

课程名称	高级语言程序设计		实验名称	高级语言程序设计实践	
班级		姓名		学号	
仪器组号			实验日期		
实验报告要求：1.实验目的 2.实验内容（题目描述，源代码，运行截图，调试情况） 3.实验体会					
<div>一、实验目的</div> <div><div>1. 认识了解 c++基本语法</div><div>2. 掌握条件语句和循环语句</div><div>3. 掌握数组和字符串的表达。</div></div> <div>二、实验内容</div> <div><div>1、第一题</div><div>1.1 题目描述</div><div><div>A. 问题描述</div><p>在进行文章重复度检查时，经常需要统计一段英文中的单词数量，并找出长度最长的单词。</p><p>设有如下定义：char str[500];</p><p>编写程序，通过利用 cin.getline(str,500);实现从键盘输入一小段英文（其中可以包含空格，但在同一行），利用函数统计该段英文中包含几个单词，输出统计出的单词数量、最长单词的长度以及长度最长的单词，空格隔开。</p><p>注意：函数声明使用 void split(char *str);如果有最长的单词不只一个，输出最先找到的那个。</p><div><div>B. 输入</div><p>一小段英文，不要超过 500 个字符</p><div>C. 输出</div><p>单词数量、最长单词的长度以及长度最长的单词，空格隔开。</p></div></div><div>1.2 源代码</div><pre>#include <iostream> #include <cmath> #include <cstring> using namespace std; void split(char* str) { int i, j, t = 0, l[20], max = 0, m = 0; for (i = 0; *(str + i); i++)</pre></div>					

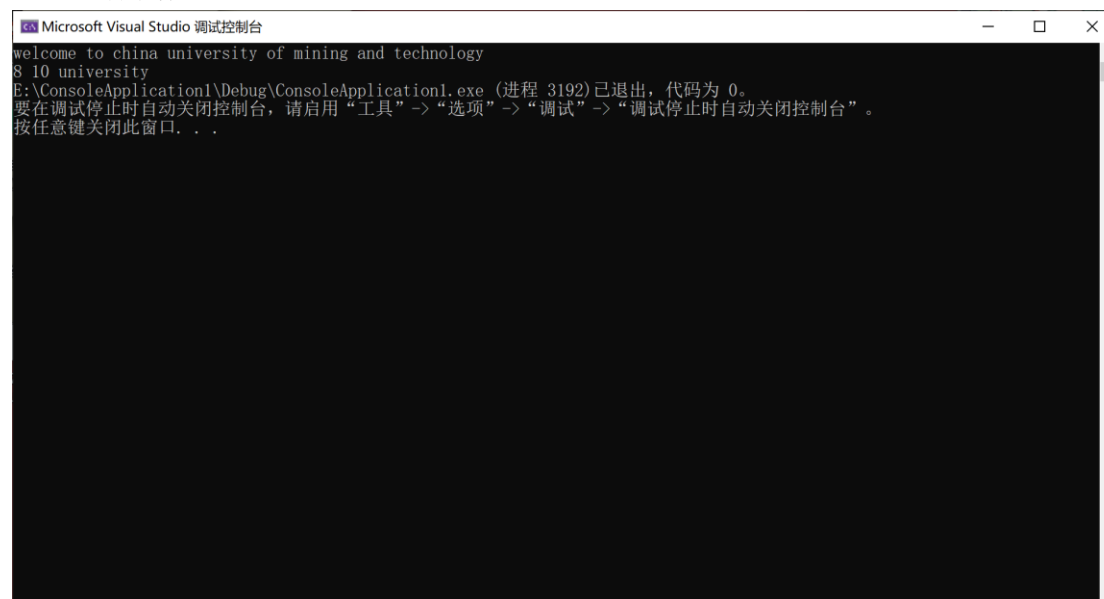
```

{
    if (*(str + i) == ' ')
    {
        l[t] = i;
        t++;
    }
}
l[t] = i;
max = l[0];
for (j = 0; j < t; j++)
{
    if (max < (l[j + 1] - l[j]))
    {
        max = l[j + 1] - l[j];
        m = j;
    }
}
cout << t + 1 << ' ' << max - 1 << ' ';
for (i = l[m] + 1; i < l[m + 1]; i++)
    cout << *(str + i);
}

int main()
{
    char str[500];
    cin.getline(str, 500);
    split(str);
    return 0;
}

