**计算机程序设计基础（C++)**

**实验报告**

专业班级： 软件工程2402

学 号： 8209240203

姓 名： 丁晨阳

**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。

1. 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？

答：**函数类型决定了返回值的类型，在定义函数时，根据需要的返回值类型，给函数定义相应类型。**

2. 本实验中主函数调用函数时采用的是何种传递方式？

答：**在本实验1中调用函数采用的是引用传递 2、4、5中采用的是值传递。**

**四、算法分析，程序结果**

1. #include <iostream>

using namespace std;

void gcd(int &a, int &b, int &result) {

    int x = a, y = b; // 保留原始值

    while (y != 0) {

        int temp = y;

        y = x % y;

        x = temp;

    }

    result = x; // 将结果通过引用返回

}

int main() {

    int m, n, result;

    // 输入自然数 m 和 n

    cout << "请输入两个自然数 m 和 n: ";

    cin >> m >> n;

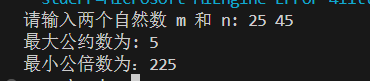
    // 调用函数求最大公约数

    gcd(m, n, result);

    // 输出结果

    cout << "最大公约数为: " << result << endl;

    cout<<"最小公倍数为："<< (m\*n)/result<< endl;

****    return 0;

}

**2**.#include<iostream>

#include<math.h>

using namespace std;

bool isPrime(int n)

{

    if(n==1) return false;

    for(int i = 2;i<=sqrt(n);i++)

    {

        if(n%i==0)

        {

            return false;

        }

    }

    return true;

}

int main()

{

    int cnt=0;

    for(int i=2; ; i++)

    {

        if(cnt==200)

        break;

        if(isPrime(i))

        {

            cnt++;

            cout<<i<<" ";

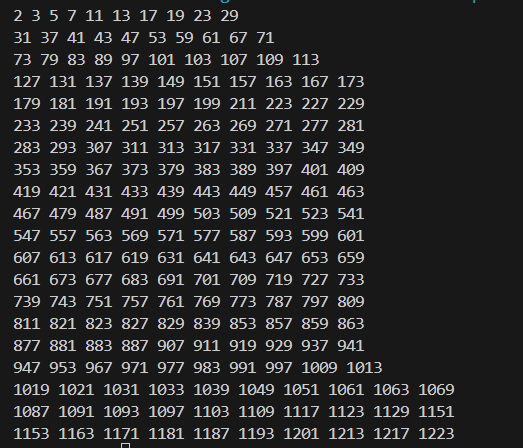
            if(cnt%10==0)

            cout<<endl;

        }

    }

    return 0;

****}

3.

4. **mytriangle.h:**

#include<iostream>

using namespace std;

bool is\_valid(double side1, double side2, double side3);//定义判断三角形是否有效的函数

double area(double side1, double side2, double side3);//定义计算面积函数

**mytriangle.cpp:**

#include"mytriangle.h"

bool is\_valid(double side1, double side2, double side3)//判断三角形是否有效

{

if (side1 + side2 > side3 && side1 + side3 > side2 && side2 + side3 > side1)//两边之和大于第三边

{

return true;//返回true

}

else

{

return false;//返回false

}

}

double area(double side1, double side2, double side3)//计算面积

{

double s = (side1 + side2 + side3) / 2;//面积公式

double square = sqrt(s \* (s - side1) \* (s - side2) \* (s - side3));//面积公式

return square;//返回面积大小

}

**test3.4.cpp:**

#include"mytriangle.h"

int main()

