

计算机程序设计基础 (C++)

实验报告

专业班级：_____

学 号：_____

姓 名：_____

实验报告成绩：

实验	实验一	实验二	实验三	实验四	实验五	总评
成绩						

批阅教师：

实验三 函数

一、实验目的

本实验主要培养、训练学生对函数的理解，要求：

1. 掌握函数的定义、声明的方法；
2. 掌握函数的编写要求；
3. 掌握函数的调用方法；
4. 掌握函数参数的传递方法；
5. 掌握变量的作用域；
6. 掌握多文件编程方法。

二、实验内容与要求

1. 输入自然数 m 和 n ,

(1) 求他们的最大公约数 (或称最大公因数)。

要求输入、输出在主函数中进行，求公约数由函数实现。

(2) 在函数中求最大公约数与最小公倍数。(提示：使用引用参数)

2. 编写程序满足：声明一个函数，判断一个整数是否为素数，使用如下函数头：

`bool is_prime(int num)` ,如果 `num` 是素数函数返回 `true`，否则返回 `false`；

利用函数 `is_prime` 找出前 200 个素数，并按每行 10 个输出：

2 3 5 7 11 13 17 19 23 29

3、编程实现摄氏温度到华氏温度的转换：

编写一个头文件，包含下面两个函数：

```
double celsius_to_fah(double cel) //摄氏温度到华氏温度
double fahrenheit_to_cels(double fah) //华氏温度到摄氏温度
```

实现头文件，并编写测试程序，调用函数显示如下结果：

Celsius	Fahrenheit		Fahrenheit	Celsius
40.0	105.0		120.0	48.89
39.0	102.0		110.0	43.33
.....
31.0	87.8		30.0	-1.11

（测试程序为主模块，即 `main()` 函数所在的 CPP 文件，头文件 `mytemperature.h` 只有函数声明；函数定义写在另一 CPP 文件 `mytemperature.cpp`）

4、创建名为 `mytriangle.h` 的头文件，包括：

```
bool is_valid(double side1,double side2,double side3)
```

```
double_area(double side1,double side2, double side3)
```

```
面积=sqrt(s(s-side1)(s-side2)(s-side3))
```

其中 $s=(side1+side2+side3)/2$

写测试程序：读取三角形三边长，如输入合法，计算面积，否则输出错误信息。

（测试程序为主模块，即 `main()` 函数所在的 CPP 文件，头文件 `mytriangle.h` 只有函数声明；函数定义写在另一 CPP 文件 `mytriangle.cpp`）

3 与 4 选一个完成

5、猴子吃桃：猴子第一天摘若干桃子，当即吃了一半，还不过瘾，又吃了一个。第二天又将剩下的桃子吃掉一半，又多吃一个，以后每天如此，到第 10 天，发现只剩最后一个桃子，问，第一天猴子共摘多少桃子（用递归实现）。

三、实验思考题

1. 本实验中函数中返回的值为什么与函数类型一致？

函数返回值是按照函数声明时的返回类型来确定的，这种一致性有助于确保程序的类型安全性和正确性。函数声明时的返回类型是为了告诉编译器根据该类型去开辟内存空间，以及告诉编译器函数的返回值应该被解释为哪种类型，以便在函数调用时正确地处理返回值。当函数被调用时，编译器将根据具体的编译器实现和函数调用约定，为返回值分配足够的内存来存储其数据类型，以便在函数执行结束后将返回值存储在该内存位置。

函数声明时的返回类型和一致性主要是为了提供类型安全性和代码可读性。编译器会根据函数的返回类型来决定如何存储和返回值，以保障程序的正确性和可维护性。

2. 本实验中主函数调用函数时采用的是何种传递方式？

值传递。

四、算法分析，程序结果

1.

```
#include <iostream>
using namespace std;
int& gongyinshu(int x, int y);
int& gongbeishu(int x, int y);
int main()
{
    cout << "请输入两个自然数：";
    int a, b, m, n;
    while (true) //确保输入的是两个自然数
    {
        cin >> a >> b;
        if (a >= 0 && b >= 0)
        {
            m = a;
            n = b;
            break;
        }
        else
            cout << "请重新输入两个自然数：";
    }

    int yinshu = gongyinshu(m, n);
    int beishu = gongbeishu(m, n);
    cout << "他们的最大公因数是：" << yinshu << endl;
    cout << "他们的最小公倍数是：" << beishu << endl;
    return 0;
}

int& gongyinshu(int x, int y)//求最大公因数
{
    int z1;
    for (int i = 1; i <= x && i <= y; i++)
    {
        if ((x % i == 0) && (y % i == 0))
            z1 = i;
    }
    return z1;
}

int& gongbeishu(int x, int y)//求最小公倍数
{
    int i, z2;
    i = (x > y ? y : x);
    while (true)
    {
        if (i % x == 0 && i % y == 0)
        {
            z2 = i;
            break;
        }
        i++;
    }
    return z2;
}
```

Microsoft Visual Studio 调试 × + -

请输入两个自然数：40 30
他们的最大公因数是：10
他们的最小公倍数是：120

D:\Learning\C++\experiment\第二次实验\experiment3\3-1\x64\Debug\3-1.exe (进程 9696)已退出，代码为 0。
按任意键关闭此窗口。 . . .

