

实验名称 实验二：函数的应用与指针 实验室 学校机房 实验日期 2016. 10. 10

一、实验目的及要求

- 掌握函数声明、定义和使用的方法。
- 掌握引用的概念以及函数调用引用传值的机制。
- 掌握内联函数、重载函数、函数模版及默认函数参数的使用方法。
- 掌握堆内存操作的新和 delete。
- 理解 const 指针的表达及不同用法。
- 掌握指针作为参数，作为函数返回的用法，理解 void 指针的意义。
- 学会函数指针的使用。

二、实验环境

学校机房——DEV-C++实验环境。

三、实验内容

1、完成下面程序，熟悉函数调用的参数传递机制。

(1) 给定以下函数原型声明：

```
void sort1(int *a,int *b,int *c );
```

假设函数功能是把三个整数按升序排列，编程实现这个函数。

(2) 给定以下函数原型声明：

```
void sort2( int &a, int &b, int &c );
```

假设函数功能是把三个整数按降序排列，编程实现这个函数。

(3) 考虑如果用值传递能否完成上面功能。

2、编写函数，其原型为：void index(int a[], int n,int & sub)。

功能：在大小为 n 的数组 a 中，查找某个数 sub，若找到，将对应数组元素的下标赋给 sub，若没找到，将-1 赋给 sub，在主调函数中通过判断 sub 的值来判断数组中是否有该数。在这里，sub 是引用类型的参数，但起返回值的作用。

3、编写字符串查找函数 mystchr()。

该函数的功能是在字符串中查找指定字符，如果有该字符，则返回该字符在字符串中第一次出现的位置，否则返回 0，然后便协主函数进行验证。函数原型为：

```
int mystchr( char string[], char c );。
```

4、编写字符串反转函数 mystrev()。

该函数功能是将指定字符串中的字符顺序颠倒排列，然后再编写主函数进行

验证。函数原型为：

```
void mystrrrev( char string[]);。
```

5、分析程序运行结果，然后上机运行，掌握全局变量、局部变量、静态变量的作用。

```
1. #include <iostream>
2. using namespace std;
3. int n = 0;
4. int func(int x = 10);
5. int main()
6. {
7.     int a,b;
8.     a = 5;
9.     b = func(a);
10.    cout << "\nlocal a=" << a<< endl << "local b=" << b<<endl << "global n="
    << n<<endl;
11.    a++;
12.    b = func(a);
13.    cout << "\nlocal a=" << a<< endl << "local b=" << b<<endl << "global n="
    << n<<endl;
14.    func();
15.    return 0;
16. }
17.
18. int func(int x)
19. {
20.     int a=1;
21.     static int b=10;
22.     a++;
23.     b++;
24.     x++;
25.     n++;
26.    cout << "\nlocal a=" << a<< endl << "local b=" << b<<endl << "parameter x="
    << x << endl;
27.    return a+b;
28. }
```

6、掌握递归函数的编写方法。

编写一个函数，求从 n 个不同的数中取 r 个数的所有选择的个数。其个数值

$$C_n^r = \frac{n!}{r! * (n-r)!}$$

为：

其中： $n! = n * (n-1) * (n-2) * \dots * 1$ 。要求：分别用递归和非递归两种方式完成程序设计；主程序中设计一个循环，不断从输入接收 n 和 r 的值，计算结果

并输出，当用户输入 0 0 时，程序结束；能检查输入数据的合法性，要求 $n \geq 1$ 并且 $n \geq r$ 。

提示：(1) 利用一个非递归函数 $fn(int\ n)$ 计算 $n!$ ，利用另一个函数 $cnr(int\ n, int\ r)$ 计算 C_n^r ，在该函数中调用 $fn()$ 。问题：你打算用什么样的变量类型来存放 $n!$ 函数返回的值？注意各种数据类型的内存字长不同，整数能存放的数据范围有限，你如何解决？(2) 利用一个递归函数实现，实现时利用公式： $C(n, r) = C(n, r-1) * (n - r + 1) / r$ 递归实现。如果 $r = 0$ ，则 $C(n, r) = 1$ ，如果 $r = 1$ ，则 $C(n, r) = n$ 。

