**计算机程序设计基础（C++)**

**实验报告**

专业班级： 软工2403

学 号： 8209240304

姓 名： 鞠昀壮

**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？

这是编写代码的语言规范，返回什么类型的值就设定什么类型的函数，如果返回值的类型与函数类型不同，就要进行强制转换。

1. 本实验中主函数调用函数时采用的是何种传递方式？

用的是引用传递

1. **算法分析，程序结果**

**1.**

#include<iostream>

using namespace std;

int count1(int num1, int num2) {

int max = 0, yu = 1;

int a = num1;

int b = num2;

while (yu != 0) {

yu = num1 % num2;

num1 = num2;

num2 = yu;

}

max = num1;

return max;

}

int count2(int num1, int num2) {

int max = 0,min=0, yu = 1;

int a = num1;

int b = num2;

while (yu != 0) {

yu = num1 % num2;

num1 = num2;

num2 = yu;

}

max = num1;

min = a \* b / max;

return min;

}

int main()

{

int num1 = 0, num2 = 0, max = 0, min = 0, yu = 1, a = 0, b = 0;

cin >> num1 >> num2;

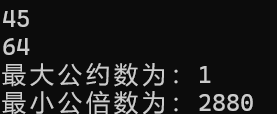
cout << "最大公约数为：" << count1(num1, num2) << endl;

cout << "最小公倍数为：" << count2(num1, num2) << endl;

system("pause");

return 0;

}



#include <iostream>

using namespace std;

void getGCDandLCM(int m, int n, int& gcd, int& lcm) {

int a = m;

int b = n;

while (b != 0) {

int temp = b;

b = a % b;

a = temp;

}

gcd = a;

lcm = (m \* n) / gcd;

}

int main() {

int m, n;

cout << "请输入两个自然数m和n：";

cin >> m >> n;

int greatest\_common\_divisor;

int least\_common\_multiple;

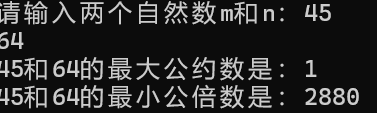
getGCDandLCM(m, n, greatest\_common\_divisor, least\_common\_multiple);

cout << m << "和" << n << "的最大公约数是：" << greatest\_common\_divisor << endl;

cout << m << "和" << n << "的最小公倍数是：" << least\_common\_multiple << endl;

return 0;

}



**2.**#include<iostream>

using namespace std;

bool is\_prime(int num) {

for (int i = 2; i < num; i++) {

if (num % i == 0) {

return false;

break;

}

}

return true;

}

int main() {

int num = 0;

int t = 0;

while ( t < 200) {

if (is\_prime(num)) {

cout << num<<" ";

t++;

}

if (t%10==0) {

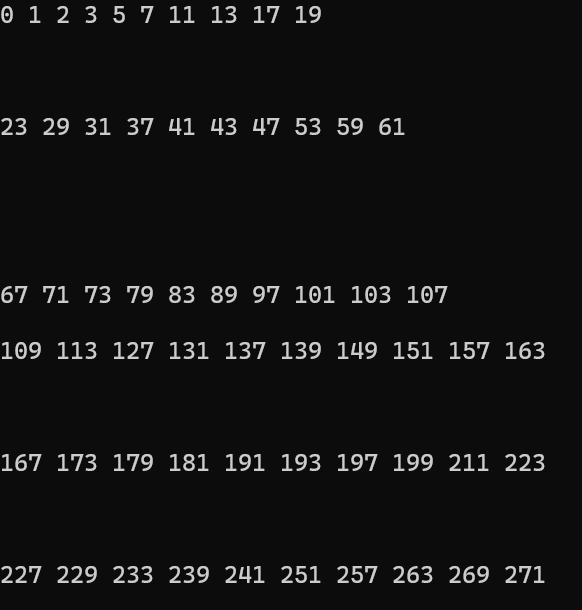
cout << endl;

}

num++;

}

}



**3.**

#include "mytemperature.h"

#include <iostream>

#include <iomanip>

int main() {

std::cout << "Celsius\tFahrenheit\t|\tFahrenheit\tCelsius\n";

std::cout << std::fixed << std::setprecision(2);

for (double c = 40.0; c >= 31.0; c -= 1.0) {

double f = celsius\_to\_fah(c);

std::cout << std::setw(7) << c << "\t\t" << std::setw(7) << f << "\t\t|\t";

double f2 = 120.0 - (40.0 - c) \* 10.0;