2.

```
#include <iostream>
using namespace std;
bool zhishu(int x);
int main()
{
    for (int i = 2, k = 1; i <= 200; i++)
    {
        if (zhishu(i))
        {
            cout << i << '\t';
            if (k >= 10 && k % 10 == 0) //当输出十个数后就换行
            {
                cout << endl;
            }
            k++;
        }
    }
    return 0;
}

bool zhishu(int x)
{
    if (x == 2)
        return true;
    else
    {
        for (int y = 2; y < x; y++) //能被除了1和自己本身整除的数不是质数
        {
            if (x % y == 0)
            {
                return false;
            }
        }
        return true;
    }
}
```

Microsoft Visual Studio 调试 × + ▾

2	3	5	7	11	13	17	19	23	29
31	37	41	43	47	53	59	61	67	71
73	79	83	89	97	101	103	107	109	113
127	131	137	139	149	151	157	163	167	173
179	181	191	193	197	199				

D:\Learning\C++\experiment\第二次实验\experiment3\3-2\x64\Debug\3-2.exe (进程 17200)已退出，代码为 0。
按任意键关闭此窗口. . .

3.

mytemperature.h

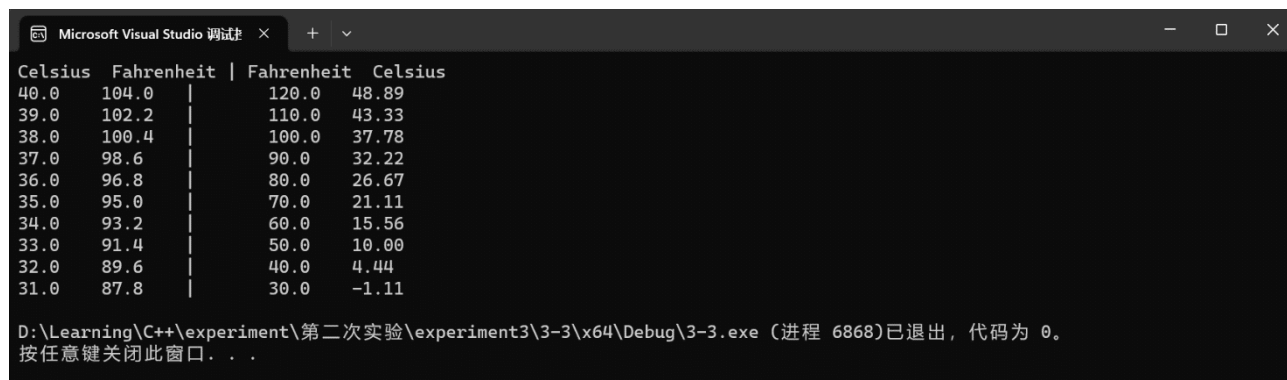
```
double celsius_to_fah(double cel);  
double fahrenheit_to_cels(double fah);
```

mytemperature.cpp

```
#include <iostream>  
#include "mytemperature.h"  
using namespace std;  
double celsius_to_fah(double cel)  
{  
    double fah;  
    fah = 9 * cel / 5 + 32;  
    return fah;  
}  
double fahrenheit_to_cels(double fah)  
{  
    double cel;  
    cel = (fah - 32) * 5 / 9;  
    return cel;  
}
```

测试程序

```
#include <iostream>  
#include <iomanip>  
#include "mytemperature.h"  
using namespace std;  
int main()  
{  
    cout << "Celsius " << " Fahrenheit " << "|" << " Fahrenheit " << " Celsius" << endl;  
    for (double a = 40.0, b = 120.0; a != 30.0 && b != 20.0; a--, b = b - 10)  
    {  
        cout << fixed << setprecision(1) << a << '\t' << fixed << setprecision(1) << celsius_to_fah(a) << '\t' << "|" << '\t' << fixed << setprecision(1) << b << '\t' << fixed << setprecision(2) << fahrenheit_to_cels(b) << endl;  
    }  
    return 0;  
}
```



```
Microsoft Visual Studio 调试  × + -
```

Celsius	Fahrenheit	Fahrenheit	Celsius
40.0	104.0	120.0	48.89
39.0	102.2	110.0	43.33
38.0	100.4	100.0	37.78
37.0	98.6	90.0	32.22
36.0	96.8	80.0	26.67
35.0	95.0	70.0	21.11
34.0	93.2	60.0	15.56
33.0	91.4	50.0	10.00
32.0	89.6	40.0	4.44
31.0	87.8	30.0	-1.11

D:\Learning\C++\experiment\第二次实验\experiment3\3-3\x64\Debug\3-3.exe (进程 6868)已退出，代码为 0。
按任意键关闭此窗口。 . . .

