暨南大学本科实验报告专用纸

课程名称 高级语言程序设计实验 成绩评定

实验项目名称 指针的使用 指导教师 张鑫源

实验项目编号 ⑫ 实验项目类型 实验地点 家中

学生姓名 位雨昕 学号 2019051112

学院 智能科学与工程学院 系 专业 信息安全

实验时间 2020 年 6 月 2 日 上午～6 月 8 日 下午 温度 ℃湿度

（一）实验目的

1．进一步了解 Visual Studio 的使用以及 C 语言程序的结构；

2．掌握 C 语言中的常用函数的使用方法；

3．了解值传递和引用传递两种参数传递方式两种方式的机制；

4．锻炼个人的编程操作能力。

（二）实验内容和要求

内容：

以指针的方式实现对二维数组的冒泡排序。对于整型的二维数组 a，设计函数 sort 实现对数组 a 中的元素进行排序。

要求：

1. 设计合理的输出展示实验结果；
2. 排序函数 sort 的参数为指针变量（指针的类型为 int *)；
3. 排序函数 sort 的参数为数组指针；
4. 排序函数 sort 的参数为指针数组；
5. 数组 a 的行和列的值由用户指定，且函数 sort 的排序规则也由用户指定。

（三）主要仪器设备

仪器：计算机

实验环境：Visual Studio Community 2019

（四）源程序

```
#include<stdio.h>
#include <stdlib.h>
#include<time.h>

void bubble_sort1(int* p, int n)//指针作为参数
{
    int choice;
    printf("请选择排序方式：从小到大请输入 1，从大到小请输入 2\n");
    scanf_s("%d", &choice);
    int i, j;
    for (i = 0; i < n; i++)//先从小到大排
    {
        for (j = i; j < n; j++)
        {
            if (*(p + i) > * (p + j))
            {
                int temp;
                temp = *(p + i);
                *(p + i) = *(p + j);
                *(p + j) = temp;
            }
        }
    }
    printf("排序结果为：\n");
    if (choice == 1)//正序输出(从小到大)
    {
        for (i = 0; i < n; i++)
        {
            printf("%4d", *(p + i));
        }
    }
    else if (choice == 2)//逆序输出(从大到小)
    {
        for (i = n - 1; i >= 0; i--)
        {
            printf("%4d", *(p + i));
        }
    }
    else
        printf("输入有误，请重启程序并重新输入");
    printf("\n");

