3.1

#include<iostream>

int GCD(int& x, int& y)

{

int a, b, z = 1;

if (x > y)

a = x, b = y;

else

a = y, b = x;

for (;b != 0;)

{

z = b;

b = a % b;

a = z;

}

return a;

}

int LCM(int& x, int& y, int& z)

{

return (x \* y) / z;

}

int main() {

int m, n, gcd, lcm;

std::cout << "请输入m的值：" << std::endl;

std::cin >> m;

std::cout << "请输入n的值：" << std::endl;

std::cin >> n;

gcd = GCD(m, n);

lcm = LCM(m, n,gcd);

std::cout << "最大公约数是：" << gcd << std::endl;

std::cout << "最小公倍数是：" << lcm << std::endl;

}



3.2

#include<iostream>

using namespace std;

bool is\_prime(int num) {

if (num <= 1)

return false;

int i = 2;

while (i \* i <= num) {

if (num% i == 0)

return false;

i++;

}

return true;

}

int main() {

int count = 0, row = 0;

for (int num = 0;;num++) {

if (is\_prime(num) == true) {

cout << num << '\t';

count++;

if (count == 10) {

cout << '\n';

count = 0;

row++;

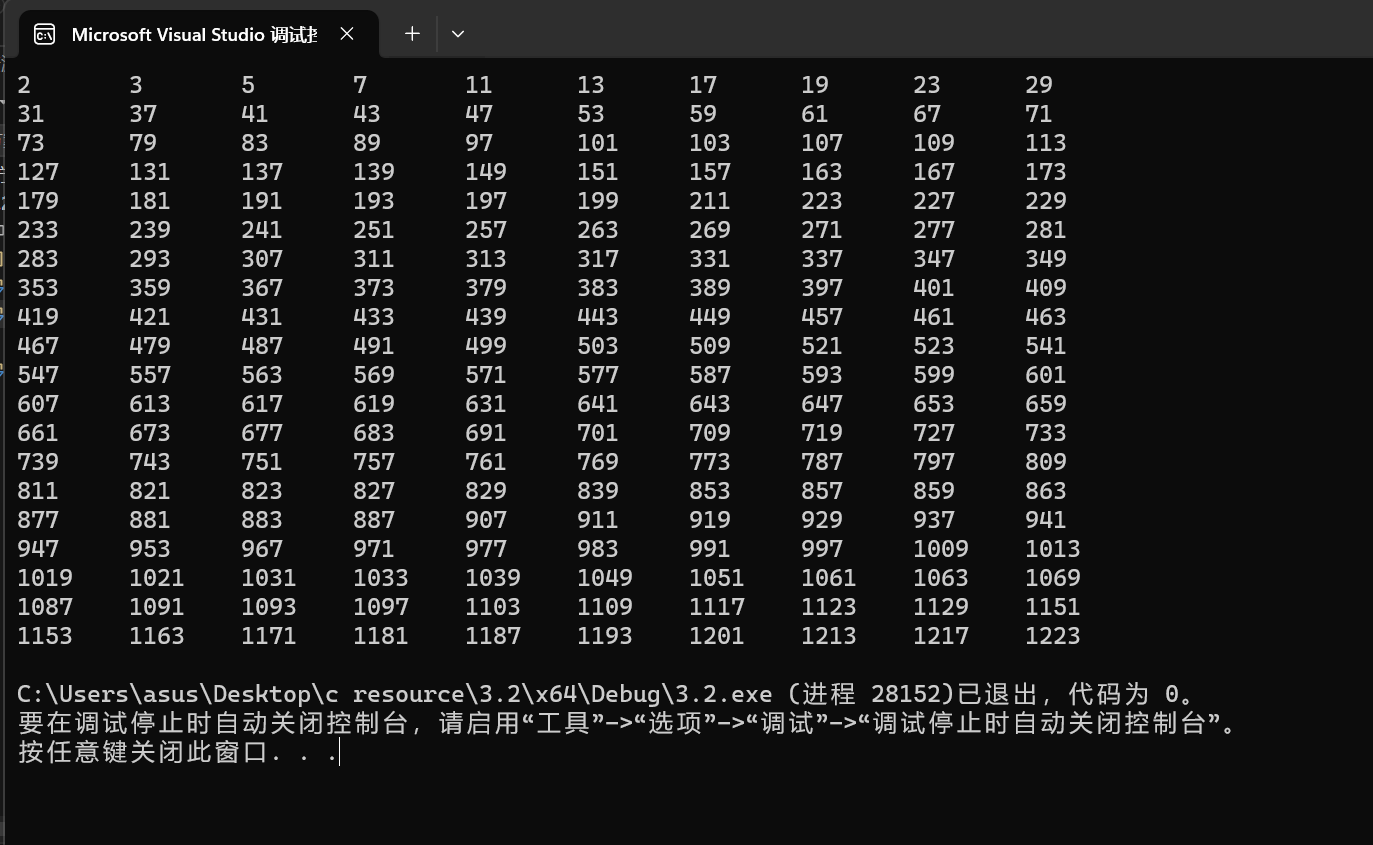
}

}

if (row == 20)break;

