《计算机程序设计》作业 **№-09及第8次上机**

# 作业内容要点： 指针

【 姓名 马宇骁 学号 PB19151769 】

**【要求】本次作业程序要求用指针实现**

1. **在计算机上编程程序，加上必要的注释。**
2. **上机实验，经助教检查通过后，复制源码并记录实验结果，完成报告。**
3. **实验报告：记录调试及改错过程；****知识点或方法技巧的收获心得.**

##### 1 、将nxn方阵中的前4个最小元素放置到方阵的四个角。【实验要点： 二维数组、指针、函数】

**说明：**

**1) 设计一个函数，实现将任意的nxn方阵（n>=3）的前四个最小元素放置到方阵四个角的位置（顺序为：左上、右上、左下、右下）；元素集合不变，四角之外的其余元素位置变动不作限制。**

**2) 在主函数中输入，输入内容第一行为n， 第二行为n2个整数即方阵的元素。调用上述函数，然后输出处理后的方阵。**

**输入样例：**

**3**

**4 2 3 1 5 7 6 8 9**

**输出样例：**

**1 6 2**

**9 5 7**

**3 8 4**

1. 【源码】

#include<stdio.h>

#include <string.h>

#include <stdlib.h>

int \*form(int \*m,int n){

int tmp;

int a[n][n];

int siz = n\*n;

int aa[siz];

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

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

a[j][k] = \*(m+j\*n+k);

aa[n\*j+k] = a[j][k];

}

}

//bubble

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

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

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

tmp = aa[j];

aa[j] = aa[j+1];

aa[j+1] = tmp;

}

}

}

int min[4] = {aa[0],aa[1],aa[2],aa[3]};

int temp[4] = {a[0][0],a[0][n-1],a[n-1][0],a[n-1][n-1]};

for(int i=0;i<4;i++){

if(min[i]==a[0][0] || min[i]==a[0][n-1] || min[i]==a[n-1][0] || min[i]==a[n-1][n-1]){

min[i] = 999;

}

if(min[0]==temp[i] || min[1]==temp[i] || min[2]==temp[i] || min[3]==temp[i]){

temp[i] = 999;

}

}

// find location

int loc[4][2];

for(int i=0;i<4;i++){

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

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

if(min[i]==999){

loc[i][0] = 999;

loc[i][1] = 999;

}

else if(a[j][k] == min[i]){

loc[i][0] = j;

loc[i][1] = k;

}

}

}

}

int l=0;

for(int i=0;i<4;i++){

if(loc[i][0]!=999){

while(l<4){

if(temp[l]!=999){

a[loc[i][0]][loc[i][1]] = temp[l];

l++;

break;

}

else l++;

}

}

}

a[0][0] = aa[0];

a[0][n-1] = aa[1];

a[n-1][0] = aa[2];

a[n-1][n-1] = aa[3];

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

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

aa[n\*j+k] = a[j][k];

}

}

return aa;

}

int main(){

int n;

scanf("%d",&n);

int m[n][n];

int a[n\*n];

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

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

scanf("%d",&a[i\*n+j]);

m[i][j] = a[i\*n+j];

printf("%d\t",m[i][j]);

}

printf("\n");

}

printf("\n");

int \*b;

b = form(a,n);

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

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

printf("%d\t",b[j\*n+k]);