{

int side1, side2, side3;//定义三边长

cout << "请分别输入三边边长" << endl;//提醒输入三边长

cin >> side1 >> side2 >> side3;//输入三边长

int val = is\_valid(side1, side2, side3);//调用判断三角形是否有效的函数

if (val == true)//如果有效

{

double square = area(side1, side2, side3);//调用计算面积函数

cout << "您定义的三角形面积为:" << square << endl;//输出面积

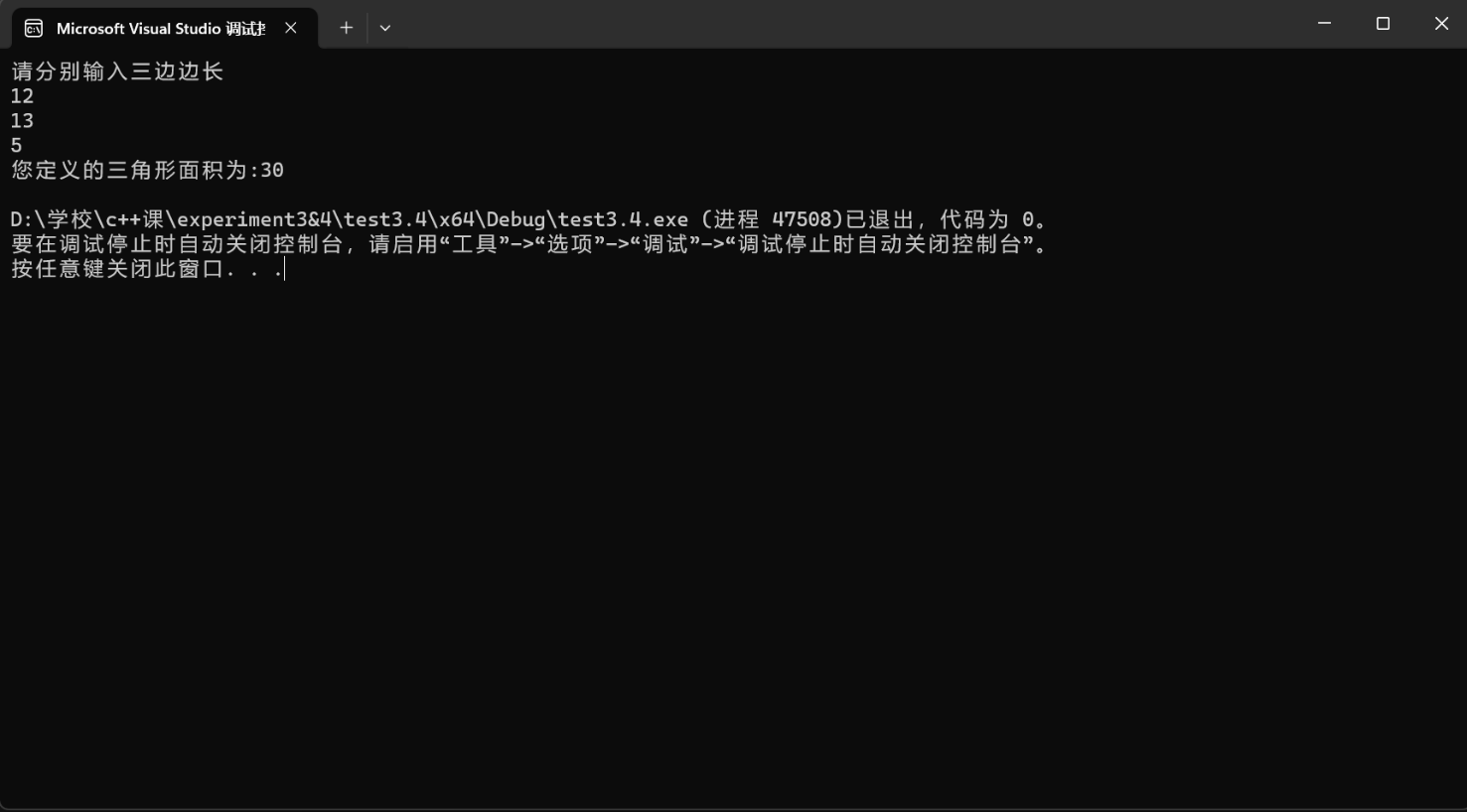
}

else//如果无效

{

cout << "您定义的三边长不合法" << endl;//提示无效

}

**** return 0;

}

5. #include<iostream>

using namespace std;

int dfs\_monkey(int day)