```

```
}
void bubble_sort2(int(*p)[10], int m, int n)//数组指针(行指针)作为参数
{
    int choice;

    printf("请选择排序方式：从小到大请输入 1，从大到小请输入 2\n");
    scanf_s("%d", &choice);
    int i, j;
    int k;
    for (i = 0; i < m; i++)
    {
        for (j = 0; j < n; j++)
        {
            for (k = 0; k < n - j - 1; k++)
            {
                if (*(*(p + i) + k) > * (*(p + i) + k + 1))
                {
                    int temp;
                    temp = *(*(p + i) + k);
                    *(*(p + i) + k) = *(*(p + i) + k + 1);
                    *(*(p + i) + k + 1) = temp;
                }
            }
        }
    }


    printf("排序结果为：\n");
    if (choice == 1)//正序输出(从小到大)
    {

        for (i = 0; i < m; i++)
        {
            for (j = 0; j < n; j++)
                printf("%4d", *(*(p + i) + j));
            printf("\n");
        }
    }
    else if (choice == 2)//逆序输出(从大到小)
    {

        for (i = 0; i < m; i++)
        {
            for (j = n - 1; j >= 0; j--)
```

```
87.                 printf("%4d", *(*(p + i) + j));
88.             printf("\n");
89.         }
90.     }
91.     else
92.         printf("输入有误，请重启程序并重新输入");
93.
94. }
95. void bubble_sort3(int* p[10], int m, int n)//指针数组作为参数
96. {
97.     int choice;
98.     printf("请选择排序方式：从小到大请输入 1，从大到小请输入 2\n");
99.     scanf_s("%d", &choice);
100.     int i, j;
101.     int k;
102.     for (i = 0; i < m; i++)
103.     {
104.         for (j = 0; j < n; j++)
105.         {
106.             for (k = 0; k < n - j - 1; k++)
107.             {
108.                 if (p[i][k] > p[i][k + 1])
109.                 {
110.                     int temp;
111.                     temp = p[i][k];
112.                     p[i][k] = p[i][k + 1];
113.                     p[i][k + 1] = temp;
114.                 }
115.             }
116.         }
117.     }
118.     printf("排序结果为：\n");
119.     if (choice == 1)//正序输出(从小到大)
120.     {
121.
122.         for (i = 0; i < m; i++)
123.         {
124.             for (j = 0; j < n; j++)
125.                 printf("%4d", *(*(p + i) + j));
126.             printf("\n");
127.         }
128.     }
129.     else if (choice == 2)//逆序输出(从大到小)
130.     {
```

```
131.
132.         for (i = 0; i < m; i++)
133.         {
134.             for (j = n - 1; j >= 0; j--)
135.                 printf("%4d", *(*(p + i) + j));
136.             printf("\n");
137.         }
138.     }
139.     else
140.         printf("输入有误，请重启程序并重新输入");
141. }
142.
143. int main()
144. {
145.     int A[10][10], A2[10][10];
146.     int a[30] = { 0 };//一维数组
147.     int ii, jj;//I、J 分别为数组 A 的行和列
148.     int i, j;
149.     printf("请输入二维数组 A 的行数：\n");
150.     scanf_s("%d", &ii);
151.     printf("请输入二维数组 A 的列数：\n");
152.     scanf_s("%d", &jj);
153.     srand((unsigned)time(NULL));
154.     printf("生成的二维数组 A 如下：\n");
155.     for (i = 0; i < ii; i++)
156.     {
157.         for (j = 0; j < jj; j++)
158.         {
159.             A[i][j] = rand() % 100;
160.         }
161.     }
162.     for (i = 0; i < ii; i++)
163.     {
164.         for (j = 0; j < jj; j++)
165.         {
166.             printf("%4d", A[i][j]);
167.         }
168.         printf("\n");
169.     }
170.     int k = 0;
171.     for (i = 0; i < ii; i++) //将二维数组转为一维数组，便以排序
172.     {
173.         for (j = 0; j < jj; j++)
174.         {
```

```
            a[k] = A[i][j];
            k++;
        }
    }
    for (i = 0; i < ii; i++)//复制一个数组 A
    {
        for (j = 0; j < jj; j++)
        {
            A2[i][j] = A[i][j];
        }
    }
    int* pA[10];
    for (i = 0; i < ii; i++)
    {
        pA[i] = A2[i];
    }

    //指针作为参数
    printf("【将指针作为参数】\n");
    bubble_sort1(a, ii * jj);

    //数组指针
    printf("\n【将数组指针作为参数】\n");
    bubble_sort2(A, ii, jj);

    //指针数组
    printf("\n【将指针数组作为参数】\n");
    bubble_sort3(pA, ii, jj);

    return 0;
}
```