}

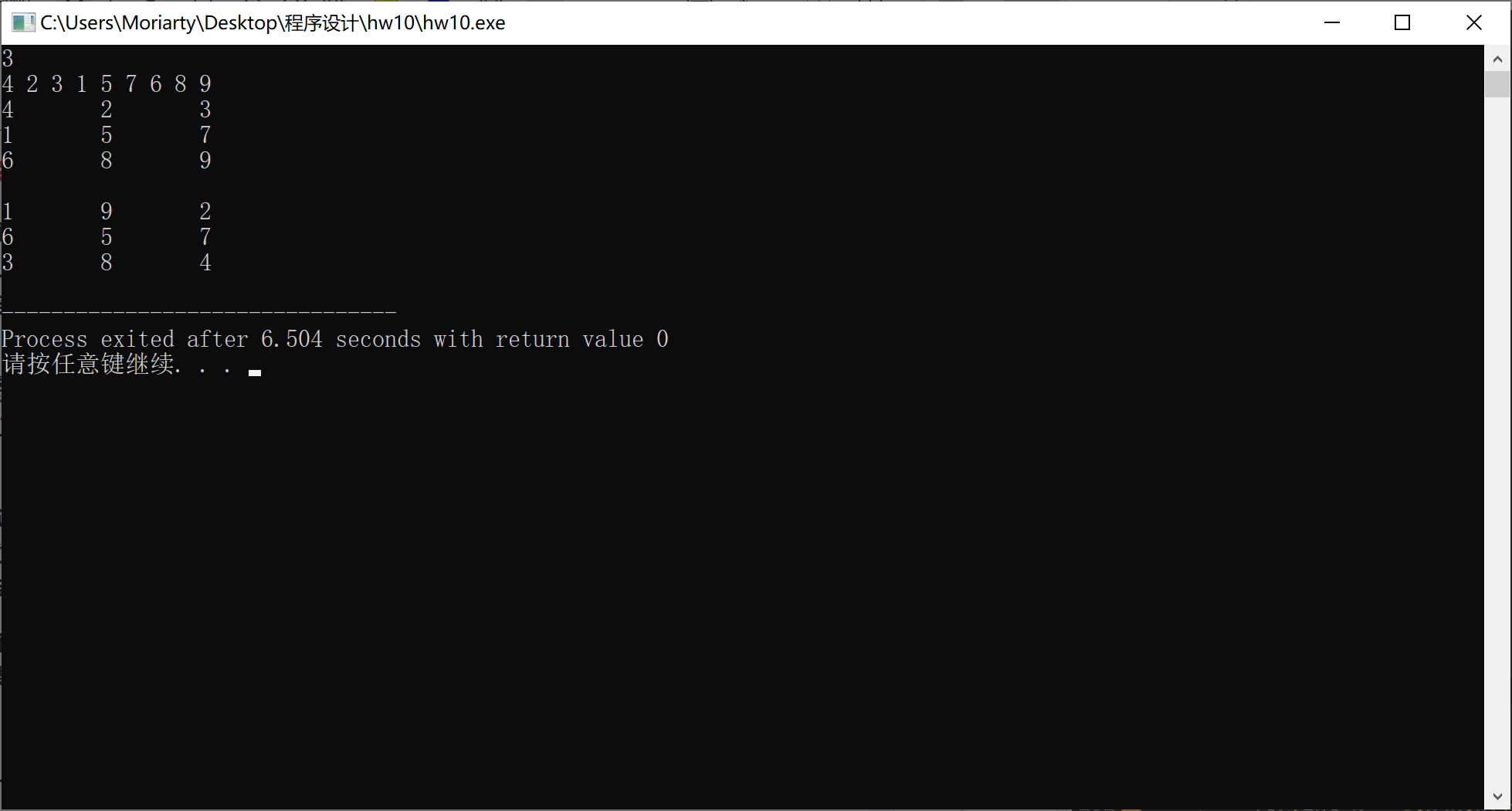
printf("\n");

}

return 0;

}

1. 【运行结果】



1. 【实验报告】

数组的问题，通过记录四个角落的数字，判断是否为最小值，再判断最小值是否在四个角落，然后记录最小值不在角落的位置用四个角落不是最小值的数字去替换，最后用最小值覆盖四个角落完成最小程度保留原有数组的调整。