7、将上面的程序改成多文件结构

将上面用非递归方式写成的程序改成用多文件结构表示。要求将 $main()$ 函数放在一个文件中，将另外两个函数放在另一个文件中，将函数原型说明放在一个头文件中。建立一个项目，将这三个文件加到你的项目中，编译连接使你的程序正常运行。

思考问题：多文件结构中头文件的作用是什么？将程序划分为多个文件有什么好处？

8、阅读下列四段程序，每段程序都有不合理的地方，分析什么地方不合理，解释原因。

程序一：

```
1. #include <iostream>
2. using namespace std;
3. void GetMemory(char *p)
4. {
5.     p = new char[100];
6. }
7. int main(void)
8. {
9.     char *str = NULL;
10.    GetMemory(str);
11.    strcpy(str, "hello world");
12.    cout <<str;
13.    return 0;
14. }
```

程序二：

```
1. #include <iostream>
2. using namespace std;
3. void GetMemory(char **p)
4. {
5.     *p = new char[100];
6. }
7. int main()
8. {
9.     char *str = NULL;
```

```

10.    GetMemory(&str);
11.    strcpy(str, "hello world");
12.    cout <<str;
13.    return 0;
14. }

```

程序三：

```

1. #include <iostream>
2. using namespace std;
3. char* GetMemory()
4. {
5.     char p[5] = {'a', 'b', 'c', 'd', '\0'};
6.     return p;
7. }
8. int main(void)
9. {
10.    char *str = NULL;
11.    str = GetMemory();
12.    cout <<str;
13.    return 0;
14. }

```

程序四：

```

1. #include <iostream>
2. using namespace std;
3. int main(void)
4. {
5.     char *str = new char[100];
6.     strcpy(str, "hello");
7.     delete[] str;
8.     if(str != NULL)
9.     {
10.        strcpy(str, "world");
11.        cout <<str;
12.    }
13.    return 0;
14. }

```

9、阅读下列程序，指出错误的语句和出错的原因，然后上机调试并改正错误。

```

1. int main()
2. {

```

```

3.     int a = 15, b = 20;
4.     const int c = 105;
5.     const int *p1 = &c;
6.     int *const p2 = &b;
7.     const int *const p3 = &c;
8.     p1 = &b;
9.     *p1 = 100;
10.    p2 = &a;
11.    *p2 = a;
12.    p3 = &b;
13.    *p3 = 100;
14.    return 0;
15. }

```

10、运行下面程序，考虑内存泄露的危害性。

```

1. int main()
2. {
3.     int s = 999999999;
4.     while(1)
5.     {
6.         new int[s];
7.     }
8.     return 0;
9. }

```

程序运行时你可以打开任务管理器，来查看系统的内存使用变化情况。

11、编一程序，将字符串“Hello,C++!”赋给一个字符数组，然后从第一个字母开始间隔地输出该串（请用指针完成）。

12、编写一个函数，用于去掉字符串尾部的空格符。

函数原型为：char *mytrim(char *string);其中参数 string 为字符串，返回值为指向 string 的指针。

13、编写一个函数，用于去掉字符串前面的空格。

函数原型为：char *myltrim(char *string);其中参数 string 为字符串，返回值为指向 string 的指针。

14、编写程序

编写一程序（应该有多个函数），允许从键盘输入任意多个英语单词（单词可以重复），中间用空格分开，输入 0 表示输入结束。该程序可以统计同一个英语单词被输入几次，最后对英文单词按字典顺序输出，后面跟上该单词被输入的次数。（提示，尝试用结构体组织数据，把单词和该单出现的次数用一个结构体来描述。）

比如输入：book am book is book am 0

输出：am 2

book 3

is 1

四、实验步骤

1. 从主菜单中选择“文件”“源程序”；
2. 在屏幕新的空白文档编辑区中输入以下内容；
3. 依次编译、运行、执行。

五、实验结果及总结

这次 C++ 的实验，反映出了自己对指针和调用函数的基础知识不足，遇到的问题没有很好的逻辑思维。通过自己和同学的交流和讨论后，试编和改错，最后能够顺利的编写出来，带来的也有成就感，并增加了我对 C++ 的兴趣和爱好。而且实验还让我理解了在 C++ 程序中的输入输出和字符串的统计，还了解到了一点递归的方法。希望自己继续努力把基础打好，一步一步理解。