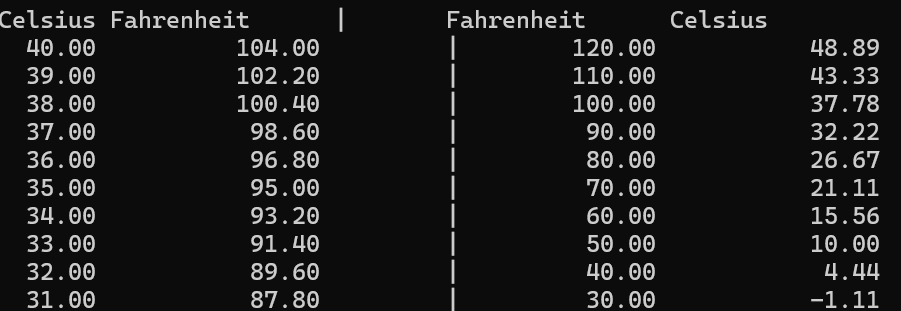
double c2 = fahrenheit\_to\_cels(f2);

std::cout << std::setw(7) << f2 << "\t\t" << std::setw(7) << std::setprecision(2) << c2 << "\n";

}

return 0;

}

**5.**

#include <iostream>

int peach\_count(int n) {

if (n == 10) {

return 1;

}

return (peach\_count(n + 1) + 1) \* 2;

}

int main() {

int total\_peaches = peach\_count(1);

std::cout << "第一天猴子共摘了 " << total\_peaches << " 个桃子。" << std::endl;

return 0;

}



1. **遇到的问题与解决方法**

**遇到了一些计算方法的问题与编辑头文件的新知识**

**在网络上搜索相关知识启发思路后，顺利地用递归的方法实现了功能**

1. **体会**

**在实现功能时，我们不仅要满足于现有的知识，还要探索多种方式完成任务，以满足多样化的需求，并提升自己的专业能力。**

**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 “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\_\_\_\_\_\_\_\_

(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**

1. **算法分析，程序结果**
2. **：1.**

#include <iostream>

int main() {

int input\_numbers[10];

int result\_numbers[10];

int count = 0;

std::cout << "请依次输入10个数字：" << std::endl;

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

std::cin >> input\_numbers[i];

}

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

bool isNew = true;

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

if (input\_numbers[i] == result\_numbers[j]) {

isNew = false;

break;

}

}

if (isNew) {

result\_numbers[count] = input\_numbers[i];

count++;

}

}

std::cout << "不同的数字为：";

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

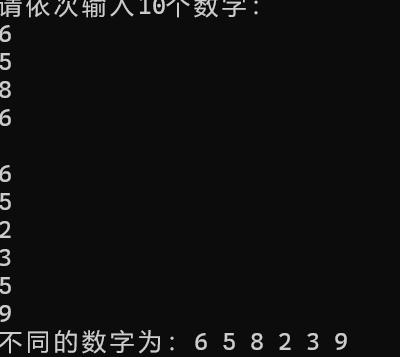
std::cout << result\_numbers[i] << " ";

}

std::cout << std::endl;

return 0;

}



**2.**#include <iostream>

void swap(double& num1, double& num2) {

double temp = num1;

num1 = num2;

num2 = temp;

}

void bubbleSort(double list[], int listSize) {

bool changed = true;

do {

changed = false;

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

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

swap(list[j], list[j + 1]);

changed = true;

}

}

} while (changed);

}

int main() {

double numbers[10];

std::cout << "请输入10个双精度数字：" << std::endl;

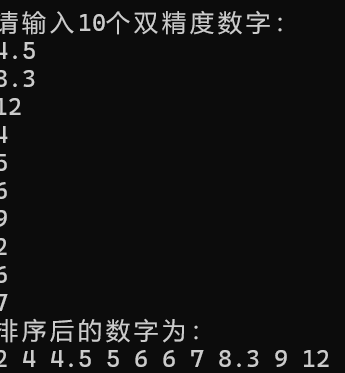
for (int i = 0; i < 10; i++) {

std::cin >> numbers[i];

}

bubbleSort(numbers, 10);

std::cout << "排序后的数字为：" << std::endl;

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

std::cout << numbers[i] << " ";

}

std::cout << std::endl;

return 0;

}

**3.**

#include<iostream>

using namespace std;

bool change(bool n) {

if (n = 0) {

n = 1;

return n;

}

if (n = 1) {

n = 0;

return n;

}

}

int main() {

bool state[100] = { 0};

for (int i = 0; i < 100; i++) {

for (int j = 1; j< 101; j=j+i+1) {

state[j] = !state[j];