}

}



3.4

[mytriangle.h]:

bool is\_valid(double side1, double side2, double side3);

double area(double side1, double side2, double side3);

[mytriangle.cpp]

#include <iostream>

#include <corecrt\_math.h>

using namespace std;

bool is\_valid(double side1, double side2, double side3) {

if ((side1 <= 0 || side2 <= 0 || side3 <= 0) || (side1 + side2 <= side3 || side1 + side3 <= side2 || side2 + side3 <= side1))

return 0;

else

return 1;

}

double area(double side1, double side2, double side3) {

double s;

s = (side1 + side2 + side3) /2;

return sqrt(s \* (s - side1) \* (s - side2) \* (s - side3));

}

[main.cpp]

#include<iostream>

#include "mytriangle.h"

using namespace std;

int main() {

double side1, side2, side3;

cout << "请输入三角形三边长：" << endl;

cin >> side1 >> side2 >> side3;

if (is\_valid(side1, side2, side3) == 1) {

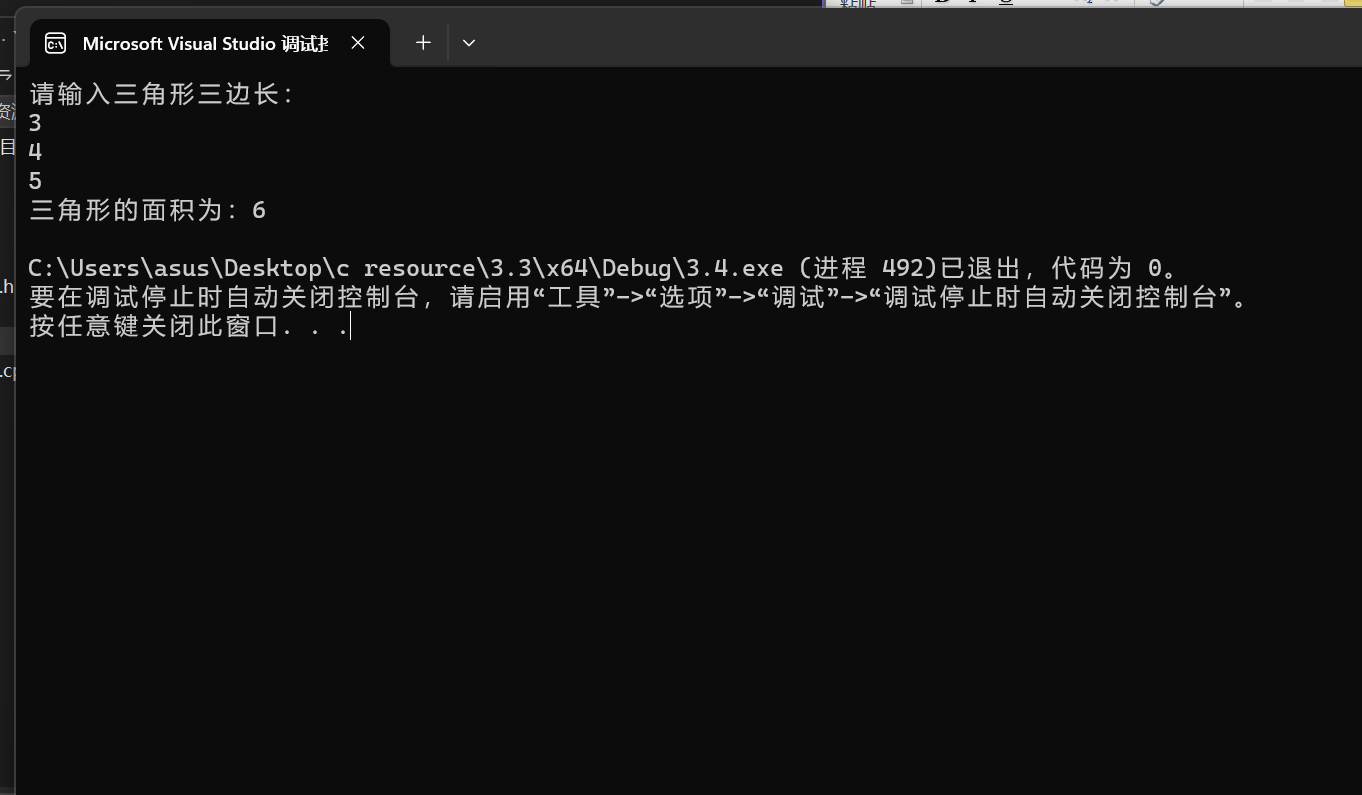
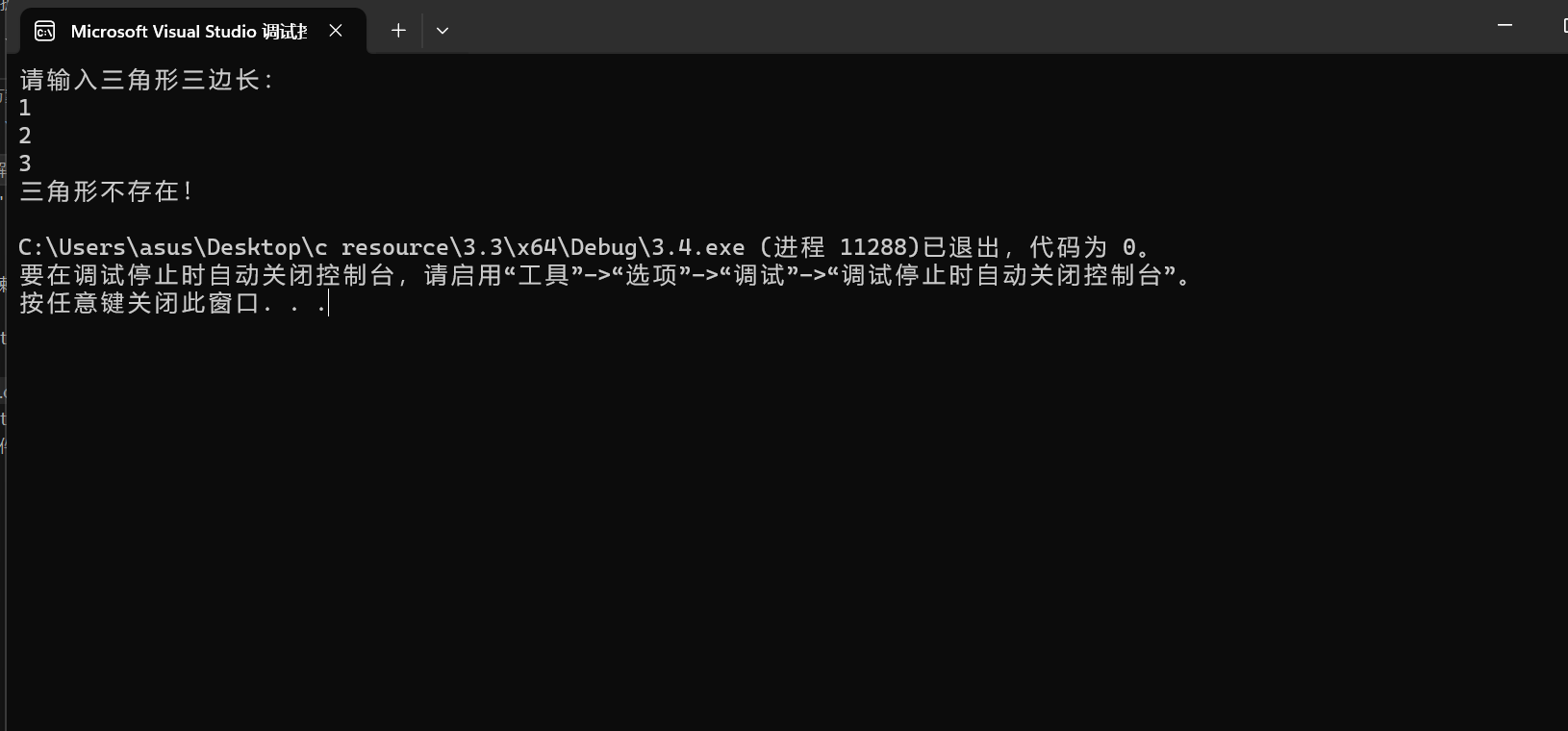
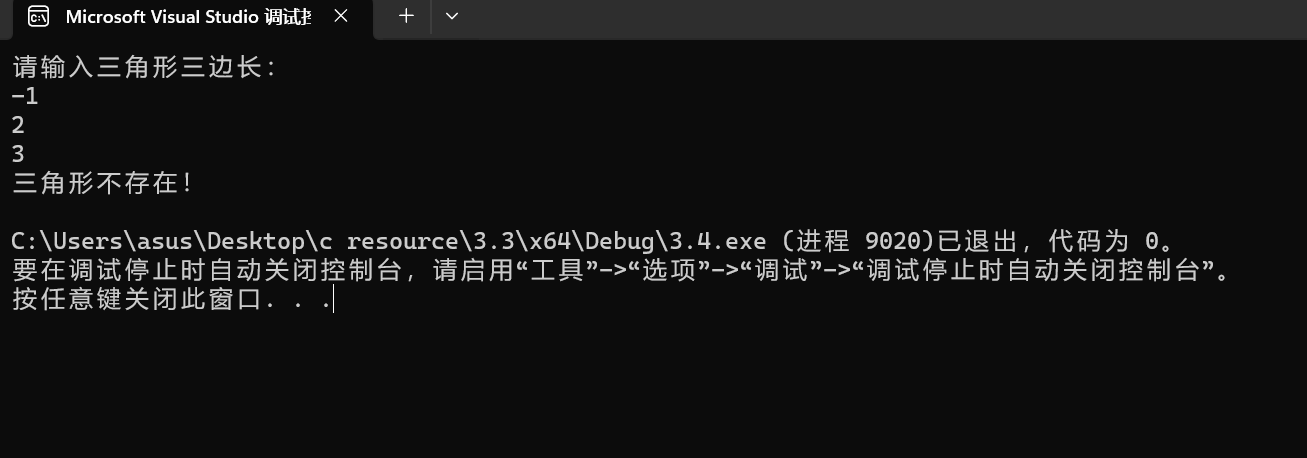
cout << "三角形的面积为：" << area(side1, side2, side3) << endl;

