

作业一

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。

1. 程序设计

(1) 类的定义如下:

```
class palindrome{
    int n;
    int y;
public:
    palindrome(int x);
    ~palindrome();
    void huiwen();
    void show();
};
```

(2) 主函数定义如下:

```
int main()
{
    int a;
    cout<<"输入 a 的值"<<endl;
    cin>>a;
    palindrome p(a);
    p.huiwen();
    p.show();
    system("pause");
    return 0;
}
```

(3) 程序测试结果如下:

作业一

```

C:\V2010\Debug\text
输入a的值
1323262
该数不是回文数!
请按任意键继续. . .

```

2. 算法设计

- ①创建一个名为 **palindrome** 的类，包含两个私有成员变量 **n** 和 **y**。**n** 用于存储输入的整数，**y** 用于标记该整数是否为回文数（1 表示是，0 表示不是）。
- ②**palindrome(int x)**：这个构造函数接收一个整数 **x**，将其赋值给 **n**，并将 **y** 初始化为 0。
- ③**huiwen()**函数首先定义了一个整数数组 **b** 用于存储 **n** 的每一位数字，以及一个计数器 **p** 用于记录数字的位数。
- ④使用一个循环，将 **n** 除以 10，并存储余数到数组 **b** 中，同时更新 **p** 的值。这个循环会一直执行直到 **n** 为 0。
- ⑤接着，使用另一个循环来检查数组 **b** 的正序和倒序是否相同。如果发现不同，则设置 **t1** 为 0（表示不是回文数）并跳出循环。
- 如果 **t1** 保持为 1，说明 **n** 在十进制下是回文数。
- ⑥在 **huiwen()**函数中，定义另一个整数数组 **c** 用于存储 **n** 的二进制表示。
- 类似于十进制回文的检查，使用一个循环将 **n** 转换为二进制形式，并将每一位存储到数组 **c** 中。
- ⑥使用另一个循环来检查二进制数组 **c** 的正序和倒序是否相同。如果发现不同，则设置 **t2** 为 0（表示不是回文数）并跳出循环。
- 如果 **t2** 保持为 1，说明 **n** 在二进制下是回文数。
- ⑦在 **huiwen()**函数的最后，如果 **t1** 和 **t2** 都为 1，那么设置 **y** 为 1，表示 **n** 是一个回文数。
- ⑧**show()**函数根据 **y** 的值输出结果。如果 **y** 为 0，输出"该数不是回文数！"；如果 **y** 为 1，输出"该回文数是："后跟输入的整数 **n**。

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

作业一

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 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。

1.程序设计

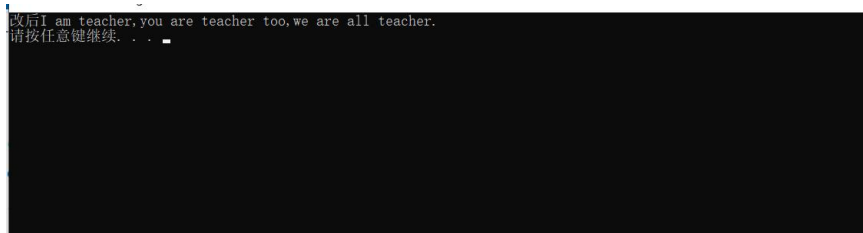
(1) 类的定义如下：

```
class cstring{
    char *str;
    char *str1;
    char *str2;
    int flag;
public:
    cstring(char*s,char s1[],char *s2);
    void replace();
    void show();
    ~cstring();
};
```

(2) 主函数定义如下：

```
int main()
{
    char s[]="I am student,you are student too,we are all student.";
    char s1[]="student";
    char s2[]="teacher";
    cstring test(s,s1,s2);
    test.replace();
    test.show();
    system("pause");
    return 0;
}
```