{

   if(day==10) return 1;

   return (dfs\_monkey(day+1)+1)\*2;

}

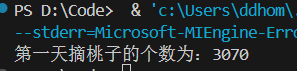
int main()

{

    cout<<"第一天摘桃子的个数为："<<dfs\_monkey(0)<<endl;

    return 0;

}



**五、遇到的问题与解决方法**

在使用自定义头文件mytriangle.h时，写语句为#include<mytriangle.h>一直出现报错无法解决，经过查阅资料后发现，调用头文件时，如果使用<>，则编译器会去指定的文件夹查找对应的头文件进行加载；而使用“ ”时，编译器会优先从当前项目的文件夹查找头文件，因此，将原语句改为 #include “mytriangle.h” 后，程序顺利编译运行。

**六、体会**

函数是简化代码和理清逻辑的重要工具，尤其在大型项目中尤为关键。通过分文件编写，将函数在头文件中声明，功能划分到不同源文件，主函数中调用，可以提升代码的可读性和逻辑性。使用函数时需注意细节，例如需要修改参数值时应使用引用或地址传递，而非值传递。此外，合理运用`const`防止意外修改参数，结合函数重载等技巧，可以进一步减少代码冗余，提升程序质量。

循环与判断语句是程序易出错的环节。在编写打印前200个素数的程序时，我曾因换行条件处理不当，导致多余空行问题。这提醒我，在编写逻辑时必须细致严谨，快速发现和解决问题是提高效率的关键。通过加强逻辑梳理和注重细节，编程效率和代码质量都能显著提升。

**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

Enter ten numbers: 1 2 3 2 1 6 3 4 5 2

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个存物柜，100个学生。开学第一天所有存物柜都是关闭的。第一个学生（记为S1）来到学校后，打开所有的存物柜。第二个学生S2，从第二个存物柜（记为L2）开始，每隔两个存物柜，将它们关闭。第三个学生S3从第三个存物柜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，那些存物柜是开着的？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，每个元素代表存物柜是开（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

Enter

Enter list1: 5 1 5 16 61 111

Enter list1: 4 2 4 5 6

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

Enter

Enter

Enter the first string: welcome

Enter the second string: We welcome you!

indexOf(“welcome”, “We welcome you!”) is 3

Enter

Enter the first string: welcome

Enter the second string: We invite you!

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;

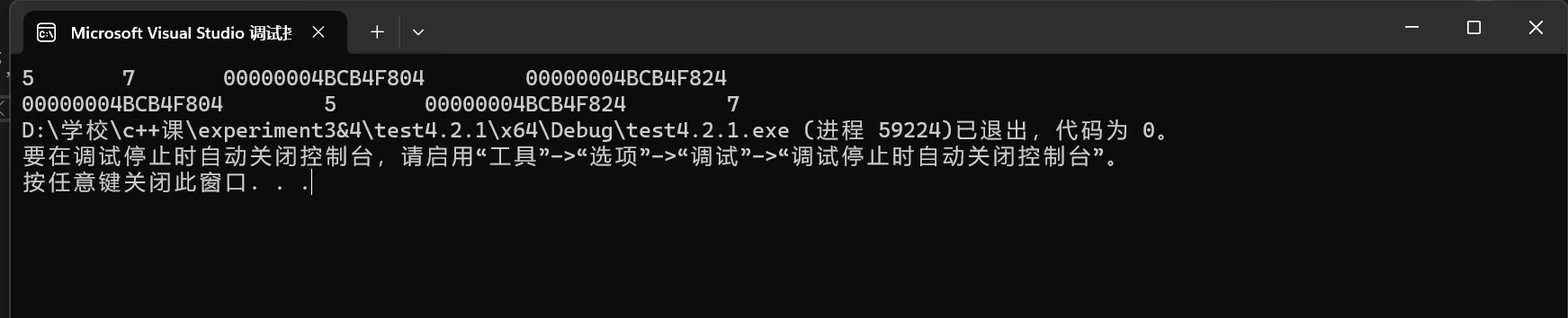
pj=&j;