}

else

cout << "三角形不存在！" << endl;

}



3.5

#include<iostream>

using namespace std;

int main() {

int num = 1;

for (int i = 0;i < 10;i++) {

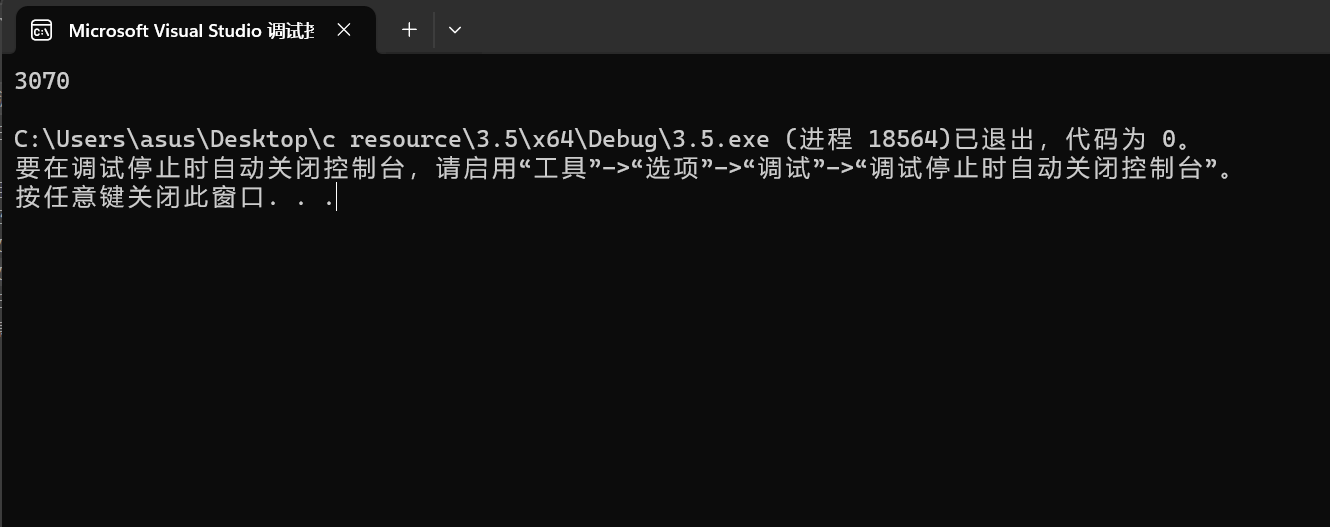
num = (num + 1) \* 2;

}

cout << num << endl;

}

第三次实验我个人认为我完成度较高，并没有明显的卡壳，而且循环与函数的掌握有了明显的进步，虽然偶尔需要找出潜藏的bug，但这次实验算是较为满意的。



4.1

#include<iostream>

using namespace std;

int main() {

int a[10];

cout << "Enter ten numbers：" << endl;

for (int i = 0;i < 10;i++)

cin >> a[i];

cout << "The distict numbers are:" << endl;

cout << a[0]<<" ";

int b[10], num = 0;

for (int j = 1;j < 10;j++) {

for (int k = 0;k < j;k++) {

if (a[j != a[k]]) {

if (k == j - 1) cout << a[j] << " ";

else continue;