4.

mytriangle.h

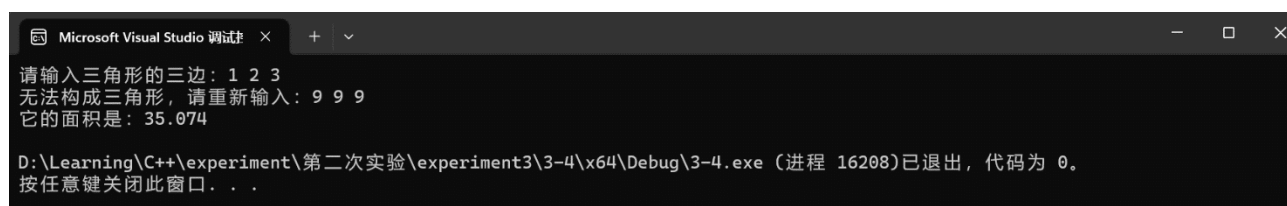
```
bool is_valid(double sidel, double side2, double side3);  
double area(double sidel, double side2, double side3);
```

mytriangle.cpp

```
#include <iostream>  
#include <math.h>  
using namespace std;  
bool is_valid(double sidel, double side2, double side3)  
{  
    if (sider + side2 > side3 && side2 + side3 > sider && sider + side3 > side2)  
        return true;  
    else return false;  
}  
double area(double sider, double side2, double side3)  
{  
    double s = (sider + side2 + side3) / 2;  
    double mianji = sqrt(s * (s - sider) * (s - side2) * (s - side3));  
    return mianji;  
}
```

测试程序

```
#include <iostream>  
#include <iomanip>  
#include "mytriangle.h"  
using namespace std;  
int main()  
{  
    cout << "请输入三角形的三边: ";  
    double a, b, c;  
    while (true)  
    {  
        cin >> a >> b >> c;  
        if (is_valid(a, b, c))  
            break;  
        else//输入的三边无法构成三角形, 则做出提示, 并要求重新输入  
        {  
            cout << "无法构成三角形, 请重新输入: ";  
        }  
    }  
    cout << "它的面积是: " << area(a, b, c) << endl;  
    return 0;  
}
```



```
Microsoft Visual Studio 调试  x + v  
请输入三角形的三边: 1 2 3  
无法构成三角形, 请重新输入: 9 9 9  
它的面积是: 35.074  
  
D:\Learning\C++\experiment\第二次实验\experiment3\3-4\x64\Debug\3-4.exe (进程 16208)已退出, 代码为 0。  
按任意键关闭此窗口. . .
```

5.

```
#include <iostream>
using namespace std;
unsigned int jisuan(int taozi);
int main()
{
    cout << "第一天猴子摘了" << jisuan(1) << "个桃子。" << endl;
    return 0;
}
unsigned int jisuan(int day)
{
    int taozi;
    if (day == 10) //第十天发现只有1个桃子，说明第十天没有吃
    {
        taozi = 1;
    }
    else
        taozi = (jisuan(day + 1) + 1) * 2;
    return taozi;
}
```

Microsoft Visual Studio 调试 × + ▾

第一天猴子摘了1534个桃子。

D:\Learning\C++\experiment\第二次实验\experiment3\3-5\x64\Debug\3-5.exe (进程 16176)已退出，代码为 0。
按任意键关闭此窗口。 . . .