(3) 程序测试结果如下：

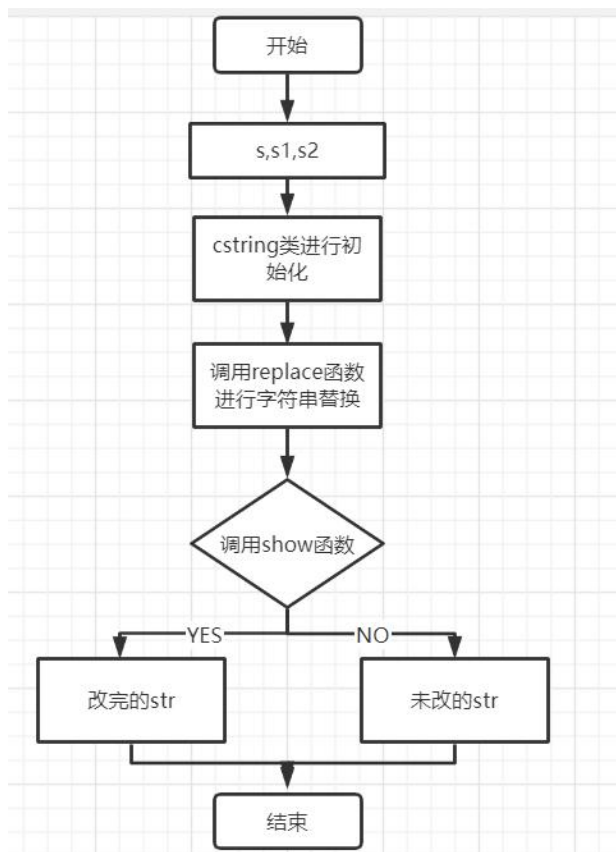


```
改后I am teacher,you are teacher too,we are all teacher.
请按任意键继续...
```

作业一

2. 算法设计

- ①创建一个名为 CString 的类，包含私有数据成员 char *str, char *str1, char *str2, int flag。
- ②在公有成员函数 CString (char *s,char s1[],char *s2)中，通过参数 s、s1 和 s2 初始化数据成员 str、str1 和 str2。将 flag 设置为默认值 0。
- ③实现公有成员函数 Replace(), 用于判断 str 字符串中是否出现 str1。如果出现，将其用 str2 进行替换，并将 flag 标记为 1；否则，不做任何操作，将 flag 标记为 0
- ④实现公有成员函数 Show(), 根据 flag 的值，在屏幕上显示相应的信息。如果替换成功，显示目标关键字、替换关键字和替换后的原始字符串；如果替换不成功，则显示原始字符串。
- ⑤实现析构函数~CString(), 用于释放动态分配的内存空间。
- ⑥在主程序中，定义字符串 char s[]="I am 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。
- ⑦创建一个 CString 类对象 test，并使用 s、s1 和 s2 初始化 test。
- ⑧调用 test.Replace()函数进行替换操作。
- ⑨调用 test.Show()函数显示替换结果。



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：

作业一

(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。

1.程序设计

(1) 类的定义如下:

```
class carray{
    int *a;
    int n;
    int b;
public:
    carray(int a[],int n,int x);
    void transform();
    void show();
    ~carray();
};
```

(2) 主函数定义如下:

```
int main()
{
    int arr[]={1,2,3,4,5,6,7,8,9,10};
    int m=10,b;
    cout<<"输入线性变换常量"<<endl;
    cin>>b;
    carray test(arr,m,b);
    test.transform();
    test.show();
    system("pause");
    return 0;
}
```

(3) 程序测试结果如下:

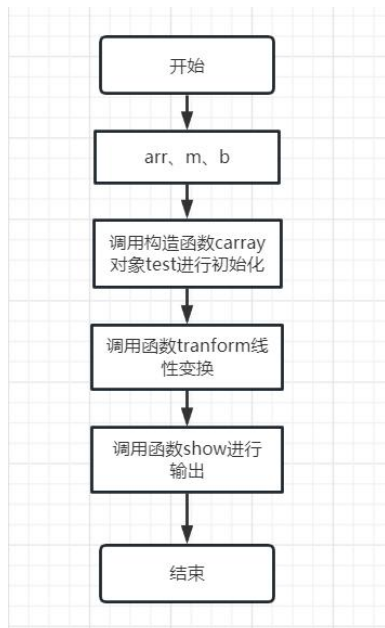
```

输入线性变换常量
5
变换后的数组为:
5      11      17      23      29      35      41      47      53      59
请按任意键继续. . .