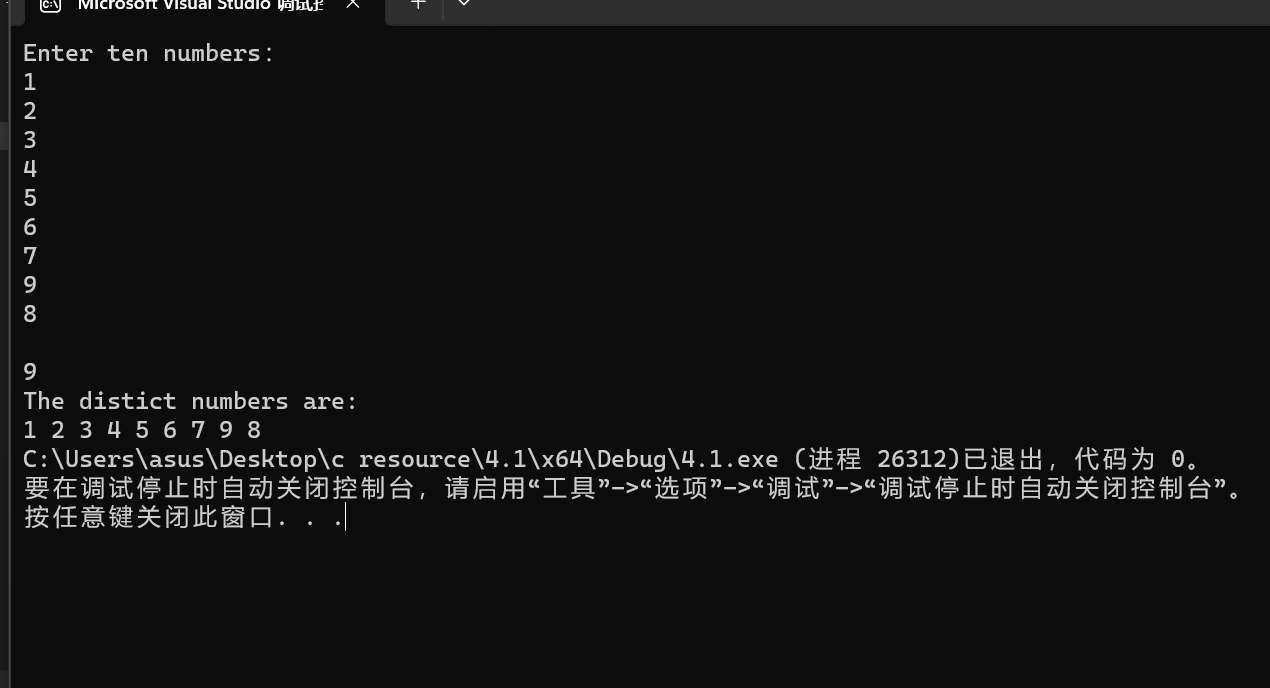
}

else break;

}

}

}



4.2

#include<iostream>

#define N 10

using namespace std;

void BubbleSort(double s[N]) {

double t;

bool changed = true;

do {

changed = false;

for (int j = 0;j < N - 1;j++) {

if (s[j] > s[j + 1]) {

t = s[j];

s[j] = s[j + 1];

s[j + 1] = t;

changed = true;

}

}

} while (changed == true);

}

int main() {

double a[N];

cout << "Enter ten numbers:" << endl;

for (int i = 0;i < N;i++)

cin >> a[i];

cout << "Bubblesorted ten numbers are:" << endl;

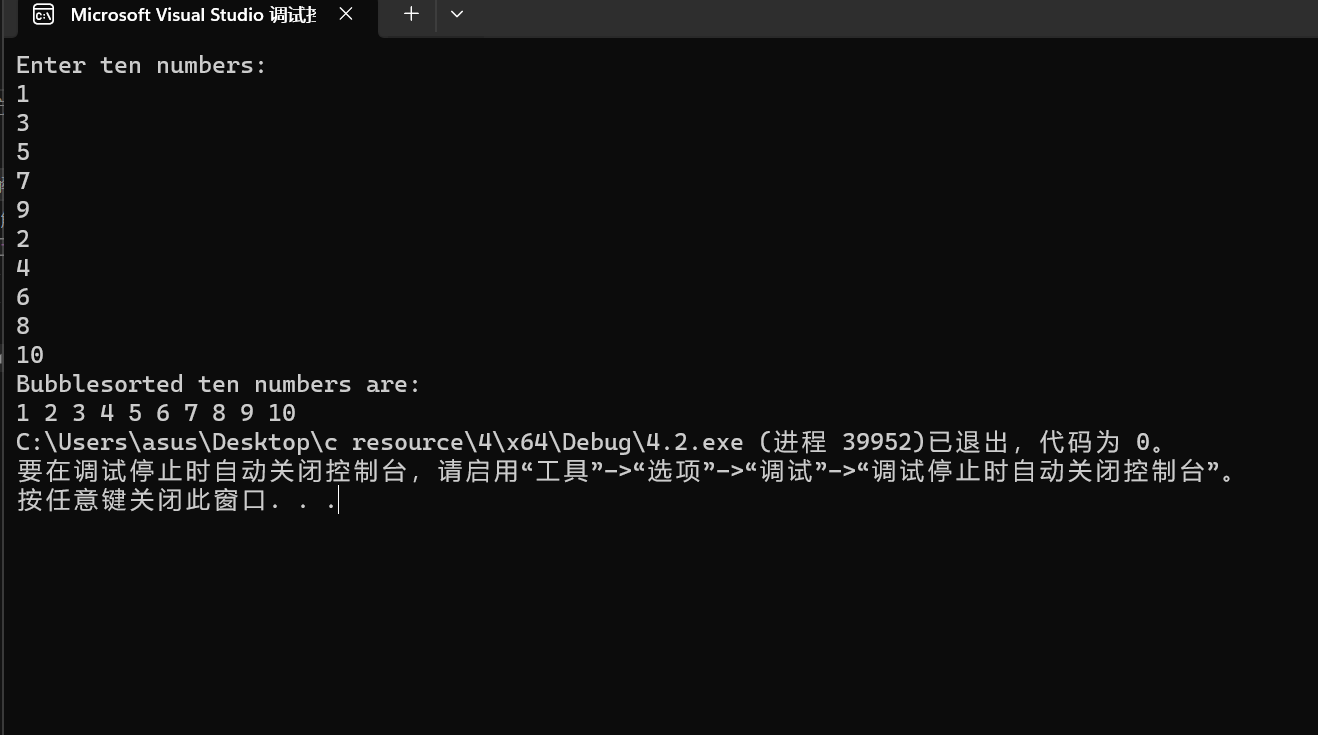
BubbleSort(a);