五、遇到的问题与解决方法

在实验五中，最开始是用 `for` 循环完成的，但后来发现题目中要求用递归实现，而相关的知识点有所遗忘，最后通过重温教材重新掌握了函数中递归的使用。

六、体会

C++ 知识点很多，需要我及时复习总结，并勤加练习使用避免遗忘。

实验四 数组与指针

【实验目的】

- 1、进一步加深对数组的理解，掌握数组的定义方法；
- 2、掌握数组的处理方法、数组作为函数参数的使用方法，以及搜索与排序的应用。
- 3、掌握指针的概念、指针变量定义格式以及指针的运算；
- 4、掌握指针与数组、函数的关系；
- 5、理解内存动态分配的含义、熟练掌握内存动态分配方法；
- 6、掌握递归函数的定义方法。

【实验内容与步骤】

（一）数组

- 1、打印不同的数：

编写一个程序，读入 10 个数，输出其中不同的数（即如果一个数出现多次，只打印一次）。

提示：读入的数如果是一个新的值，则将其存入一个数组。否则，将其丢弃。输入完毕后，数组中保存的就是不同的数。

下面是一个运行样例：

```
Enter ten numbers: 1 2 3 2 1 6 3 4 5 2 ↵Enter
The distinct numbers are: 1 2 3 6 4 5
```

- 2、起泡排序：

利用起泡排序算法编写一个排序函数。起泡排序算法分若干趟对数组进行处理。每趟处理中，对相邻元素进行比较。若为降序，则交换；否则，保持原顺序。此技术被称为起泡排序（bubble sort）或下沉排序（sinking sort），因为较小的值逐渐地“冒泡”到上部，而较大值逐渐下沉到底部。

算法可描述如下：

```
bool changed = true;
do
{
    changed = false;
    for (int j = 0; j < listSize - 1; j++)
        if (list[j] > list[j+1])
        {
            swap list[j] with list[j+1];
            changed = true;
        }
} while (changed);
```

很明显，循环结束后，列表变为升序。容易证明 do 循环最多执行 listSize - 1 次。

编写测试程序，读入一个含有 10 个双精度数字的数组，调用函数并显示排列后的数字。

- 3、游戏：存物柜问题：

一个学校有 100 个储物柜，100 个学生。开学第一天所有储物柜都是关闭的。第一个学生（记为 S1）来到学校后，打开所有的储物柜。第二个学生 S2，从第二个储物柜（记为 L2）开始，每隔两个储物柜，将它们关闭。第三个学生 S3 从第三个储物柜 L3 开始，每隔三个，将它们的状态改变（开着的关上，关着的打开）。学生 S4，从 L4 开始，每隔四个改变它们的状态。学生 S5，从 L5 开始，每隔五个改变状态。依此类推，直至学生 S100 改变 L100 的状态。

当所有学生完成这个过程，那些储物柜是开着的？编写一个程序求解此问题，显示所有开着的柜子号码，号码之间用一个空格隔开。

提示：使用一个 100 个布尔型元素的数组，每个元素代表储物柜是开(true)或关(false)。最初所有的储物柜都是关闭的。

4、合并两个排列好的数组：

编写如下函数，合并两个排列好的数组，形成一个新的排列好的数组。

```
void merge(const int list1[], int size1, const int list2[], int size2, int list3[])
```

使用 size1+size2 次比较实现函数。编写测试程序，提示用户输入两个排列好的数组，并显示合并以后的数组。下面是一个运行样例。注意，输入数据的第一个数字是数组的元素数，而不是数组的一部分。假定数组大小不超过 80。

```
Enter list1: 5 1 5 16 61 111 ↵Enter
Enter list1: 4 2 4 5 6 ↵Enter
The merged list is 1 2 4 5 5 6 16 61 111
```

5、检验子串：

编写如下函数，检验 C 字符串 s1 是否是 C 字符串 s2 的子串。如果匹配，返回 s1 在 s2 中的下标，否则返回-1。

```
int indexOf(const char s1[], const char s2[])
```

编写测试程序，读入两个 C 字符串，检验 C 字符串 s1 是否是 C 字符串 s2 的子串。下面是程序的运行样例：

```
Enter the first string: welcome ↵Enter
Enter the second string: We welcome you! ↵Enter
indexOf("welcome", "We welcome you!") is 3
```

```
Enter the first string: welcome ↵Enter
Enter the second string: We invite you! ↵Enter
indexOf("welcome", "We invite you!") is -1
```

6、字符串中每个字母出现的次数：

请使用如下函数头编写函数，数出字符串中每个字母出现的次数。

```
void count(const char s[], int counts[])
```

counts 是一个有 26 个元素的整数数组。const[0], const[1], ..., const[25]分别记录 a, b, ..., z 出现的次数。字母不分大小写，例如字母 A 和字母 a 都被看作 a。

编写测试程序，读入字符串并调用 count 函数，显示非零的次数。下面是程序的一个运行样例：

```
↵Enter
```

Enter a string: Welcome to New York!

c: 1 times
e: 3 times
k: 1 times
l: 1 times
m: 1 times
n: 1 times
o: 3 times
r: 1 times
t: 1 times
w: 2 times
y: 1 times