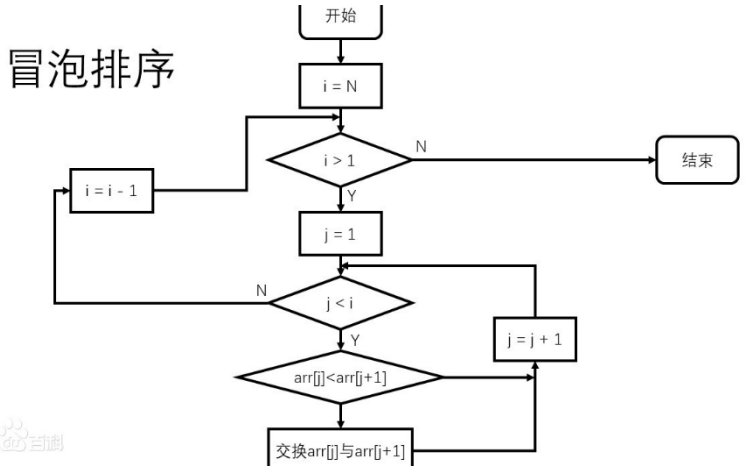
（五）实验步骤与调试

实验步骤：

1. 对实验进行基本构思；
2. 启动 Visual Studio，创建新项目。将源程序写在新项目中；
3. 利用“本地 Windows 调试器”进行调试；
4. 进行多次调试，得出理想结果。