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<iostream>

using namespace std; //#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;

}

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

1. 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**

**1.** #include<iostream>

#include<vector>

#include <set>

using namespace std;

int main()

{

    vector<int> numbers;

    set<int> set;

    for(int i=0;i<10;i++)

    {

        int x;

        cin>>x;

        set.insert(x);

    }

    for(auto it=set.begin();it!=set.end();it++)

    {

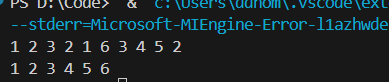
        numbers.push\_back(\*it);

    }

    for(int i:numbers)

    cout<<i<<" ";

    return 0;

****}

2. #include<iostream>

#include<vector>

using namespace std;

void bubbleSort(vector<int>&a)

{

    for(int i=0;i<a.size();i++)

    {

        for(int j=0;j<a.size()-1-i;j++)

        {

            if(a[j]>a[j+1])

            {

                int temp=a[j];

                a[j]=a[j+1];

                a[j+1]=temp;

            }

        }

    }

}

int main()

{

    vector<int> a(10);

    cout<<"排序前的数组为：";

    for(int i=0;i<10;i++)

    cin>>a[i];

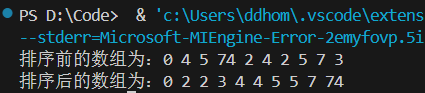
    bubbleSort(a);

    cout<<"排序后的数组为：";

    for(int i:a)

    cout<<i<<" ";

    return 0;

****}

3. #include <iostream>

#include<vector>

using namespace std;

int main()

{

    vector<bool> condition(100,true);

    for(int i=1;i<100;i++)

    {

        for(int j=i;j<100;j+=i+2)

        {

            condition[j]=!condition[j];

        }

    }

    cout<<"开着的柜子为："<<endl;

    for(int i=0;i<condition.size();i++)

    {

        if(condition[i])

        cout<<i+1<<" ";

    }

    return 0;

}

****

4. #include<iostream>

#include<vector>

#include<algorithm>

using namespace std;

int main()

{

    int len1, len2;

    cout<<"输入数组1的元素个数和元素：";

    cin>>len1;

    vector<int> list1(len1);

     for(int i=0;i<len1;i++)

    {

        cin>>list1[i];

    }

    cout<<"输入数组2的元素个数和元素：";

    cin>>len2;

    vector<int> list2(len2);

    for(int i=0;i<len2;i++)

    {

        cin>>list2[i];

    }

    vector<int> list;

    for(int i=0;i<list1.size();i++)

    list.push\_back(list1[i]);

    for(int i=0;i<list2.size();i++)

    list.push\_back(list2[i]);

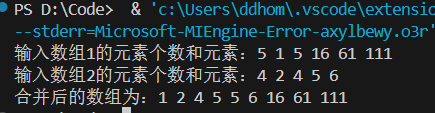
    sort(list.begin(), list.end());

    cout<<"合并后的数组为：";

    for(int i=0;i<list.size();i++)

    cout<<list[i]<<" ";

    return 0;

****}

**5.**

#include<iostream>

#include <string.h>

using namespace std;

int indexOf(const char s1[], const char s2[])//定义寻找子串

{

    int len\_s1 = strlen(s1);//求s1长度

    int len\_s2 = strlen(s2);//求s2长度

    int equal = 0;//计数=0

    for (int i = 0; i < len\_s2; i++)