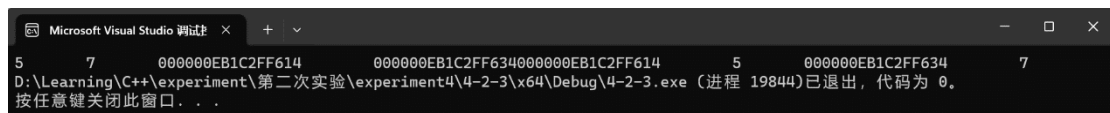
(二) 指针

1、上机验证下列程序的运行结果（有错误的话自己补充完善）

(1) void main()

```
{  
    int i,j,*pi,*pj;    //此处的*表示定义指针变量，而非间接运算符  
    pi=&i;  
    pj=&j;  
    i=5;j=7;  
    cout<<i<<'\t'<<j<<'\t'<<pi<<'\t'<<pj;  
    cout<<&i<<'\t'<<*&i<<'\t'<<&j<<'\t'<<*&j;  
}
```

运行结果：



上述结果中，pi 与 &i, pj 与 &j 是地址值，随编译程序而变化，不确定。

(2) int main() //C 语言程序，要了解

```
{  
    int a[]={1,2,3};  
    int *p,i;  
    p=a;    //将数组 a 首地址送给 p  
    for (i=0;i<3;i++)  
        printf("%d,%d,%d,%d\n",a[i],p[i],*(p+i),*(a+i));    //与 cout 功能差不多  
}
```

运行结果：

1,1,1,1

2,2,2,2

3,3,3,3

通过这两道题目，希望学生掌握数组元素与指向数组的指针的不同。

a[i]表示数组中下标为 i 的元素。

a[i]←p[i]←*(p+i)←*(a+i)

a 是数组名，表示数组首地址，(p+i)表示数组中第 i 个元素的地址，*(p+i) 相当于 a[i]。

(3)通过如下的问题理解递归函数的定义与调用（递归未讲，可以后做）

```

//#include "stdio.h"

void f(char *st,int i)
{
    st[i]='\0';
    cout<<st;    // printf("%s\n",st);
    if (i>1) f(st,i-1);
}

void main()
{
    char st[]="abcd";
    f(st,4);
}

```

补充完整，运行时输出为 abcdabcaba

(4)下面程序的主函数中能保证 p[0]输出 1，p[1]输出 2 吗？如何修改以保证之（提示：在函数 f 中使用 new 生成动态数组；在 main 中用 delete 释放。）

```

#include<iostream>
using namespace std;
int *f()
{
    int list[]={1,2,3,4};
    return list;
}

void main()
{
    int *p=f();
    cout<<p[0]<<endl;
    cout<<p[1]<<endl;
}

```

```

#include<iostream>
using namespace std;
int* f()
{
    int* list = new int [4];
    list[0] = 1;
    list[1] = 2;
    list[2] = 3;
    list[3] = 4;
    return list;
}

void main()
{
    int* p = f();
    cout << p[0] << endl;
    cout << p[1] << endl;
    delete[] p;
}

```

2、程序设计

(1)编写函数检查字符串 s1 是否为字符串 s2 的子串，若是，返回第一次匹配的下标，否则返回-1。在主程序中输入字符串 s1 与 s2，调用函数实现。

函数原型：int indexof(const char *s1,const char *s2);

(2)编写一个函数将以字符串形式表示的一个 16 进制数转换为 10 进制数，并在主函数中测试。函数原型 int parseHex(const char *const hexString);

如：调用函数 parseHex("A5");返回 165

(3)主程序中建立一动态数组（使用 new），数组元素及元素个数由键盘输入，动态调试观察指针及指针指向的内容；设计一个函数对数组由小到大排序；主程序中用指针方式输出数组元素；最后释放数组内存（delete）。

【完成实验报告】

实验报告只要求写程序设计部分

三、算法分析，程序结果

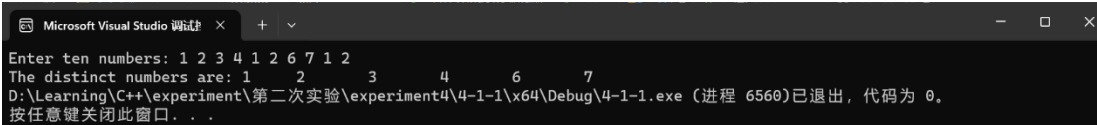
(一) 数组

1.

```
int main()
{
    double a[10], b;
    int k = 0;
    cout << "Enter ten numbers: ";
    for (int i = 1; i <= 10; i++)//i来计数，保证输入10次，当满足条件时，a[k]等于输入值，然后k+1
    {
        cin >> b;
        if (i == 1)//第1个数字肯定可以计入
        {
            a[k] = b;
            k++;
        }
        else
        {
            for (int j = 0; j < k; j++)//判断第i个数字之前是否有相同数字计入了
            {
                if (a[j] == b)
                    break;
                else
                {
                    if ((j == (k - 1)) && (a[k-1] != b))
                    {
                        a[k] = b;
                        k++;
                    }
                }
            }
        }
    }

    cout << "The distinct numbers are: ";
    for (int c = 0; c < k; c++)
    {
        cout << a[c] << ' ';
    }

    return 0;
}
```