for (int i = 0;i < N;i++)

cout << a[i] << " ";

return 0;

}



4.3

#include<iostream>

using namespace std;

int main()

{

const int num = 100;

bool list[num];

for (int i = 0;i < num;i++) {

list[num] = false;

}

for (int j = 0;j < num;j++) {

for (int k = j;k < num;k += j + 1) {

if (list[k]) list[k] = false;

else list[k] = true;

}

}

for (int i = 0;i < num;i++) {

if (list[i]) cout << i + 1 << '\t';

}

cout << endl;

return 0;

}

https://img-blog.csdnimg.cn/20190411191310516.png

4.4

#include<iostream>

using namespace std;

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

int t;

for (int i = 0;i < size1;i++) list3[i] = list1[i];

for (int i = 0;i < size2;i++) list3[size1 + i] = list2[i];

cout << "The merged list is :";

for (int j = 0;j < (size1 + size2) - 1;j++) {

for (int k = 0;k < (size1 + size2) - 1;k++) {

if (list3[k] > list3[k + 1]) {

t = list3[k];

list3[k] = list3[k + 1];

list3[k + 1] = t;

}

}

}

for (int m = 0;m < (size1 + size2);m++)

cout << list3[m] << '\t';

}

int main() {

int size1, size2, size3;

cout << "Enter size1:";

cin >> size1;

cout << "Enter list1:";

int\* list1 = new int[size1];

for (int i = 0;i < size1;i++)

cin >> list1[i];

cout << "Enter size2:";

cin >> size2;

cout << "Enter list2:";

int\* list2 = new int[size2];

for (int i = 0;i < size2;i++)

cin >> list2[i];

size3 = size1 + size2;

int\* list3 = new int[size3];

merge(list1, size1, list2, size2, list3);

delete[]list1;list1 = NULL;

delete[]list2;list2 = NULL;

delete[]list3;list3 = NULL;

cout << endl;

return 0;

}



4.5

#include<iostream>

#include<string>

using namespace std;

int indexOf(char\* s1, char\* s2) {

int a, b, index, j, k;

a = strlen(s1);

b = strlen(s2);

bool\* s3 = new bool[a];

for (int m = 0;m < a;m++) {

s3[m] = false;

}

for (int i = 0;i < b - a + 1;i++) {

for(j=0,k=i;(j<a)&&(k<i+a);j++,k++)

if (s1[j] == s2[k]) {

index = i;

s3[j] = true;

}

int h = 0;

for (int n = 0;n < a;n++) {

if (s3[n] == false && i == b - a) {

return -1;

break;

}

if (s3[n]) h++;

}

if (h == a) return index;

}

delete[]s3;s3 = NULL;

}

int main() {

char s1[999], s2[999];

cout << "Enter the first string:";

cin.getline(s1, 999);

cout << "Enter the second string:";