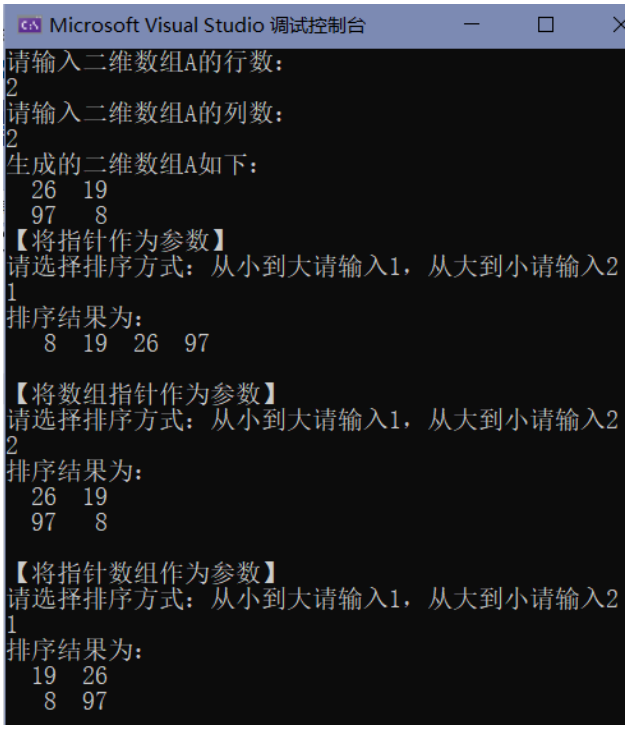
实验思路：

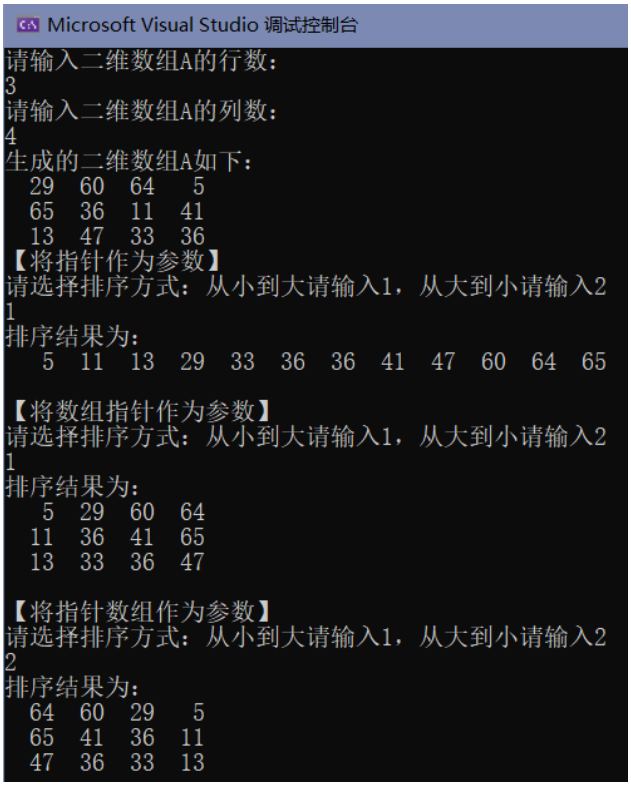
1. 新建源文件，定义 bubble_sort1、bubble_sort2、bubble_sort3 三个函数分别将指针、数组指针、指针数组作为参数，其功能为进行对二维数组的排序并正/逆序输出；
2. 将二维数组 A 先转化为一维数组 a；用 bubble_sort1（参数为指针变量）进行冒泡排序时，一方面使得操作指针的代码更加简单，另一方面使排序结果更加直接地呈现；bubble_sort1 内部操作的就是一维数组；通过二重循环对一维数组进行遍历并交换指针指向，实现排序；
3. 用 bubble_sort2（参数为数组指针）进行冒泡排序时，通过三重循环对二维数组进行遍历，实现冒泡排序；
4. 由于经函数 bubble_sort2 处理，通过引用传递二维数组 A 已经完成了排序，如果处理原本的 A 数组并不能检验函数 bubble_sort3 的功能。因此在调用 bubble_sort2 进行处理之前，定义一个二维数组 A2 用于复制 A，以此检验 bubble_sort3 的调用情况。用 bubble_sort3（参数为指针数组）进行冒泡排序时，同样通过三重循环进行对二维数组的冒泡排序。由于指针数组是由指针组成的数组，所以使用方式和二维数组类似，参考二维数组每一行的冒泡排序进行编写；
5. 实验要求由用户指定排序方式，即可以选择从小到大排序或是从大到小排序。为使代码更加简洁，使用统一先进行从小到大排序，后根据用户选择决定进行正序或逆序输出，从而实现不同排序方式的输出。
6. 冒泡排序法流程：



（六）实验结果与分析

实验结果：





可以看到，将指针/数组指针/指针数组作为参数的三个函数均能实现将二维数组排序并由用户指定方式进行输出的功能。

分析：

1. 在定义以指针作为参数的函数 bubble_sort1 时，由于此时处理的是一维数组，根据冒泡排序法的步骤进行双重循环移动指针进行排序。

```
for (i = 0; i < n; i++)//先从小到大排
{
    for (j = i; j < n; j++)
    {
        if (*(p + i) > * (p + j))
        {
            int temp;
            temp = *(p + i);
            *(p + i) = *(p + j);
            *(p + j) = temp;
        }
    }
}
```

2. 在定义以数组指针作为参数的函数 bubble_sort2 时，此时处理的是二维数组，分别对每行进行排序：最外层循环控制的是行数，内层循环进行每一行元素的冒泡排序，最内循环为每趟比较的次数；(*(*(p + i) + j)为数组 p 第 i 行第 j 列元素所对应的值，与 p[i][j]是等价的。由于此函数处理的是 A 数组本身，因此用后面 bubble_sort3 进行处理时要用此前复制的数组 A2。

```
 for (i = 0; i < m; i++)  {
    for (j = 0; j < n; j++)
    {
        for (k = 0; k < n - j - 1; k++)
        {
            if (*(*(p + i) + k) > * (*(p + i) + k + 1))
            {
                int temp;
                temp = *(*(p + i) + k);
                *(*(p + i) + k) = *(*(p + i) + k + 1);
                *(*(p + i) + k + 1) = temp;
            }
        }
    }
}
```

3. 在定义以指针数组作为参数的函数 bubble_sort3 时，此时处理的是二维数组，分别对每行进行排序：最外层循环控制的是行数，内层循环进行每一行元素的冒泡排序。由于指针数组本质仍为数组，其形式与二维数组类似。

```
for (i = 0; i < m; i++)
{
    for (j = 0; j < n; j++)
    {
        for (k = 0; k < n - j - 1; k++)
        {
            if (p[i][k] > p[i][k + 1])
            {
                int temp;
                temp = p[i][k];
                p[i][k] = p[i][k + 1];
                p[i][k + 1] = temp;
            }
        }
    }
}
```