Enter ten numbers: 1 2 3 4 1 2 6 7 1 2
The distinct numbers are: 1 2 3 4 6 7
D:\Learning\C++\experiment\第二次实验\experiment4\4-1-1\x64\Debug\4-1-1.exe (进程 6560)已退出，代码为 0。
按任意键关闭此窗口...

2.

```
#include <iostream>
using namespace std;
void swap(double* ip[10]);
int main()
{
    double list[10];
    double* list_ip[10];
    cout << "输入十个数: ";
    for (int a = 0; a < 10; a++)
    {
        double b;
        cin >> b;
        list[a] = b;
    }
    for (int c = 0; c < 10; c++)
    {
        list_ip[c] = &list[c];
    }
    swap(list_ip);
    cout << "由小到大排序为: ";
    for (int f = 0; f < 10; f++)
    {
        cout << list[f] << '\t';
    }
    return 0;
}
void swap(double* ip[10])//起泡排序
{
    bool changed = true;
    do
    {
        changed = false;
        for (int d = 0; d < 9; d++)
        {
            if (*ip[d] > *ip[d + 1])
            {
                double e = *ip[d];
                *ip[d] = *ip[d + 1];
                *ip[d + 1] = e;
                changed = true;
            }
        }
    } while (changed);
}
```

Microsoft Visual Studio 调试 × + -

输入十个数: 12.21 23.6 77.78 12 -50.13 98 123.321 45.5 46.67 10.023
由小到大排序为: -50.13 10.023 12 12.21 23.6 45.5 46.67 77.78 98 123.321
D:\Learning\C++\experiment\第二次实验\experiment4\4-1-2\x64\Debug\4-1-2.exe (进程 21416)已退出, 代码为 0。
按任意键关闭此窗口. . .

3.

```
int main()
{
    double L[100];
    double* L_change[100];
    for (int a = 0; a < 100; a++)//最开始所有柜门都是关闭的
    {
        L[a] = false;
        L_change[a] = &L[a];
    }

    change(L_change);
    cout << "开启的柜门编号为: " << endl;
    for (int e = 0; e < 100; e++)
    {
        if (L[e] == true)
            cout << e + 1 << '\t';
    }

    return 0;
}

void change(double* ip[100])
{
    for (int c = 1; c <= 100; c++)//c代表第几个学生
    {
        for (int d = c; d <= 100; d = d + c)//d代表第几个柜子
        {
            if (*ip[d - 1] == false)
                *ip[d - 1] = true;
            else *ip[d - 1] = false;
        }
    }
}
```

Microsoft Visual Studio 调试 × + ▾

开启的柜门编号为:
1 4 9 16 25 36 49 64 81 100

D:\Learning\C++\experiment\第二次实验\experiment4\4-1-3\x64\Debug\4-1-3.exe (进程 11104)已退出, 代码为 0。
按任意键关闭此窗口。 . . .

4.

```
#include <iostream>
using namespace std;
void merge(double* ip1[80], double* ip2[80], double* ip[160], int len1, int len2);
void swap(double* ip[160], int len1, int len2);
int main()
{
    double list1[80], list2[80], list_merged[160];
    double* list1_ip[80];
    double* list2_ip[80];
    double* list_ip[160];
    int length1, length2;
    cout << "分别输入两个数组，输入数据的第一个数字是数组的元素数，数组大小不超过80。" << endl;
    cout << "Enter list1: ";
    for (int a = 0; a < 80; a++)
    {
        double b;
        cin >> b;
        if (a == 0)
            length1 = b; // 第一个数字作为数组的元素数
        else
        {
            list1[a - 1] = b;
            list1_ip[a - 1] = &list1[a - 1];
            if (a == length1)
                break;
        }
    }
    cout << endl;
    cout << "Enter list2: ";
    for (int a = 0; a < 80; a++)
    {
        double b;
        cin >> b;
        if (a == 0)
            length2 = b; // 第一个数字作为数组的元素数
        else
        {
            list2[a - 1] = b;
            list2_ip[a - 1] = &list2[a - 1];
            if (a == length2)
                break;
        }
    }
    for (int a = 0; a < 160; a++)
    {
        list_ip[a] = &list_merged[a];
    }
    merge(list1_ip, list2_ip, list_ip, length1, length2); // 合并数组
    swap(list_ip, length1, length2); // 对合并后的数组排序
    cout << "The merged list is ";
    for (int c = 0; c < length1 + length2; c++)
    {
        cout << list_merged[c] << " ";
    }
    return 0;
}

void merge(double* ip1[80], double* ip2[80], double* ip[160], int len1, int len2) // 这个函数用于合并数组
{
    for (int i = 0; i < len1; i++)
    {
        *ip[i] = *ip1[i];
    }
    for (int k = len1; k < len1 + len2; k++)
    {
        *ip[k] = *ip2[k - len1];
    }
}
```



```

void swap(double* ip[160], int len1, int len2)//这个函数用于对数组进行起泡排序
{
    bool changed = true;
    do
    {
        changed = false;
        for (int j = 0; j < len1+len2-1; j++)
        {
            if (*ip[j] > *ip[j + 1])
            {
                double h = *ip[j];
                *ip[j] = *ip[j + 1];
                *ip[j + 1] = h;
                changed = true;
            }
        }
    } while (changed);
}