**2 、 编程实现求方差的命令行。【实验要点：命令行参数，指针】**

**说明：**

**1) 通过命令行参数输入若干实数，把这些实数的字符串形式转换为double类型的数据，并根据参数个数将这些数据存储于动态分配的内存中。**

**2) 计算这些数据的方差并输出。**

**3) 提示：可使用库函数pow，atof。原型如下：**

#include <math.h>

double pow(double x, double y); //幂函数，返回xy

#include <string.h>

double atof(const char \*str); //将字符串str转化为浮点数并返回该浮点数

**输入样例：(假定可执行程序的名称是fangcha.exe )**

***fangcha.exe 12 3 5 9***

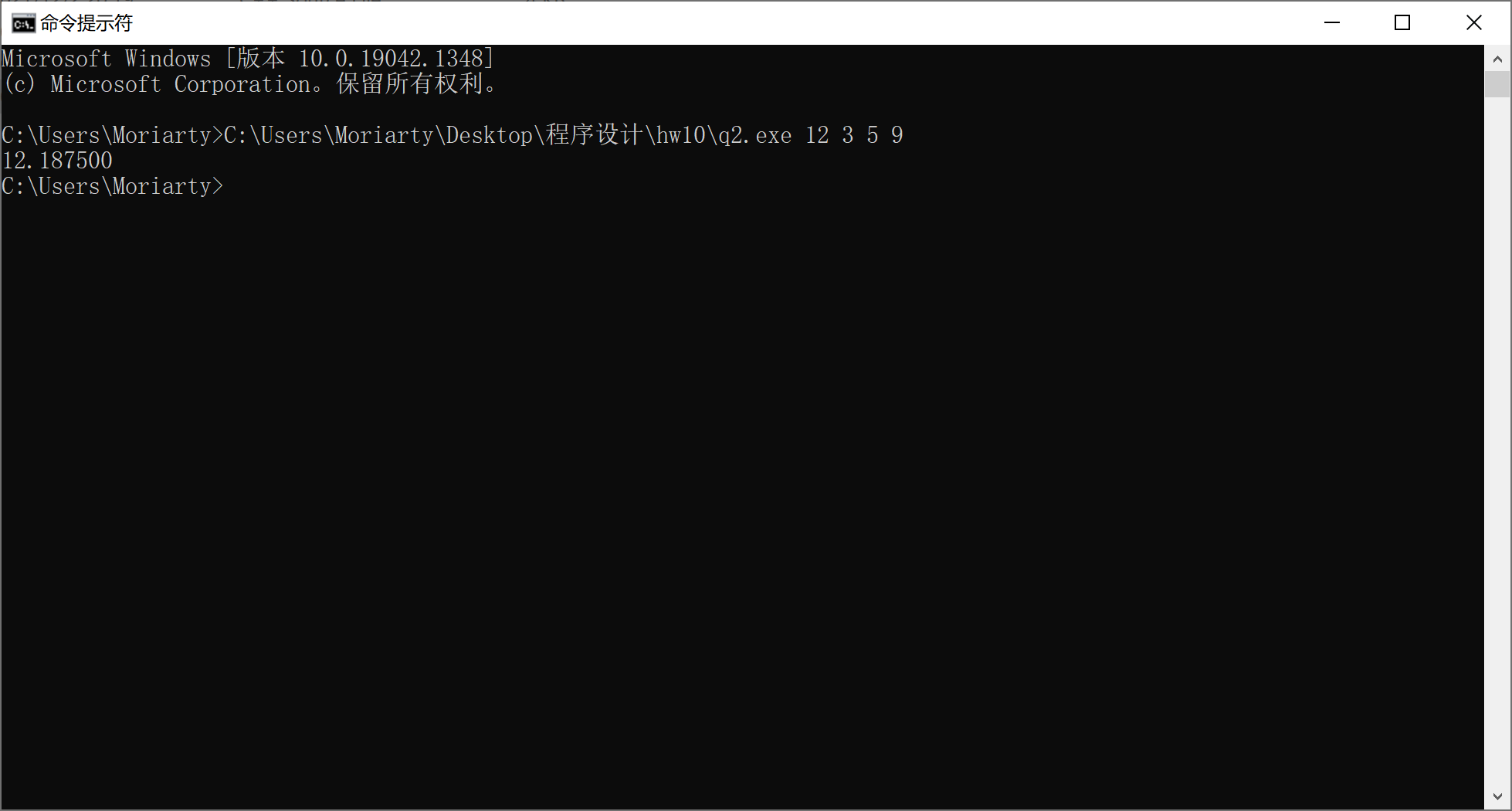
**输出样例：**

***12.187500***

1. 【源码】

* #include<stdio.h>
* #include <math.h>
* #include <stdlib.h>
* #include <string.h>
* int main(int argc,char \*argv[])
* {
* double sum=0,avg,var;
* int n=argc-1;
* double \*p = (double\*)malloc(n\*sizeof(double));
* for(int i=1; i<argc; i++)
* {
* p[i-1] = (double)atof(argv[i]);
* sum += p[i-1];
* }
* avg = sum/n;
* for (int j=0; j<n; j++)
* {
* var += pow(p[j]-avg,2)/n;//求方差
* }
* printf("%lf",var);
* free(p);
* }

1. 【运行结果】

* 

1. 【实验报告】

这个比较简单，求方差并用cmd打开读取。

##### 3. 将一个字符串循环右移n位。用指针实现。【实验要点：字符串，指针】

**说明：**

**1) 输入字符串，长度限50字符以内，若超出50则只取前50个；**

**2) n是用户输入的一个正整数；**

**3) 循环右移一位是指将最后一个字符移到字符串最左面，其余字符均向右移动一个位置。循环右移n位是将上述循环右移一位的动作重复n次。(怎样减少循环次数从而提高程序效率？ 是否可以不用循环嵌套？)**

**4) 编写函数完成循环右移功能，在主函数中输入字符串和整数n，调用循环右移函数，输出右移之后的字符串。**

**输入样例：**

**abcdefghijklmn**

**10**

**输出样例：**

**efghijklmnabcd**

1. 【源码】

void reserve(char \*p, char \*q)

{

while(p < q)

{

\*p ^= \*q; //利用异或运算得到两字符交换

\*q ^= \*p;

\*p ^= \*q;

p++;

q--;

}

}

void loopMove(char \*str, int steps)