}

}

for (int i = 0; i < 100; i++) {

if (state[i]) {

cout << i << " ";

}

}

return 0;

}



**4.**

#include <iostream>

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

int i = 0, j = 0, k = 0;

while (i < size1 && j < size2) {

if (list1[i] < list2[j]) {

list3[k++] = list1[i++];

}

else {

list3[k++] = list2[j++];

}

}

while (i < size1) {

list3[k++] = list1[i++];

}

while (j < size2) {

list3[k++] = list2[j++];

}

}

int main() {

int list1[80], list2[80], list3[160];

int size1, size2;

std::cout << "请输入第一个已排序数组的元素个数：";

std::cin >> size1;

std::cout << "请依次输入第一个已排序数组的元素：";

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

std::cin >> list1[i];

}

std::cout << "请输入第二个已排序数组的元素个数：";

std::cin >> size2;

std::cout << "请依次输入第二个已排序数组的元素：";

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

std::cin >> list2[i];

}

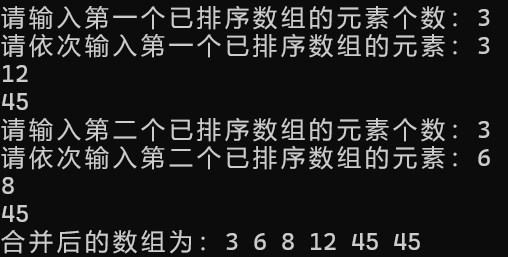
merge(list1, size1, list2, size2, list3);

std::cout << "合并后的数组为：";

for (int i = 0; i < size1 + size2; i++) {

std::cout << list3[i] << " ";

}

 std::cout << std::endl;

return 0;

}

**5.**

#include <iostream>

using namespace std;

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

int len1 = 0, len2 = 0;

while (s1[len1] != '\0') {

len1++;

}

while (s2[len2] != '\0') {

len2++;

}

for (int i = 0; i <= len2 - len1; i++) {

int j;

for (j = 0; j < len1; j++) {

if (s2[i + j] != s1[j]) {

break;

}

}

if (j == len1) {

return i;

}

}

return -1;

}

int main() {

const int MAX\_LENGTH = 100;

char s1[MAX\_LENGTH];

char s2[MAX\_LENGTH];

cout << "请输入第一个字符串：" << endl;

cin.getline(s1, MAX\_LENGTH);

cout << "请输入第二个字符串：" << endl;

cin.getline(s2, MAX\_LENGTH);

int result = indexOf(s1, s2);

if (result != -1) {

cout << s1 << " 是 " << s2 << " 的子串，下标为 " << result << endl;

}

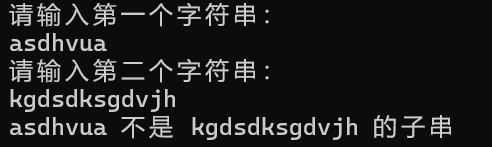
else {

cout << s1 << " 不是 " << s2 << " 的子串" << endl;

}

return 0;

}



**6.**

#include<iostream>

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

for (int i = 0; s[i] != '\0'; i++) {

char ch = s[i];

if ((ch >= 'a' && ch <= 'z') || (ch >= 'A' && ch <= 'Z')) {

if (ch >= 'A' && ch <= 'Z') {

ch = ch - 'A' + 'a';

}

counts[ch - 'a']++;

}

}

}

int main() {

const int MAX\_LENGTH = 100;

char s[MAX\_LENGTH];

int counts[26] = { 0 };

std::cout << "请输入一个字符串：" << std::endl;

std::cin.getline(s, MAX\_LENGTH);

count(s, counts);

std::cout << "字符串中字母出现的次数如下：" << std::endl;

for (int i = 0; i < 26; i++) {

if (counts[i] > 0) {

char ch = 'a' + i;

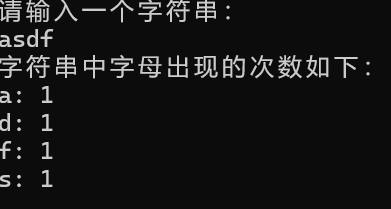
std::cout << ch << ": " << counts[i] << std::endl;

}

}

return 0;

}



**（二）1.**

#include <iostream>

using namespace std;

int main()

{

int i, j, \* pi, \* pj;

pi = &i;

pj = &j;

i = 5;

j = 7;

cout << i << '\t' << j << '\t' << pi << '\t' << pj << endl;

cout << &i << '\t' << \*&i << '\t' << &j << '\t' << \*&j << endl;

return 0;