```

Microsoft Visual Studio 调试 × + ▾

分别输入两个数组，输入数据的第一个数字是数组的元素数，数组大小不超过80。
Enter list1: 6 123 545 68 78 154 998

Enter list2: 7 45.54 23.2 65.9 78.879 544.4 -32.65 23
The merged list is -32.65 23 23.2 45.54 65.9 68 78 78.879 123 154 544.4 545 998
D:\Learning\C++\experiment\第二次实验\experiment4\4-1-4\x64\Debug\4-1-4.exe (进程 12800)已退出，代码为 0。
按任意键关闭此窗口。 . . .

5.

```
#include <iostream>
#include <cstring>
using namespace std;
int indexOf(const char s1[], const char s2[]);
int main()
{
    char s1[100], s2[100];
    char i;
    cout << "Enter the first string: ";
    cin.getline(s1, 100);
    cout << "Enter the second string: ";
    cin.getline(s2, 100);
    cout << "indexOf( " << s1 << " , " << s2 << " ) is: " << indexOf(s1, s2);
    return 0;
}

int indexOf(const char s1[], const char s2[])
{
    int length1 = strlen(s1), length2 = strlen(s2);
    if (length1 > length2)
        return -1; // 如果s1长度比s2还大, s1不可能是s2的子字符串
    else
    {
        for (int i = 0; i <= length2 - length1; i++)
        {
            int match = 0;
            for (int k = 0, j = i; k < length1; k++, j++)
            {
                if (s1[k] == s2[j])
                    match++;
            }
            if (match == length1) // 当s1和s2对应的字符相同时, match++, 当判断了length1个字符后, 如果match=length1, 那么s1即为s2的子字符串
                return i;
        }
        return -1;
    }
}
```

Microsoft Visual Studio 调试 × + -

Enter the first string: love
Enter the second string: C++ is so fantastic that we all love it!
indexOf("love", "C++ is so fantastic that we all love it!") is: 32
D:\Learning\C++\experiment\第二次实验\experiment4\4-1-5\x64\Debug\4-1-5.exe (进程 18716)已退出, 代码为 0。
按任意键关闭此窗口...

6.

```
#include <iostream>
#include <cstring>
#include <cctype>
using namespace std;
void count(const char s[], int* counts[26]);
int main()
{
    char s[100];
    int counts[26];
    int* counts_ip[26];
    cout << "Enter a string: ";
    cin.getline(s, 100);
    for (int a = 0; a < strlen(s); a++)//将所有大写字母转换为小写，小写字母不变，方便计数
    {
        s[a] = tolower(s[a]);
    }
    for (int b = 0; b < 26; b++)
    {
        counts[b] = 0;
        counts_ip[b] = &counts[b];
    }
    count(s, counts_ip);
    for (int c = 0; c < 26; c++)
    {
        if (counts[c] > 1)
            cout << char(c + 97) << ":" << counts[c] << " times" << endl;
        else
        {
            if (counts[c] == 1)//英语语法, 1 time, n(>1) times
                cout << char(c + 97) << ":" << "1 time" << endl;
        }
    }
    return 0;
}

void count(const char s[], int* counts[26])
{
    for (int i = 0; i < strlen(s); i++)
    {
        if (97 <= int(s[i]) && int(s[i]) <= 122)
        {
            int letter = int(s[i]);
            *counts[letter-97] += 1;
        }
    }
}
```

