int n：需要排序的数的个数。

（2）公有成员函数

- Sample ()：构造函数，初始化成员数据n，初始值为 0。

友元类Process

公有成员函数

- void getdata(Sample &s)：从键盘输入数据，对数组A进行赋值。
- void selectsort(Sample &s)：对数组A中的元素进行升序排序。
- void disp(Sample &s)：输出数组中的元素。

（3）在主程序中定义对象对该类进行测试。