}



**2.**

#include <stdio.h>

int main() {

int a[] = { 1, 2, 3 };

int\* p, i;

p = a;

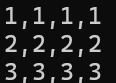
for (i = 0; i < 3; i++) {

printf("%d,%d,%d,%d\n", a[i], p[i], \*(p + i), \*(a + i));

}

return 0;

}



**3.**

#include <stdio.h>

using namespace std;

void f(char\* st, int i)

{

st[i] = '\0';

printf(" % s\n", st); // printf(“%s\n”,st);

if (i > 1) f(st, i - 1);

}

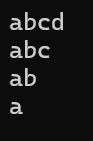
int main()

{

char st[] = "abcd";

f(st, 4);

}



**4.**

#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;

}

int main() {

int\* p = f();

cout << p[0] << endl;

cout << p[1] << endl;

delete[] p;

return 0;

}



**1.**

#include <iostream>

#include <cstring>

using namespace std;

int indexof(const char\* s1, const char\* s2) {

int len1 = strlen(s1);

int len2 = strlen(s2);

if (len1 > len2) {

return -1;

}

for (int i = 0; i <= len2 - len1; i++) {

int j;

for (j = 0; j < len1; j++) {

if (s2[i + j] != s1[j]) {

break;

}

}

if (j == len1) {

return i;

}

}

return -1;

}

int main() {

char s1[100], s2[100];

cout << "请输入字符串s1：" << endl;

cin >> s1;

cout << "请输入字符串s2：" << endl;

cin >> s2;

int result = indexof(s1, s2);

if (result == -1) {

cout << "s1不是s2的子串" << endl;

}

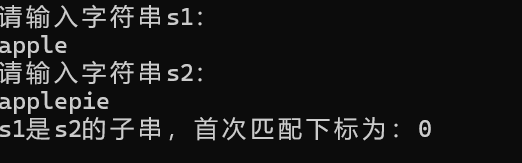
else {

cout << "s1是s2的子串，首次匹配下标为：" << result << endl;

}

return 0;

}



**2.**

#include <iostream>

#include <cctype>

#include <string>

using namespace std;

int parseHex(const char\* const hexString) {

int result = 0;

int len = strlen(hexString);

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

char ch = hexString[i];

int digit;

if (isdigit(ch)) {

digit = ch - '0';

}

else if (ch >= 'A' && ch <= 'F') {

digit = ch - 'A' + 10;

}

else if (ch >= 'a' && ch <= 'f') {

digit = ch - 'a' + 10;

}

else {

return -1;

}

result = result \* 16 + digit;

}

return result;

}

int main() {

char hexStr[100];

cout << "请输入十六进制字符串: ";

cin >> hexStr;

int decimal = parseHex(hexStr);

if (decimal != -1) {

cout << "转换后的十进制数为: " << decimal << endl;

}

else {

cout << "输入的十六进制字符串包含非法字符，请重新输入。" << endl;

}

return 0;

}



**3.**

#include <iostream>

using namespace std;

void swap(int\* a, int\* b) {

int temp = \*a;

\*a = \*b;

\*b = temp;

}

void bubbleSort(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]) {

swap(&arr[j], &arr[j + 1]);

}

}

}

}

int main() {

int size;

cout << "请输入数组元素个数: ";

cin >> size;

int\* arr = new int[size];

cout << "请依次输入数组元素: ";

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

cin >> arr[i];

}

bubbleSort(arr, size);

cout << "排序后的数组元素为: ";

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

cout << \*(arr + i) << " ";

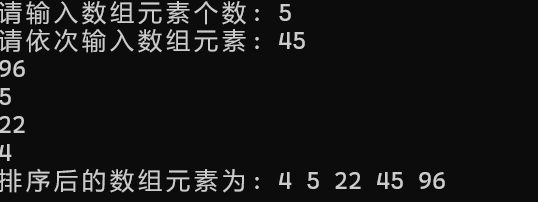
}

cout << endl;

delete[] arr;

return 0;

}



**四、遇到的问题与解决方法**

**在编写代码时偶尔遇到拿不准指针、变量定义的问题，以及递归、遍历的层次混乱的问题。这是由于逻辑思维、层次意识缺乏练习的结果。最终依靠请教同学、从头至尾数次浏览代码寻找问题，甚至是用草稿假设结构再打代码的方式得到解决**

**五、体会**

编程语言的学习遇到难关是正常的，越是遇到难题，越该发扬我们坚持不懈的精神，用创新的思维解决问题。