(二) 指针

1.

```
#include <iostream>
#include <cstring>
using namespace std;
int indexof(const char* s1, const char* s2);
int main()
{
    char s1[100], s2[100];
    char* s1_ip = s1;
    char* s2_ip = s2;
    cout << "Enter the first string: ";
    cin.getline(s1, 100);
    cout << "Enter the second string: ";
    cin.getline(s2, 100);
    cout << "indexOf( " << s1 << " ", " << s2 << " ") is: " << indexof(s1_ip, s2_ip);
    return 0;
}

int indexof(const char* s1, const char* s2)
{
    int length1 = strlen(s1);
    int length2 = strlen(s2);
    if (length1 > length2)
        return -1; // 如果s1长度比s2还大, s1不可能是s2的子字符串
    else
    {
        for (int i = 0; i <= length2 - length1; i++)
        {
            int match = 0;
            for (int k = 0, j = i; k < length1; k++, j++)
            {
                if (s1[k] == s2[j])
                    match++;
            }
            if (match == length1) // 当s1和s2对应的字符相同时, match++, 当判断了length1个字符后, 如果match=length1, 那么s1即为s2的子字符串
                return i;
        }
        return -1;
    }
}
```

Microsoft Visual Studio 调试 × + ▾

Enter the first string: CSU
Enter the second string: Welcome to CSU!
indexOf("CSU", "Welcome to CSU!") is: 11
D:\Learning\C++\experiment\第二次实验\experiment4\4-2-1\x64\Debug\4-2-1.exe (进程 6308)已退出, 代码为 0。
按任意键关闭此窗口...

2.

```
#include <iostream>
#include <cstring>
#include <cmath>
int parseHex(const char* const hexString);
using namespace std;
int main()
{
    char hexString[100];
    char* hexString_ip = hexString;
    cout << "请输入一个十六进制数: ";
    cin.getline(hexString, 100);
    cout << "十进制为: " << parseHex(hexString_ip) << endl;
    return 0;
}

int parseHex(const char* const hexString)
{
    int length = strlen(hexString);
    int intString = 0;
    for (int i = 0; i < length; i++)
    {
        int number;
        if (65 <= int(hexString[i]) && int(hexString[i]) <= 70) //将A-F转化为10-15
        {
            number = int(hexString[i]) - 55;
        }
        else
        {
            number = int(hexString[i]) - 48; //1-9的ASCII值是48-57
            intString = intString + number * (pow(16, (length - i - 1)));
        }
    }
    return intString;
}
```

Microsoft Visual Studio 调试 × + -

请输入一个十六进制数: A5
十进制为: 165

D:\Learning\C++\experiment\第二次实验\experiment4\4-2-2\x64\Debug\4-2-2.exe (进程 16656)已退出, 代码为 0。
按任意键关闭此窗口...

3.

```
#include <iostream>
using namespace std;
void swap(double* ip, int a);
int main()
{
    int number;
    cout << "输入元素个数: ";
    cin >> number;
    double* shuzu = new double[number];
    cout << "输入数组元素: ";
    for (int i = 0; i < number; i++)
    {
        double a;
        cin >> a;
        shuzu[i] = a;
    }
    swap(shuzu, number);
    for (int j = 0; j < number; j++)
    {
        cout << shuzu[j] << " ";
    }
    delete[] shuzu;
    return 0;
}
void swap(double* ip, int a)//此函数用于给数组排序
{
    bool changed = true;
    do
    {
        changed = false;
        for (int k = 0; k < a - 1; k++)
        {
            if (ip[k] > ip[k + 1])
            {
                double c = ip[k];
                ip[k] = ip[k + 1];
                ip[k + 1] = c;
                changed = true;
            }
        }
    } while (changed);
}
```

Microsoft Visual Studio 调试 × + -

输入元素个数: 6
输入数组元素: 12.32 32.33 -56.79 159.6 66.66 78.79
-56.79 12.32 32.33 66.66 78.79 159.6
D:\Learning\C++\experiment\第二次实验\experiment4\4-2-3\x64\Debug\4-2-3.exe (进程 16992)已退出, 代码为 0。
按任意键关闭此窗口. . .

四、遇到的问题与解决方法

在好几个实验中，经常遇到编译器没有报错，说明没有语法问题，但输出结果不正确或者无法输出。通过不断尝试输出中间变量以及断点调试，最终发现往往是少了一个“=”或者其他符号的错误导致判断或循环出错。

五、体会

- (1) 指针的使用（结合数组、函数）非常容易混淆和出错，需要我多加练习和总结；
- (2) 在程序的编写过程中，极有可能会因为一个很小的错误（比如一个符号）导致整个程序得不到理想的输出，而这种错误往往很难发现，这需要我们锻炼思维的严谨性，养成耐心细心的习惯。