六、源程序清单（每题包括 3 部分：题目+程序+结果截图）

- 1、完成下面程序，熟悉函数调用的参数传递机制。

（1）给定以下函数原型声明：

```
void sort1( int *a, int *b, int *c );
```

假设函数功能是把三个整数按升序排列，编程实现这个函数。

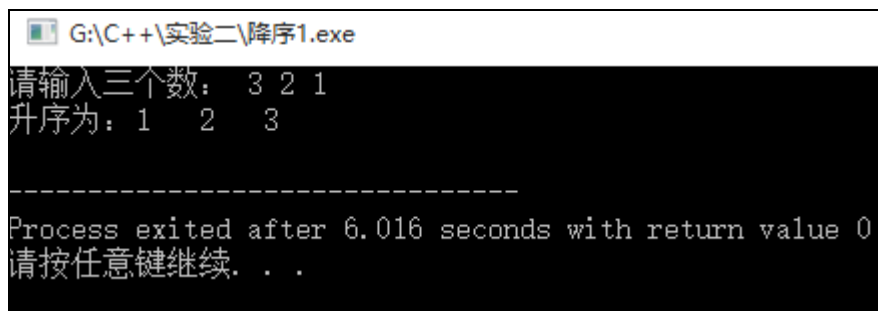
```
1. #include<iostream>
2. using namespace std;
3. int main()
4. {
5.     void sort1(int *a,int *b,int *c);
6.     int x1,x2,x3,*t1,*t2,*t3;
7.     cout<<"请输入三个数: ";
8.     cin>>x1>>x2>>x3;
9.     t1=&x1; t2=&x2; t3=&x3;
10.    sort1(t1,t2,t3);
11.    cout<<"升序为: ";
12.    cout<<x1<<" "<<x2<<" "<<x3<<endl;
13.    return 0;
14. }
15.
16. void sort1(int *a,int *b,int *c)
17. {
18.     void swap(int *p1,int *p2);
19.     if(*a>*b)    swap(a,b);
20.     if(*a>*c)    swap(a,c);
21.     if(*b>*c)    swap(b,c);
```

```

22. }
23. void swap(int *p1,int *p2)
24. {
25.     int temp;
26.     temp=*p1;
27.     *p1=*p2;
28.     *p2=temp;
29. }

```

假设函数功能是把三个整数按升序排列，编程实现这个函数。



```

G:\C++\实验二\降序1.exe
请输入三个数: 3 2 1
升序为: 1 2 3

-----
Process exited after 6.016 seconds with return value 0
请按任意键继续. . .

```

(2) 给定以下函数原型声明:

```
void sort2( int &a, int &b, int &c );
```

假设函数功能是把三个整数按降序排列，编程实现这个函数。

```

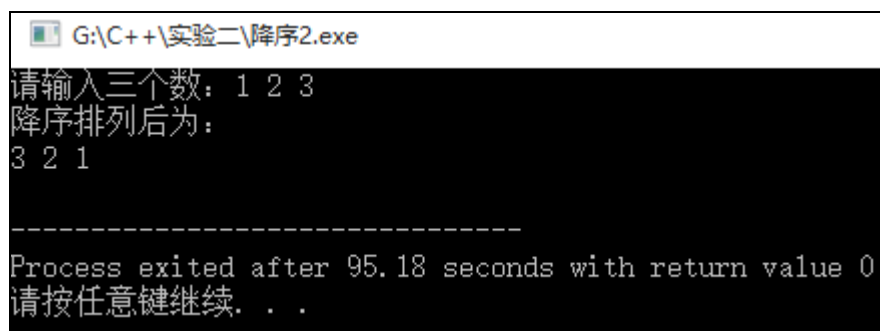
1. #include<iostream>
2.
3. using namespace std;
4.
5. void sort2(int &a,int &b,int &c)
6. {
7.     if(a>b||a==b)
8.         if(a>c||a==c)
9.             if(b>c||b==c)
10.                 cout<<a<<" "<<b<<" "<<c<<endl;
11.         else
12.             cout<<a<<" "<<c<<" "<<b<<endl;
13.     else
14.         cout<<c<<" "<<a<<" "<<b<<endl;
15.     else
16.         if(b>c||b==c)
17.             if(a>c||a==c)
18.                 cout<<b<<" "<<a<<" "<<c<<endl;
19.         else
20.             cout<<b<<" "<<c<<" "<<a<<endl;
21.     else
22.         cout<<c<<" "<<b<<" "<<a<<endl;

