**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 成绩评定

实验项目名称 以指针的方式实现对二维数组的冒泡设计 指导教师 张鑫源

实验项目编号 十二 实验项目类型 实验地点

学生姓名 张强 学号 2019053448

学院 智能科学与工程 系 略 专业 信息安全

实验时间 2019 年 6月 3日 午～ 6月8 日 午 温度35 ℃湿度

1. **实验目的**

实验内容：以指针的方式实现对二维数组的冒泡排序。

要求：1.设计合理的输出展示实验结果；

2.排序函数sort的参数为数组指针；

3.排序函数sort的参数为指针数组；

4.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；

1. **主要仪器设备**

**仪器：**笔记本电脑

**实验环境：**visual stdio 2019

1. **源程序**

写出程序的源程序。

#include <time.h>

void sort1(int\* p[],int row,int column,int flag);

void sort2(int (\*p)[100], int row, int column, int flag);

int main() {

int rows, columns,flag;//flag

int i, j,temp=0;

int b[100][100];

srand((unsigned)time(NULL)); //

printf("请输入数组的行数：\n");//定义数组的维度，输入行数和列数

scanf\_s("%d", &rows);

printf("请输入数组的列数：\n");

scanf\_s("%d", &columns);

int\*\* a1;

a1 = (int\*\*)malloc(rows \* sizeof(int\*));

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

(a1)[i] = (int\*)malloc(columns\* sizeof(int));

}

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

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

a1[i][j] = rand()%100;

b[i][j] = a1[i][j]; //创建数组b，用于函数sort2的传值

}

}

printf("随机生成的数组如下:\n");

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

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

printf("%3d", a1[i][j]);

}

printf("\n");

}

printf("正序排列请输入1，逆序排列请输入2\n");

scanf\_s("%d",&flag);

printf("排序后的数组如下:\n");

printf("指针数组作为参数为：\n");

sort1(a1, rows, columns,flag);

printf("\n数组指针作为参数为：\n");

sort2(b, rows, columns, flag);

for (i = 0; i < rows; i++)

{

free (a1[i]);

}

free(a1);

return 0;

}

void sort1(int\* p[], int row, int column,int flag) {

int i,j,k=0;

int temp = 0;

int\* p1;

p1 = (int\*)malloc((column \* row )\* sizeof(int));

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

for (j = 0; j < column; j++,k++) {

p1[k] = p[i][j];

}

}

if (flag == 1) {

for (j = 0; j < row \* column; j++) { //正序排列

for (i = 0; i < row \* column - 1; i++) {

if (p1[i] > p1[i + 1]) {

temp = p1[i + 1];

p1[i + 1] = p1[i];

p1[i] = temp;

}

}

}

}

else {

for (j = 0; j < row \* column; j++) { //逆序排列

for (i = 0; i < row \* column - 1; i++) {

if (p1[i] < p1[i + 1]) {

temp = p1[i + 1];

p1[i + 1] = p1[i];

p1[i] = temp;

}

}

}

}

for (int i = 0; i < row\*column; i++) { //输出排列后的数

printf("%3d", p1[i]);

}

free(p1);

}

void sort2(int(\*p)[100], int row, int column, int flag) {

int i, j, k = 0;

int temp = 0;

int\* p1;

p1 = (int\*)malloc((column \* row) \* sizeof(int)); //动态生成一维数组存储二维数组

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

for (j = 0; j < column; j++, k++) {

p1[k] = \*(p[i] + j);

}

}

if (flag == 1) {

for (j = 0; j < row \* column; j++) {

for (i = 0; i < row \* column - 1; i++) {

if (p1[i] > p1[i + 1]) {

temp = p1[i + 1];

p1[i + 1] = p1[i];

p1[i] = temp;

}

}

}

}

else {

for (j = 0; j < row \* column; j++) {

for (i = 0; i < row \* column - 1; i++) {

if (p1[i] < p1[i + 1]) {

temp = p1[i + 1];

p1[i + 1] = p1[i];

p1[i] = temp;

}

}

}

}

for (int i = 0; i < row \* column; i++) {//输出排列之后的数

printf("%3d", p1[i]);

}

free(p1);

}

void sort2(int(\*p)[100], int row, int column, int flag) {

int i, j, k = 0;

int temp = 0;

int\* p1;

p1 = (int\*)malloc((column \* row) \* sizeof(int)); //动态生成一维数组存储二维数组

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

for (j = 0; j < column; j++, k++) {

p1[k] = \*(p[i] + j);

}

}

if (flag == 1) {

for (j = 0; j < row \* column; j++) { //正序排列

for (i = 0; i < row \* column - 1; i++) {

if (p1[i] > p1[i + 1]) {

temp = p1[i + 1];

p1[i + 1] = p1[i];

p1[i] = temp;

}

}

}

}

else {

for (j = 0; j < row \* column; j++) {//逆序排列

for (i = 0; i < row \* column - 1; i++) {

if (p1[i] < p1[i + 1]) {

temp = p1[i + 1];

p1[i + 1] = p1[i];

p1[i] = temp;

}

}

}

}

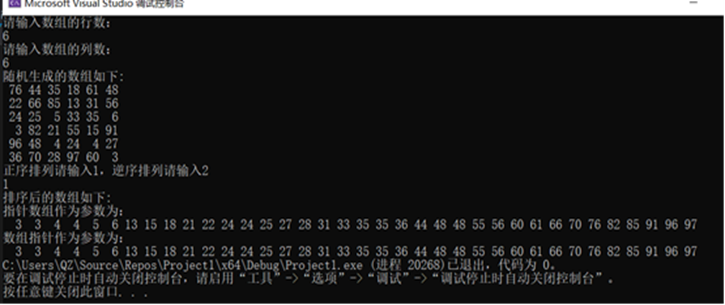
for (int i = 0; i < row \* column; i++) {//输出排列之后的数

printf("%3d", p1[i]);

}

free(p1);

**实验结果与分析**



分析：实验结果符合要求

**暨南大学本科实验报告专用纸(附页)**