```

1.3 运行截图



1.4 调试情况

2、第二题

2.1 题目描述

A. 问题描述、

编程序，按如下方法求 **A** 矩阵的转置矩阵 **B**：输入两个正整数 **m** 和 **n**，而后通过使用指针配合 **new** 运算符生成一个 **m** 行 **n** 列的二维动态数组 **A** 以及另一个 **n** 行 **m** 列的二维动态数组 **B**，之后为 **A** 输入数据（**A** 矩阵数据），逐行逐列输入，进而求出其转置矩阵 **B**（即将 **A** 中的行存放成 **B** 中的列）并输出结果，逐行逐列输出，每一行数字占一行，同一行的数空格隔开。

注意：输入保证全是整数。

B. 输入

输入 **m** 和 **n**,逐行逐列输入数组 **A** 的数据

C. 输出

逐行逐列输出矩阵 **B**。

2.2 源代码

```
#include <iostream>
#include <cmath>
#include <cstring>

using namespace std;

int main()
{
    int m, n, i, j;
    cin >> m >> n;
    int(*p)[100] = new int[100][100];
    for (i = 0; i < m; i++)
        for (j = 0; j < n; j++)
            cin >> (*(p + i) + j);
    for (j = 0; j < n; j++)
    {
        for (i = 0; i < m; i++)
            cout << (*(p + i) + j) << ' ';
        cout << endl;
    }
    return 0;
}
```

2.3 运行截图

```
Microsoft Visual Studio 调试控制台
3 2
119 65
629 679
721 564
119 629 721
65 679 564

E:\ConsoleApplication1\Debug\ConsoleApplication1.exe (进程 2604) 已退出，代码为 0。
要在调试停止时自动关闭控制台，请启用“工具”->“选项”->“调试”->“调试停止时自动关闭控制台”。
按任意键关闭此窗口。...
```

2.4 调试情况

3、第三题

3.1 题目描述

A. 问题描述

编写如下原型的函数：

```
void split(double x,int*iPart,double*fPart);
```

提取出数据 x 的整数部分与小数部分，分别放于 $*iPart$ 与 $*fPart$ 处，由于形参 $iPart$ 与 $fPart$ 都是指针，从而可实现将这两个结果“带回”到主函数中。

在主函数中

输入一个数

输出它的整数部分和小数部分，用空格隔开。

提示：一个 `double` 类型数，强制类型转换后就是 `int`，也就是整数部分，差为小数部分。这两个值用指针 $iPart$ 和指针 $fPart$ 带回（通过修改指针的目标变量值。）

B. 输入

一个数

C. 输出

整数部分 小数部分，用空格隔开

3.2 源代码

```
#include <iostream>
#include <cmath>
#include <cstring>

using namespace std;

void split(double x, int* iPart, double* fPart)
{
```

```

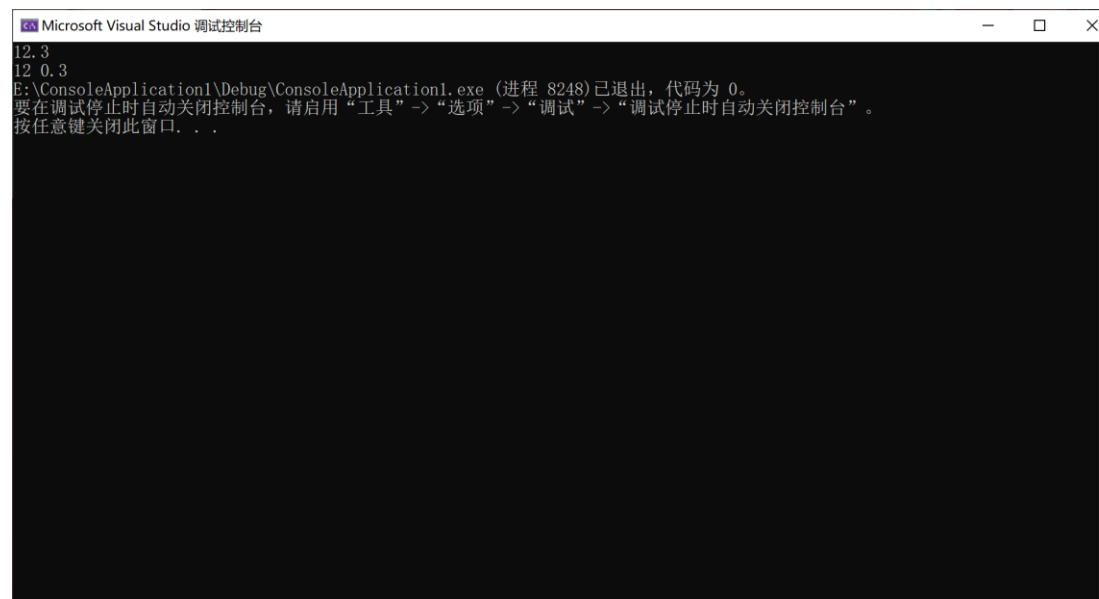
    int a;
    double b;

    a = x;
    b = x - a;
    *iPart = a;
    *fPart = b;
}

int main()
{
    double x, b, * f;
    int a, * i;
    cin >> x;
    f = &b;
    i = &a;
    split(x, i, f);
    cout << a << ' ' << b;
    return 0;
}