```

```

23. }
24.
25. int main()
26. {
27.     int a,b,c;
28.
29.     cout<<"请输入三个数: ";
30.     cin>>a>>b>>c;
31.     cout<<"降序排列后为: "<<endl;
32.
33.     sort2(a,b,c);
34. }

```



```

G:\C++\实验二\降序2.exe
请输入三个数: 1 2 3
降序排列后为:
3 2 1

-----
Process exited after 95.18 seconds with return value 0
请按任意键继续. . .

```

(3) 考虑如果用值传递能否完成上面功能。

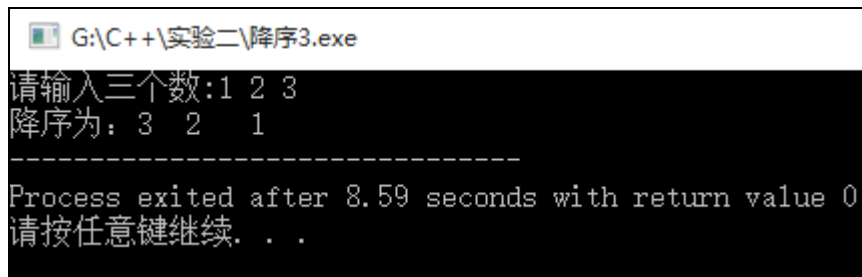
```

1. #include<iostream>
2. using namespace std;
3. int main()
4. {
5.     int a,b,c,t;
6.     cout<<"请输入三个数:";
7.     cin>>a>>b>>c;
8.     if(a<b&&b<c)    //a<b<c
9.     {
10.         t=a;a=c;c=t;
11.     }
12.     if(c<a&&a<b)    //c<a<b
13.     {
14.         t=a;a=b;b=t;
15.     }
16.     if(b<c&&c<a)    //b<c<a
17.     {
18.         t=b;b=c;c=t;
19.     }
20.     cout<<"降序为: ";
21.     cout<<a<<" "<<b<<" "<<c;

```



```
22. }
```



```
G:\C++\实验二\降序3.exe
请输入三个数:1 2 3
降序为: 3 2 1
-----
Process exited after 8.59 seconds with return value 0
请按任意键继续. . .
```

2、编写函数，其原型为：void index(int a[], int n,int & sub)。

功能：在大小为n的数组a中，查找某个数sub，若找到，将对应数组元素的下标赋给sub，若没找到，将-1赋给sub，在主调函数中通过判断sub的值来判断数组中是否有该数。在这里，sub是引用类型的参数，但起返回值的作用。

```
#include<iostream>

using namespace std;

void index(int a[],int n,int &sub);

int main()
{
    int sub,n;

    cout<<"输入数组的长度:";
    cin>>n;
    int a[n];
    cout<<"输入数组:";

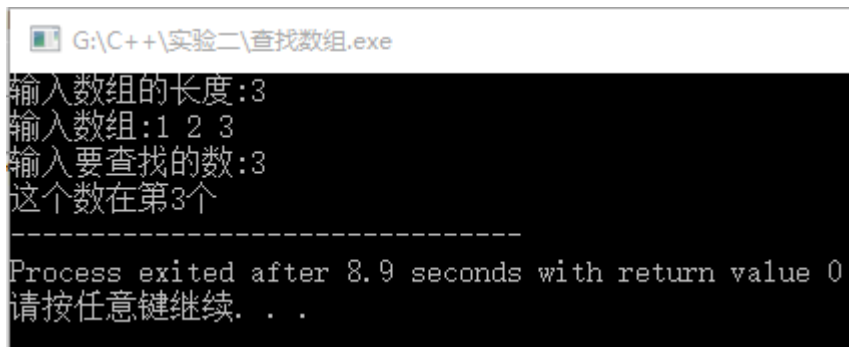
    for(int i=0;i<n;i++)
        cin>>a[i];
    cout<<"输入要查找的数:";
    cin>>sub;
    void index(int a[],int n,int &sub);
    if(sub!=-1)
        cout<<"这个数在第"<<sub<<"个";
    else
        cout<<"数组中没有这个数.";
```

```

}

void index(int a[], int n, int &sub)
{
    for(int i=0; i<n; i++)
    {
        if(a[i]==sub)
            sub=i+1;
        else
            sub=-1;
    }
}

```



```

G:\C++\实验二\查找数组.exe
输入数组的长度:3
输入数组:1 2 3
输入要查找的数:3
这个数在第3个
-----
Process exited after 8.9 seconds with return value 0
请按任意键继续. . .