{

char \*p = str;

int len = strlen(str);

char \*q = p + len - 1 - steps;

reserve(p, q); //先翻转前一段,这里调用函数只是指针的拷贝，没有返回指针，函数在用那个作用域中的指针改变了字符串数组中的值并没有返回来，所以这语句之后p=str q=str+len-1-steps 没有改变

p = q + 1;

q = str + len - 1;

reserve(p, q); //翻转后一段，同理，函数返回之后，p,q 还是 上两句改变之后的值

p = str;

reserve(p, q); //翻转整个字符串，同理，函数返回之后，p,q 还是 上句改变之后的值

}

int main()

{

char string[50];

gets(string);

int steps = 0;

printf("string: %s\n", string);

printf("input step: ");

scanf("%d", &steps);

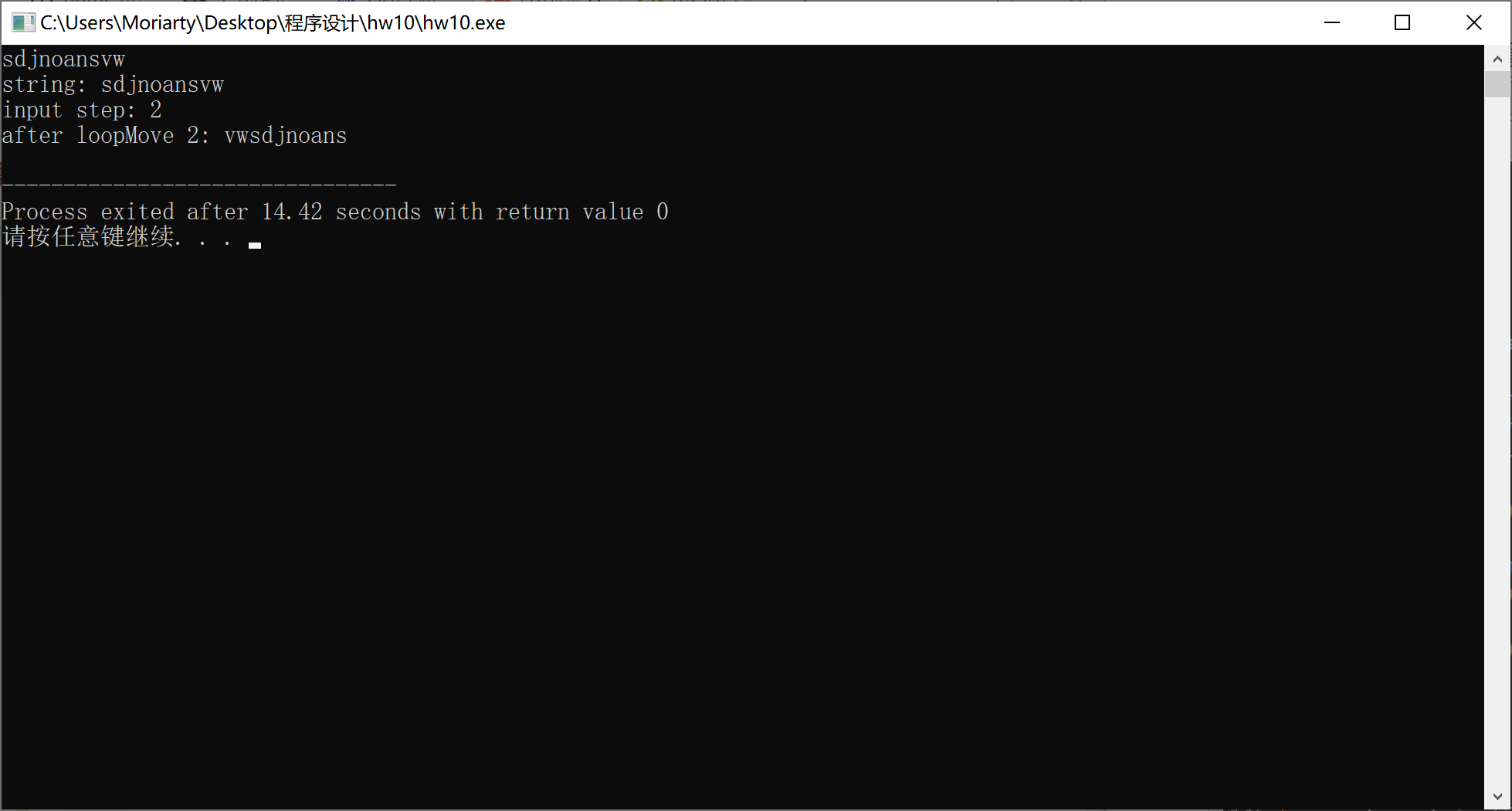
loopMove(string,steps);

printf("after loopMove %d: %s\n", steps, string);

return 0;

}

1. 【运行结果】



1. 【实验报告】

* 通过异或判断进行倒序的函数，向右移动n位就是把前n位倒序，再把剩下的倒序，最后把整个倒序就能得到。