```

3.3 运行截图



3.4 调试情况

4、第四题

4.1 题目描述

A. 问题描述

编制具有如下原型的函数 findLast:

```
char*findLast(char*sourceStr,char*subStr);
```

findLast 函数则要返回源串 sourceStr 中最后一次出现 subStr 子字符串的头字符位置。

而后编制主函数，输入两个字符串，将它们用作实参来调用这两个函数，如果返回 NULL 输出-1，否则输出子字符串出现时头字符在原字符串的下标，每个结果占一行。

要求实现程序中不可使用“string.h”头文件内有关寻找子串的标准库函数。

B. 输入

输入源串 `sourceStr`，子字符串 `subStr`。

C. 输出

子字符串 `subStr` 最后一次在源串 `sourceStr` 中出现的位置

4.2 源代码

```
#include <iostream>
#include <cmath>
#include <cstring>

using namespace std;

char* findLast(char* sourceStr, char* subStr)
{
    int x = 0, y = 0, i, j, t = 0;
    while (*(sourceStr + x))
        x++;
    x--;
    while (*(subStr + y))
        y++;
    y--;
    for (i = x; i >= 0; i--)
    {
        if (*(sourceStr + i) == *(subStr + y))
        {
            t = 0;
            for (j = 0; j <= y; j++)
                if (*(sourceStr + i - j) != *(subStr + y - j))
                    t++;
            if (t == 0)
            {
                return sourceStr + i - j + 1;
                break;
            }
        }
    }
    return NULL;
}

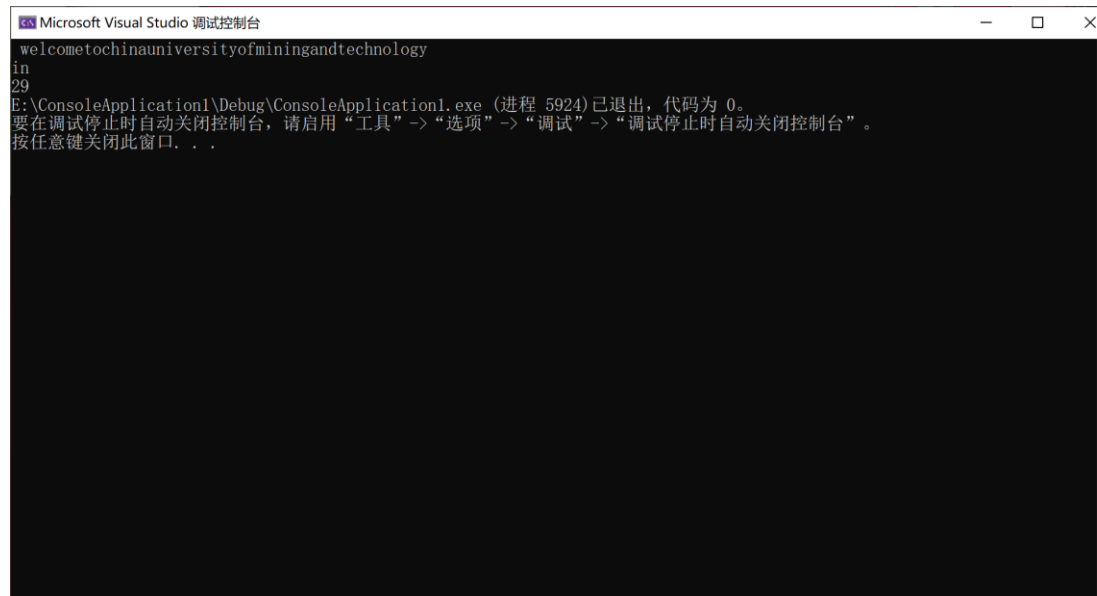
int main()
{
    char a[500], b[50], * p;
```

```

    cin >> a;
    cin >> b;
    p = findLast(a, b);
    if (p == NULL)
        cout << -1;
    else cout << p - a;
    return 0;
}