```

3、编写字符串查找函数 mystrchr()。

该函数的功能是在字符串中查找指定字符，如果有该字符，则返回该字符在字符串中第一次出现的位置，否则返回 0，然后便协主函数进行验证。函数原型为：

```
int mystrchr( char string[], char c );。
```

```

#include<iostream>

#include<string.h>

using namespace std;

int mystrchar(char string[], char x)
{
    int len=strlen(string);
    int i;

```

```

for(i=0;i<len,string[i]!='\0';i++)
{
    if(string[i]==x)
        return i+1;
    else
        continue;
}
}

int main()
{
    int a=0;
    char str[100],y;
    cout<<"输入字符串: ";
    cin>>str;
    cout<<"输入要查找的字符: ";
    cin>>y;
    a=mystrchar(str,y);
    if(a==0)
        cout<<"字符串中不存在这个字符";
    else
        cout<<"这个字符在字符串的第"<<a<<"个位置";
    return 0;
}

```

 G:\C++\实验二\查找字符串.exe

```

输入字符串: sadhfescioeh
输入要查找的字符: s
这个字符在字符串的第1个位置
-----
Process exited after 11.18 seconds with return value 0
请按任意键继续. . .

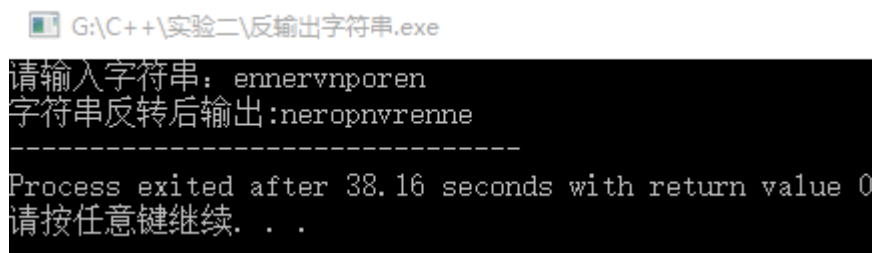
```

4、编写字符串反转函数 mystrrev()。

该函数功能是将指定字符串中的字符顺序颠倒排列，然后再编写主函数进行验证。函数原型为：

void mystrrrev(char string[]);。

```
#include<iostream>
#include<cstring>
using namespace std;
int mystrrrev(char string[])
{
    int t,i;
    int len=strlen(string);
    for(i=0;i<len/2;i++)
    {
        t=string[i];
        string[i]=string[len-i-1];
        string[len-1-i]=t;
    }
}
int main()
{
    char str[50];
    cout<<"请输入字符串：";
    cin>>str;
    mystrrrev(str);
    cout<<"字符串反转后输出："<<str;
}
```



```
G:\C++\实验二\反输出字符串.exe
请输入字符串：ennervnporen
字符串反转后输出：neropnvrenne
-----
Process exited after 38.16 seconds with return value 0
请按任意键继续...
```

5、分析程序运行结果，然后上机运行，掌握全局变量、局部变量、静态变量的

作用。

```
#include <iostream>
using namespace std;
int n = 0;
int func(int x = 10);
int main()
{
    int a,b;
    a = 5;
    b = func(a);
    cout << "\nlocal a=" << a<< endl << "local b=" << b<<endl <<
"global n=" << n<<endl;
    a++;
    b = func(a);
    cout << "\nlocal a=" << a<< endl << "local b=" << b<<endl <<
"global n=" << n<<endl;
    func();
    return 0;
}
```

```
int func(int x )
{
    int a=1;
    static int b=10;
    a++;
    b++;
    x++;
    n++;
    cout << "\nlocal a=" << a<< endl << "local b=" << b<<endl << "parameter
x=" << x << endl;
    return a+b;
}
```