    {

        equal = 0;//计数=0

        if (s1[0] == s2[i] && len\_s1 <= (len\_s2 - i + 1))//如果子串的第一项与母串的某一项相等 且子串长度小于母串剩余长度

        {

            equal++;//计数+1

            for (int j = 1; j < len\_s1; j++)//建立循环

            {

                i++;//i+1

                if (s1[j] == s2[i])//如果子串下一项和母串下一项相等

                {

                    equal++;//计数+1

                }

                else//有不相等

                {

                    break;//跳出循环

                }

            }

            if (equal == len\_s1)//如果相等数和s1长度相等

            {

                return (i - len\_s1 + 1);//返回子串下标

            }

            else

            {

                continue;//否则进行下次循环

            }

        }

    }

    return -1;//不是子串关系 返回-1

}

int main()

{

    char s1[100], s2[100];//定义两个字符串

    cout << "请输入string1(子串):";//输入字符串s1

    cin.getline(s1, 100);

    cout << "请输入string2(母串):";//输入字符串s2

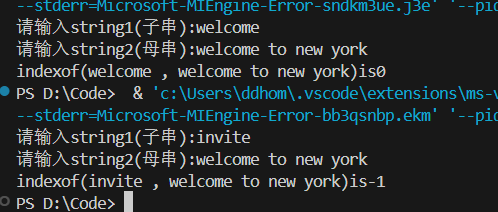
    cin.getline(s2, 100);

    int indexof=indexOf(s1, s2);//调用函数

    cout << "indexof(" << s1 << " , " << s2 << ")is" << indexof << endl;

    return 0;

}

****

6. #include<iostream>

#include<string>

#include <map>

using namespace std;

int main()

{

    map<char,int> cnt;

    string s;

    getline(cin,s);

    for(char c:s)

    {

        c=tolower(c);

        if(isalpha(c))

        cnt[c]++;

    }

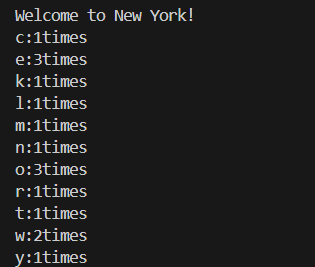
    for(auto p:cnt)

    {

        cout<<p.first<<":"<<p.second<<"times"<<endl;

    }

    return 0;

****}

**程序设计：**

**1.** #include<iostream>

#include <string.h>

using namespace std;

int indexOf(const char s1[], const char s2[])//定义寻找子串

{

    int len\_s1 = strlen(s1);//求s1长度

    int len\_s2 = strlen(s2);//求s2长度

    int equal = 0;//计数=0

    for (int i = 0; i < len\_s2; i++)

    {

        equal = 0;//计数=0

        if (s1[0] == s2[i] && len\_s1 <= (len\_s2 - i + 1))//如果子串的第一项与母串的某一项相等 且子串长度小于母串剩余长度

        {

            equal++;//计数+1

            for (int j = 1; j < len\_s1; j++)//建立循环

            {

                i++;//i+1

                if (s1[j] == s2[i])//如果子串下一项和母串下一项相等

                {

                    equal++;//计数+1

                }

                else//有不相等

                {

                    break;//跳出循环

                }

            }

            if (equal == len\_s1)//如果相等数和s1长度相等

            {

                return (i - len\_s1 + 1);//返回子串下标

            }

            else

            {

                continue;//否则进行下次循环

            }

        }

    }

    return -1;//不是子串关系 返回-1

}

int main()

{

    char s1[100], s2[100];//定义两个字符串

    cout << "请输入string1(子串):";//输入字符串s1

    cin.getline(s1, 100);

    cout << "请输入string2(母串):";//输入字符串s2

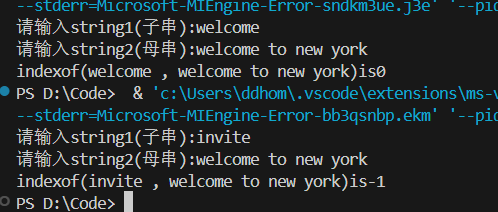
    cin.getline(s2, 100);

    int indexof=indexOf(s1, s2);//调用函数

    cout << "indexof(" << s1 << " , " << s2 << ")is" << indexof << endl;

    return 0;