```

4.3 运行截图



4.4 调试情况

5、第五题

5.1 题目描述

A. 问题描述

编写具有如下原型的函数：

```
int f(unsigned long x, int n,int & Lxn);
```

它负责将整数 x 的第 n 位（从左边数第 n 位， $n>0$ ）的数值放到引用 Lxn 中（将作为结果返回到主调函数的对应实参变量中），并将倒数第 n 位（从右边数第 n 位， $n>0$ ）的数值作为函数结果返回去。

例如当 $x=123456789, n=7$ 时，执行语句 $Rxn=f(x,n, Lxn)$; 将使返回的 Lxn 为 7，并使 Rxn 变为 3；而执行语句 $Rxn=f(12345, 6, Lxn)$ ；将使 Lxn 与 Rxn 都变为 0（超出数的“长度”即总位数时返回 0）

B. 输入

输入 x , 输入 n

C. 输出

输出 Rxn, Lxn

5.2 源代码

```
#include <iostream>
```

```

#include <cmath>
#include <cstring>

using namespace std;

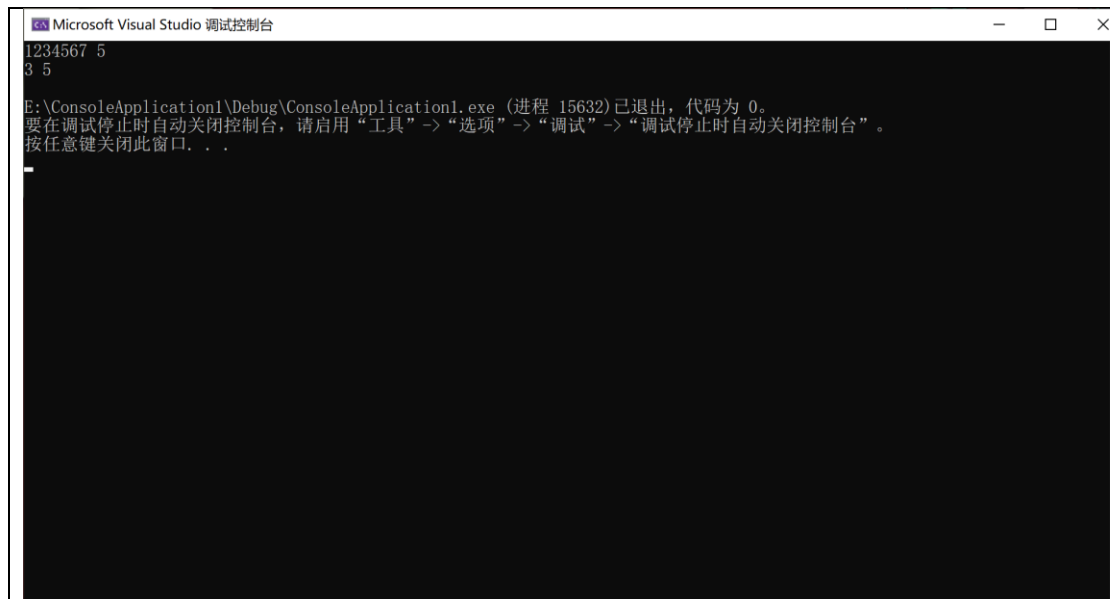
int lenght(unsigned long x)
{
    int n = 0;
    while (x != 0)
    {
        x /= 10;
        n++;
    }
    return n;
}

int f(unsigned long x, int n, int& Lxn)
{
    int number = x;
    int i, len = lenght(x);
    if (n > len)
    {
        Lxn = 0;
        return 0;
    }
    else
    {
        Lxn = int(int(x / pow(10, len - n)) % 10);
        return int(int(x / pow(10, n - 1)) % 10);
    }
}

int main()
{
    unsigned long x;
    int n, Lxn, Rxn;
    cin >> x >> n;
    Rxn = f(x, n, Lxn);
    cout << Rxn << " " << Lxn << endl;
    return 0;
}

```

5.3 运行截图



```
Microsoft Visual Studio 调试控制台
1234567 5
3 5
E:\ConsoleApplication1\Debug\ConsoleApplication1.exe (进程 15632) 已退出, 代码为 0。
要在调试停止时自动关闭控制台, 请启用“工具”->“选项”->“调试”->“调试停止时自动关闭控制台”。
按任意键关闭此窗口。...
```

5.4 调试情况

三、实验体会

通过这次实验,我更加明白了 c++理论中的一些编程规范和 c++语言特性,掌握了基本编程知识,以后会更加认真的学习 c++理论知识,并不断实践和练习,在 debug 中不断学习。