全局变量具有全局作用域。全局变量只需在一个源文件中定义，就可以作用于所有的源文件。静态局部变量具有局部作用域，它只被初始化一次，自从第一次被初始化直到程序运行结束都一直存在，它和全局变量的区别在于全局变量对所有的函数都是可见的，而静态局部变量只对定义自己的函数体始终可见。局部变量也只有局部作用域，它是自动对象（auto），它在程序运行期间不是一直存在，而是只在函数执行期间存在，函数的一次调用执行结束后，变量被撤销，其所占用的内存也被收回。静态全局变量也具有全局作用域，它与全局变量的区别在于如果程序包含多个文件的话，它作用于定义它的文件里，不能作用到其它文件里，即被 static 关键字修饰过的变量具有文件作用域。

6、掌握递归函数的编写方法。

编写一个函数，求从 n 个不同的数中取 r 个数的所有选择的个数。其个数值为：

$$C_n^r = \frac{n!}{r! * (n-r)!}$$

其中： $n! = n * (n-1) * (n-2) * \dots * 1$ 。要求：分别用递归和非递归两种方式完成程序设计；主程序中设计一个循环，不断从输入接收 n 和 r 的值，计算结果并输出，当用户输入 0 0 时，程序结束；能检查输入数据的合法性，要求 $n \geq 1$ 并且 $n \geq r$ 。

提示：（1）利用一个非递归函数 $fn(int\ n)$ 计算 $n!$ ，利用另一个函数 $cnr(int\ n, int\ r)$ 计算 C_n^r ，在该函数中调用 $fn()$ 。问题：你打算用什么样的变量类型来存放 $n!$ 函数返回的值？注意各种数据类型的内存字长不同，整数能存放的数据范围有限，你如何解决？（2）利用一个递归函数实现，实现时利用公式： $C(n, r) = C(n, r-1) * (n - r + 1) / r$ 递归实现。如果 $r = 0$ ，则 $C(n, r) = 1$ ，如果 $r = 1$ ，则 $C(n, r) = n$ 。

```
#include<iostream>

using namespace std;

int fn(int n)
{
    if(n==1 || n==0)
        return(1);
    else
        return(n*fn(n-1));
}

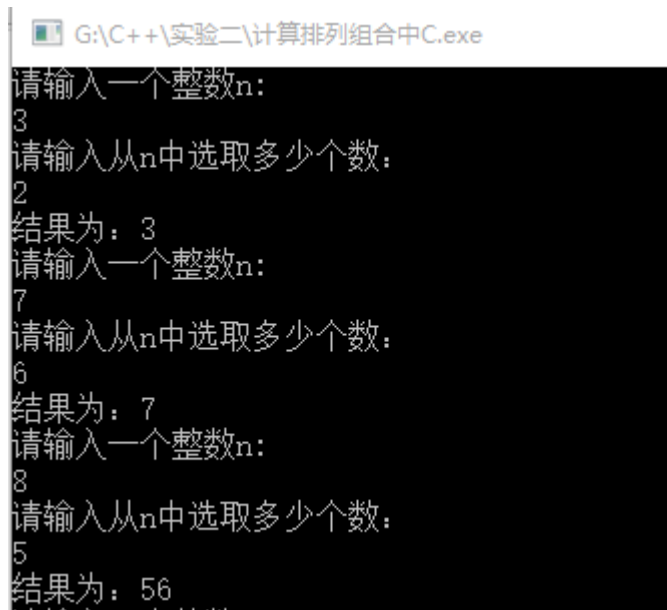
int main()
{
    int n,r,a,b,c;
    while(1)
    {
        cout<<"请输入一个整数 n:"<<endl;
        cin>>n;

        cout<<"请输入从 n 中选取多少个数："<<endl;
        cin>>r;
```

```

while(1)
{
    if(n==0&&r==0)
        return 0;
    else
        a=fn(n);
        b=fn(r);
        c=fn(n-r);
        cout<<"结果为: "<<a/(b*c)<<endl;
        break;
}
}
}

```



```

G:\C++\实验二\计算排列组合中C.exe
请输入一个整数n:
3
请输入从n中选取多少个数:
2
结果为: 3
请输入一个整数n:
7
请输入从n中选取多少个数:
6
结果为: 7
请输入一个整数n:
8
请输入从n中选取多少个数:
5
结果为: 56