```

2. 算法设计

- ①定义一个一维数组类 Carray，私有数据成员包括 int *a、int n、int b。
- ②实现公有成员函数 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。
- ③replace()函数遍历字符串 str，检查每个字符。
如果当前字符与 str1 的第一个字符匹配，开始检查 str1 是否完全匹配 str 中的子串。
使用两个指针 n1 和 n2 分别遍历 str 和 str1，如果 str1 中的任何字符与 str 中的对应字符不匹配，或者 str 中的字符串提前结束（即遇到空字符'\0'），则设置 y 为 0，表示不进行替换。
如果 str1 完全匹配，并且没有提前结束，那么执行替换操作。创建一个临时字符串 pp，将 str2 中的字符复制到 str 中 str1 出现的位置，覆盖原有的 str1 子串。
设置 flag 为 1，表示进行了替换。
- ④实现公有成员函数 Show(), 在屏幕上显示数组元素。
- ⑤实现析构函数 ~Carray(), 释放动态分配的空间。
- ⑥在主程序中定义一个数组 int arr[] = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10} 作为原始数组，从键盘输入一个整数 b 作为线性变换的常量。
- ⑦定义一个 Carray 类对象 test，用 arr 初始化 test。
- ⑧调用 Transform() 函数对数组进行变换。
- ⑨调用 Show() 函数显示变换后的数组元素。



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。

作业一

1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	16

4	8	12	16
3	7	11	15
2	6	10	14
1	5	9	13

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。

1.程序设计

(1) 类的定义如下：

```
class array{
    int a[4][4];
public:
    array(int a1[][4],int n);
    void xuanzhuan();
    void show();
    ~array();
};
```

(2) 主函数定义如下：

```
int main()
{
    int b[][4]={1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16};
    int n=4;
    array test(b,n);
    test.xuanzhuan();
    test.show();
    system("pause");
    return 0;
}
```

(3) 程序测试结果如下：

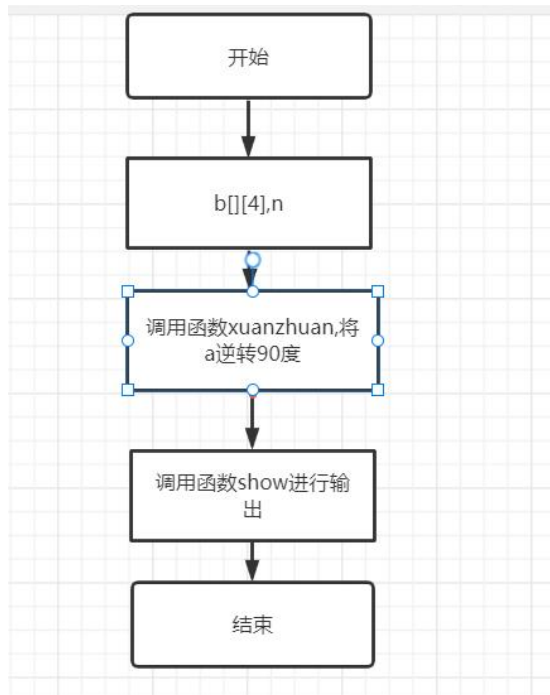
```

旋转后的矩阵为:
    4      8      12     16
    3      7      11     15
    2      6      10     14
    1      5      9      13
Press any key to continue

```

2. 算法设计

- ① 定义一个类 Array，设置私有数据成员 `int a[4][4]`，用于存放方阵。
- ② 在类 Array 中定义构造函数 `Array(int a1[][4], int n)`，用给定的参数 `a1` 初始化数据成员 `a`。将传入的二维数组 `a1` 的元素复制给成员变量 `a`。
- ③ `xuanzhuang()`：这个成员函数用于将矩阵顺时针旋转 90 度。
首先，声明一个新的 4x4 二维数组 `b`，用于存储矩阵 `a` 的当前状态。
接着，通过双重 for 循环，将矩阵 `a` 的每个元素复制到数组 `b` 中。
然后，再次使用双重 for 循环，但这次是为了将数组 `b` 的元素重新排列到矩阵 `a` 中。新矩阵的元素 `a[3-j][i]`（即新矩阵的第 `i` 行第 `j` 列）被设置为原矩阵 `b` 的元素 `b[i][j]`（即原矩阵的第 `i` 行第 `j` 列）。这样，原矩阵的列变成了新矩阵的行，实现了顺时针旋转 90 度的效果。
- ④ 在类 Array 中定义公有成员函数 `void show()`，在屏幕上显示数组元素。
- ⑤ 在主程序中，定义一个二维数组 `int b[][4]`，并初始化为原始数组。
- ⑥ 创建一个对象，调用类 Array 的构造函数，将原始数组传入对象进行初始化。
- ⑦ 调用对象的 `xuanzhuang()` 方法，实现逆时针 90 度的旋转。
- ⑧ 调用对象的 `show()` 方法，显示旋转后的方阵。