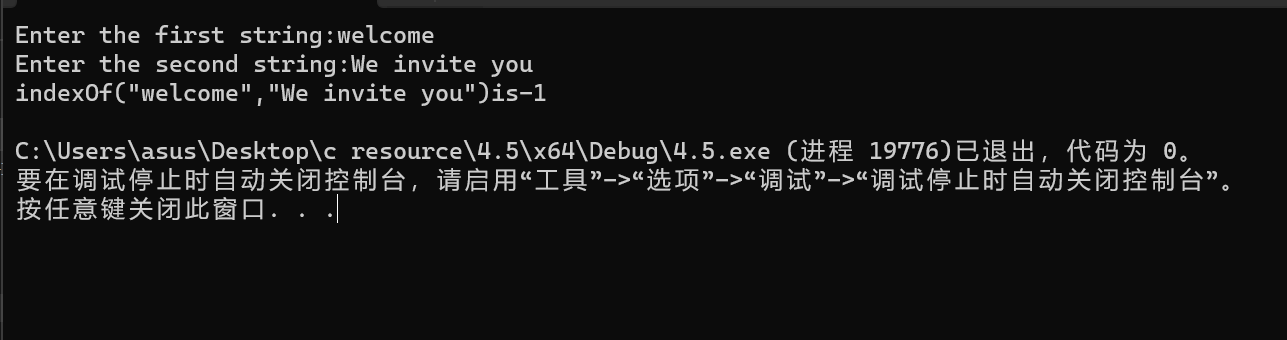
cin.getline(s2, 999);

cout << "indexOf(\"" << s1 << "\",\"" << s2 << "\")is" << indexOf(s1, s2) << endl;

return 0;

}





4.6

#include<iostream>

using namespace std;

const int num1 = 26;

const int num2 = 99;

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

for (int i = 0;i < num1;i++) counts[i] = 0;

for (int j = 0;j < num2;j++) {

counts[s[j] - 'a']++;

counts[s[j] - 'A']++;

}

}

void displayCounts(const int counts[]) {

for (int i = 0;i < num1;i++) {

if (counts[i] != 0) cout << static\_cast<char>(i + 'a') << ":" << counts[i] << " times " << endl;

}

}

int main() {

char s[num2];

int counts[num1];

cout << "Enter a string:";

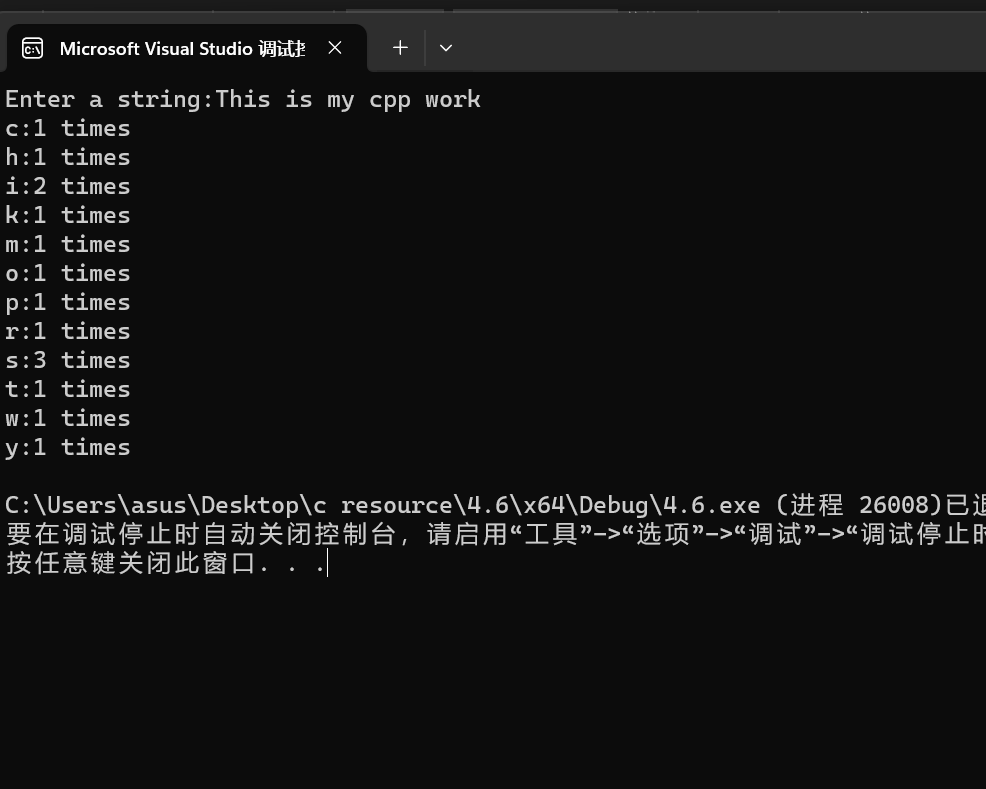
cin.getline(s, num2);

count(s, counts);

displayCounts(counts);

return 0;

}



4.7

#include<iostream>

#include<cstring>

#include<cmath>

using namespace std;

int parseHex(const char\* const hexString);

int main()

{

int size = 999;

char\* s = new char[size];

cout << "Please enter a number in hexChar : \n";

cin.getline(s, size);

cout << "Tne number " << s << " in hexChar is " << parseHex(s) ;

delete[]s;s = NULL;

return 0;

}

int parseHex(const char\* const hexString)