```

7、将上面的程序改成多文件结构

将上面用非递归方式写成的程序改成用多文件结构表示。要求将 `main()` 函数放在一个文件中，将另外两个函数放在另一个文件中，将函数原型说明放在一个头文件中。建立一个项目，将这三个文件加到你的项目中，编译连接使你的程序正常运行。

思考问题：多文件结构中头文件的作用是什么？将程序划分为多个文件有什么好处？

头文件中包含一些必要的函数，是软件自带的函数库，比如输入输出流函数所以涉及到相关函数都需要加上头文件，否则会产生错误。

8、阅读下列四段程序，每段程序都有不合理的地方，分析什么地方不合理，解释原因。

程序一：

```
#include <iostream>
using namespace std;
void GetMemory(char *p)
{
    p = new char[100];
}
int main(void)
{
    char *str = NULL;
    GetMemory(str);
    strcpy(str, "hello world");
    cout << str;
    return 0;
}
```

程序二：

```
#include <iostream>
using namespace std;
void GetMemory(char **p)
{
    *p = new char[100];
}
int main()
{
    char *str = NULL;
    GetMemory(&str);
    strcpy(str, "hello world");
    cout << str;
    return 0;
}
```

程序三：

```
#include <iostream>
using namespace std;
char* GetMemory()
{
    char p[5] = {'a', 'b', 'c', 'd', '\0'};
    return p;
}
int main(void)
{
}
```



```

        char *str = NULL;
        str = GetMemory();
        cout << str;
        return 0;
    }

```

程序四：

```

#include <iostream>
using namespace std;
int main(void)
{
    char *str = new char[100];
    strcpy(str, "hello");
    delete[] str;
    if(str != NULL)
    {
        strcpy(str, "world");
        cout << str;
    }
    return 0;
}

```

1. *p 和 str 指向同一个地址空间，这一地址空间地址为 NULL 也就是说是不存在的，接下来语句 p = (char *)malloc(num) 使得 p 重新指向另一个内存地址。注意，此时 *str 并未作任何改变。跳出函数后，执行 cout << str << endl 语句，str 未指向任何地址空间，要输出地址空间中的值当然会出错了

2. *p 指向 *str 的地址空间（注意，*str 这个变量在声明它的时候系统就为它在栈中分配了内存，这里不要将指针变量的地址与指针指向的地址搞混了！），语句 *p = (char *)malloc(num) 为 p 指向的 *str 重新指向一个地址大小为 num 的 char 型字符串，跳出函数后，打印 str，结果就是“hello”了。

9、阅读下列程序，指出错误的语句和出错的原因，然后上机调试并改正错误。

```

int main()
{
    int a = 15, b = 20;
    const int c = 105;
    const int *p1 = &c;
    int *const p2 = &b;
    const int *const p3 = &c;
    p1 = &b;
    *p1 = 100;
    p2 = &a;
    *p2 = a;
    p3 = &b;
}

```

```
        *p3 = 100;
        return 0;
    }
```

```
const int c=105;
```

//c 为常变量，即 c 的值在这里被固定为 105，之后的语句不许改变了

*p1 不能再赋值

p2 不能再赋值

p3 和*p3 都不能再赋值

10、运行下面程序，考虑内存泄露的危害性。

```
int main()
{
    int s = 99999999;
    while(1)
    {
        new int[s];
    }
    return 0;
}
```


返回的是一个指向 int 的指针 int* 要访问 new 所开辟的结构体空间，无法直接通过变量名进行，只能通过赋值的指针进行访问。用 new 和 delete 可以动态开辟，撤销地址空间。在编程时，若用完一个变量(一般是暂时存储的数组)，下次需要再用，但却又想省去重新初始化的功夫，可以在每次开始使用时开辟一个空间，在用完后撤销它。

11、编一程序，将字符串“Hello,C++!”赋给一个字符数组，然后从第一个字母开始间隔地输出该串（请用指针完成）。

```
#include<iostream>
#include<cstring>
using namespace std;
int main()
{
    char string[]={"hello,c++"};

    for(int i=0;i<9;i++)
    {
        cout<<string[i];
        cout<<" ";
    }
}
```

```
}  
}
```

 G:\C++\实验二\输出hello,c++.exe

```
hello, c++  
-----  
Process exited after 0.2481 seconds with return value 0  
请按任意键继续. . .
```