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

作业一

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。

1. 程序设计

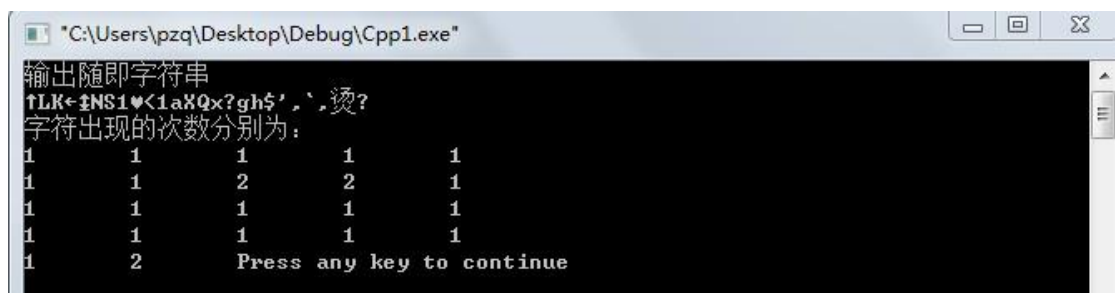
(1) 类的定义如下:

```
class Num{
    char data[25];
    int num[128];
public:
    Num();
    void process();
    void print();
    ~Num();
};
```

(2) 主函数定义如下:

```
int main()
{
    Num test;
    test.process();
    test.print();
    system("pause");
    return 0;
}
```

(3) 程序测试结果如下:



```
"C:\Users\pzz\Desktop\Debug\Cpp1.exe"
输出随机字符串
tLK<1NS1♥<1aXQx?gh$'`,`烫?
字符出现的次数分别为:
1      1      1      1      1
1      1      2      2      1
1      1      1      1      1
1      1      1      1      1
1      2      1      1      1
Press any key to continue
```

3. 算法设计

①在 main()函数中, 调用 srand(time(0))来初始化随机数生成器。这确保了每次程序运行时,

作业一

生成的随机数序列都是不同的。time(0)获取当前时间的秒数，用作随机数种子。

②Num(): 这个构造函数在创建 Num 对象时被调用。

创建一个足够大的字符数组 data，用于存储随机生成的字符串（长度为 25）。

创建一个整型数组 num，用于存储每个 ASCII 字符出现的次数（大小为 128，因为 ASCII 码的范围是 0-127）。

使用一个循环，通过 rand() % 128 生成随机的 ASCII 码，并将其作为字符存储在 data 数组中。这个循环运行 25 次，因为 data 数组的长度是 25。