{

int a, sum = 0;

a = strlen(hexString);

int\* list = new int[a];

for (int i = 0;i < a;i++)

{

if (hexString[i] >= 'A' && hexString[i] <= 'F')

list[i] = (static\_cast<int>(hexString[i]) - 55) \* (pow(16, a - 1 - i));

else

list[i] = (hexString[i] - 48) \* pow(16, a - i - 1);

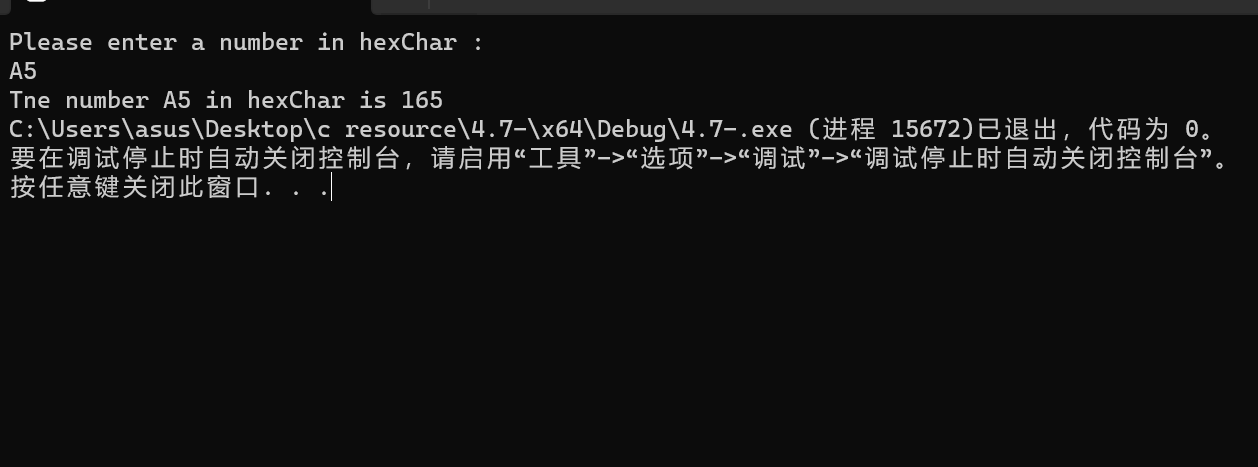
sum = sum + list[i];

}

return sum;

delete[]list;list = NULL;

}



4.8

#include<iostream>

using namespace std;

void arrange(int p[], int size);

int main()

{

int size, a;

cout << "Enter the size :" << endl;

cin >> size;

int\* p = new int[size];

cout << "Enter the nums:" << endl;

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

cin >> \*(p + i);

cout << "Enter the arrange you want:" << endl;

cin >> a;

cout << "p[" << a - 1 << "] = " << \*(p + a - 1) << endl;

arrange(p, size);

for (int b = 0;b < size;b++)

cout << \*(p + b) << " ";

cout << endl;

delete p;p = NULL;

system("pause");

return 0;

}

void arrange(int p[], int size)

{

int temp;

for (int n = 0;n < size;n++)

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

if (\*(p + m) > \*(p + m + 1))

{

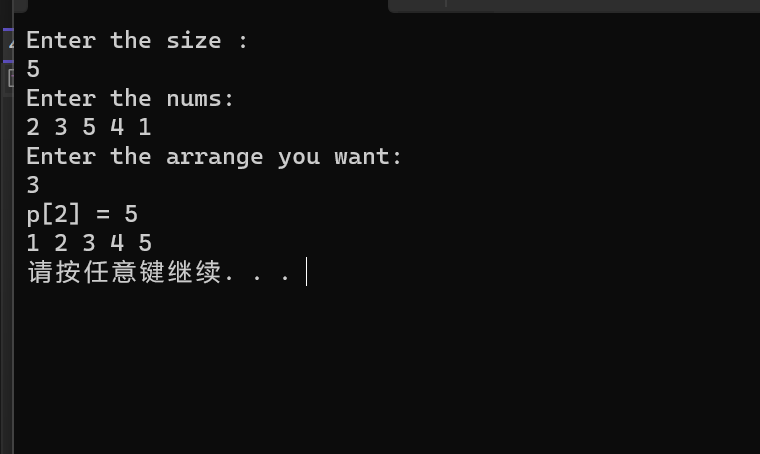
temp = \*(p + m);

\*(p + m) = \*(p + m + 1);

\*(p + m + 1) = temp;

}

}



第四次实验的确让我发现我的明显的短板。当程序复杂性提高时，我明显感觉到对代码的布局的吃力。必须承认本次实验很大一部分借助了查找，但并非单纯的誊抄，而是建立在对其代码的解读和理解下，加入了我个人一些改动后的代码。这次实验暴露出我对指针的生疏，在使用的过程中存在非常多的问题，而这些问题也将在接下来的学习中被我解决。