12、编写一个函数，用于去掉字符串尾部的空格符。

函数原型为：char *mytrim(char *string);其中参数 string 为字符串，返回值为指向 string 的指针。

```
#include<iostream>
```

```
#include<cstring>
```

```
using namespace std;
```

```
char *mylltrim(char *string)
```

```
{
```

```
    for(int i=strlen(string);string[i]!='\0';i--)
```

```
    {
```

```
        if(string[i]==' ')
```

```
            string[i]='\0';
```

```
        else
```

```
            break;
```

```
    }
```

```
}
```

```
int main()
```

```
{
```

```
    char str[50];
```


```
    cout<<"输入字符串:";
```

```
    cin>>str;
```

```
    mylltrim(str);
```

```
    cout<<"去掉字符串后的空格:"<<str<<".";
```

```
}
```

 G:\C++\实验二\去掉空格的字符串.exe

```
输入字符串:  jcnoliahefejf
去掉空格后:jcnoliahefejf.
-----
Process exited after 7.897 seconds with return value 0
请按任意键继续. . .
```

13、编写一个函数，用于去掉字符串前面的空格。

函数原型为：char *myltrim(char *string);其中参数 string 为字符串，返回值为指向 string 的指针。

```
#include<iostream>

#include<cstring>

using namespace std;

char *myltrim(char *string)
{ int k=0;
  for(int i=0;string[i]!=' ';i++)
  {
    for(int j=i;i<strlen(string);j++)
      string[j]=string[j+1];
  }
}

int main()
{char str[50];
  cout<<"输入字符串:";
  cin>>str;
  myltrim(str);
  cout<<"  去  掉  字  符  串  前  的  空  格  为  :"<<str;
jksndcowreoifj
}
```

 G:\C++\实验二\去掉字符串前的空格.exe

```
输入字符串:      snenoirgpreq
去掉字符串前的空格为:snenoirgpreq
-----
Process exited after 21.28 seconds with return value 0
请按任意键继续. . .
```

14、编写程序

编写一程序（应该有多个函数），允许从键盘输入任意多个英语单词（单词可以重复），中间用空格分开，输入 0 表示输入结束。该程序可以统计同一个英语单词被输入几次，最后对英文单词按字典顺序输出，后面跟上该单词被输入的次数。（提示，尝试用结构体组织数据，把单词和该单出现的次数用一个结构体来描述。）

比如输入： book am book is book am 0

输出： am 2

book 3

is 1

```
#include<iostream>
```

```
#include<string.h>
```

```
using namespace std;
```

```
void sort_word(string word[100],int count);
```

```
void deal_word(string word[100],int count);
```

```
int main()
```

```
{
```

```
int i,count=0;
```

```
string word[100];
```

```
cout<<"输入任意多个英语单词(单词可以重复),中间用空格分开,输入 0 表示  
输入结束"<<endl;
```

```
for(i=0;i<100;i++)
```

```
{
```

```
cin>>word[i];
```

```
if(word[i]=="0")
```

```
break;
```

```
count++;
```

```
}
```

```

    sort_word(word, count);
    deal_word(word, count);

    return 0;
}

void sort_word(string word[100], int count)
{
    int i, j, min=0;
    string temp;
    for(i=0; i<count; i++)
    {
        min=i;
        for(j=i; j<count; j++)
            if(word[j]<word[min])
            {
                temp=word[j];
                word[j]=word[min];
                word[min]=temp;
            }
    }
}

void deal_word(string word[100], int count)
{

```

```


    int i, j=0, k=-1;
    int num[20];
    for(i=0; i<count; i++)
        num[i]=0;
    i=0;

```

```

for(;i<count;i++)
{
    i=k+1;
    for(j=i;j<count;j++)
    {
        if(word[i]==word[j])
        {
            k=j;
            num[i]=num[i]+1;
        }
    }
    cout<<word[i]<<" "<<num[i]<<endl;
}
}

```



```

G:\C++\实验二\重复字符.exe
输入任意多个英语单词(单词可以重复), 中间用空格分开, 输入0表示输入结束
i h a v e b o o k 0
a 1
b 1
e 1
h 1
i 1
k 1
o 2
v 1

-----
Process exited after 19.1 seconds with return value 0
请按任意键继续. . .

```