}

****

**2.**

#include <iostream>

#include <string>

#include <sstream>

using namespace std;

// 将16进制字符串转换为10进制整数

int parseHex(const string &hexStr) {

    int decimalValue = 0;

    stringstream ss;

    ss << hex << hexStr;

    ss >> decimalValue;

    return decimalValue;

}

int main() {

    string hexStr;

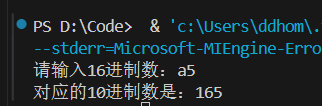
    cout << "请输入16进制数：";

    cin >> hexStr;

    int decimalValue = parseHex(hexStr);

    cout << "对应的10进制数是：" << decimalValue << endl;

    return 0;

****}

**3.**

#include<iostream>

using namespace std;

void bubble\_sort(int\* arr, int size)

{

    for (int i = 0; i < size - 1; i++)

    {

        for (int j = 0; j < size - i - 1; j++)

        {

            if (arr[j] > arr[j + 1])

            {

                int temp = arr[j];

                arr[j] = arr[j + 1];

                arr[j + 1] = temp;

            }

        }

    }

}

int main()

{

    int size;

    cout << "请输入数组大小:";

    cin >> size;

    int\* arr = new int[size];

    cout << "请输入数组数值:";

    for (int i = 0; i < size; i++)

    {

        cin >> arr[i];

    }

    bubble\_sort(arr, size);

    cout << "排序后的数组为:";

    for (int i = 0; i < size; i++)

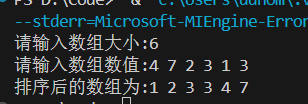
    {

        cout << arr[i] <<" ";

    }

    delete[]arr;

    return 0;

****}

**四、遇到的问题与解决方法**

在解决程序设计的第二题时，不知道应该怎样把一个16进制的字符串转换为10进制的整数。查阅资料，发现了stringstream这种c++独有的高效、类型安全的转换工具。它既可以通过<<写入，也可以通过>>读取，并且通过hex，dec，oct标志可以轻松实现不同进制之间的转化。因此，通过使用stringstream这一工具，实现了题目要求。

**五、体会**

写程序时一定会出现许多自己难以解决的问题，这个时候，我们应该学会查询相关资料，学会更优秀的处理问题的办法。在解决问题时，可以通过目前已学的知识对其进行解决，也可以额外学习一种新的更加高效的解决问题的方法，以拓宽自己的知识面，获得更多解决问题的能力，最终提高自己的综合编程能力以及学习能力。

并且，对于指针的使用，我也有了一些体会：

指针的特殊之处在于它存储的是变量的内存地址，这一特点使其用途非常广泛。在函数调用时，利用指针进行地址传递，形参可以直接修改实参的值，这是普通值传递无法实现的功能。虽然引用也能在一定程度上替代指针，但在某些场景下，指针仍有不可替代的作用。

此外，在定义函数时，可以将指针声明为`const`，以防止指针的值或其指向的地址被修改。如果函数不需要修改指针相关内容，这种方法能够有效避免意外修改，提升代码的安全性和可靠性。

本次实验中，我重点学习了指针的基础操作，例如\*p、&p、\*&p之间的区别。通过对这些概念的深入理解，可以避免在编写指针相关代码时发生错误，从而提高编程效率。

指针为程序功能的实现提供了更多可能性，同时也增加了函数的灵活性。未来的学习中，我会进一步巩固指针的知识，熟练掌握其应用，为代码编写提供更强的支持和更高的可读性。