③process(): 这个成员函数用于统计 data 数组中每个字符出现的次数。

初始化一个计数器 x，用于跟踪 num 数组中存储字符计数的位置。

对于 num 数组中的每个索引（从 1 到 128），使用一个内部循环来遍历 data 数组。

对于 data 数组中的每个字符，检查它是否与当前处理的 ASCII 码值相匹配。如果匹配，增加对应的计数器 n。

将每个 ASCII 码值的总出现次数存储在 num 数组中，并将计数器 x 递增。

④print(): 这个成员函数用于输出随机生成的字符串和每个字符出现的次数。

输出字符串 data，这是由随机字符组成的。

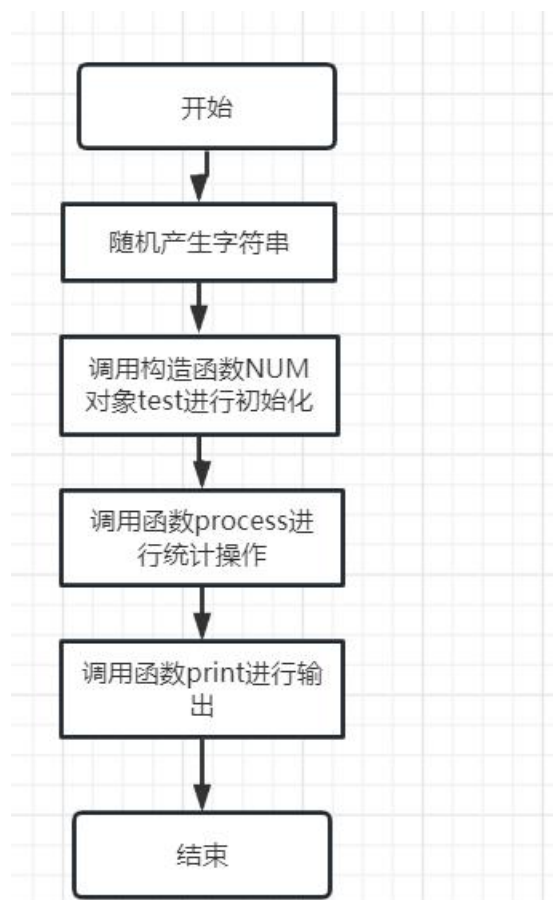
初始化一个变量 x，用于在输出时格式化换行。每输出 5 个字符的计数，就换一行。

遍历 num 数组，对于每个 ASCII 码值，如果对应的计数大于 0，则输出该计数。输出格式为计数后跟一个制表符 (\t)，然后是 ASCII 码值。

⑤在主程序中定义一个 NUM 类对象 test，用于对该类进行测试。

⑥调用 process() 函数进行字符统计。

⑦调用 print() 函数输出字符及其出现的个数。



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：

3	6	4	17		5	6	4	17
8	5	9	10	变换后的矩阵为	8	7	9	10
12	19	7	20		12	23	11	20
4	14	21	23		4	14	21	29

1. 程序设计

（1）类的定义如下：

```
class array{
    int x[4][4];
    int count;
public:
    array(int a[4][4]);
    int fun(int );
    void encode();
    void print();
    ~array();
};
```

（2）主函数定义如下：

```
int main()
{
    int a[4][4]={3,6,4,17,8,5,9,10,12,19,7,20,4,14,21,23};
    array test(a);
    test.encode();
    test.print();
}
```

```

system("pause");
return 0;
}

```

(3) 程序测试结果如下：



```

"C:\Users\pzz\Desktop\Debug\Cpp1.exe"
变换后的矩阵为：
5      6      4      17
8      7      9      10
12     23     11     20
4      14     21     29
Press any key to continue

```

4. 算法设计

算法详细设计

① `array(int a[4][4])`: 这个构造函数接收一个 4x4 的整数数组 a 作为参数。通过双重 for 循环，将参数数组 a 的每个元素复制到对象的内部数组 x 中。初始化成员变量 count 为 0，用于记录编码过程中替换元素的次数。

② `fun(int num)`: 这个成员函数用于判断一个整数 num 是否为素数。从 2 开始到 num 的平方根（因为一个合数必有一个因子小于或等于它的平方根），检查 num 是否能被任何整数整除。如果 num 能被整除，则返回 0（表示 num 不是素数）。如果没有找到能整除 num 的整数，则返回 1（表示 num 是素数）。

③ `encode()`: 这个成员函数用于对矩阵 x 进行编码。通过双重 for 循环遍历矩阵的每个元素。对于矩阵的第一行和第三列的元素，如果元素的行索引不等于列索引（即不是对角线元素），则跳过该元素。如果当前元素是素数（通过调用 fun 函数判断），则尝试找到一个非素数来替换它。从当前元素的值加 1 开始，循环寻找一个非素数，一旦找到，就替换当前元素，并增加 count 的值，表示替换了一个新的元素。

④ `print()`: 这个成员函数用于打印变换后的矩阵。使用双重 for 循环遍历矩阵的每个元素，并使用 cout 输出流打印每个元素。每个元素之间用制表符 \t 分隔，每行元素打印完毕后换行。

⑤ `main()`: 定义了一个 4x4 的整数数组 a，并初始化为给定的值。创建 array 类的对象 test，并将数组 a 作为参数传递给构造函数。调用 test.encode() 方法来执行编码过程。调用 test.print() 方法来显示编码后的矩阵。使用 system("pause") 暂停程序，等待用户操作。

⑥ 在主程序中，定义一个对象并进行测试。

