**《程序设计基础课程设计》第4周实验报告**

班级：1803012

姓名：杨煜

学号：18030100204

所做题目：一、算法设计—— 二选一

**第6题**

**1.原始题目及要求；**

**（1）快速排序算法**

要求：编写一个程序，对用户输入的若干整数，采用快速排序算法，完成从小到大的排序。

**2.题目的分析**

（1）涉及知识点

数组、快速排序算法等

（2）题目功能理解

快速排序采用分治的思想有效的降低了排序的平均时间复杂度，具体策略为将原序列划分为大于某值和小于某值的两部分。之后再对这两部分再执行快速排序。当排序规模小于10时，快速排序的效果并不如简单的插入排序，故此时选用插入排序。

**3. 题目的总体设计：设计思路、算法描述**

（1）程序模块：

（A）交换模块

交换两个个指针指向的对象的内容。

（B）插入排序模块

插入排序。

（C）划分模块

划分数组。

（D）快速排序模块

快速排序。

（2）模块调用关系

快速排序模块调用划分模块，插入排序模块，快速排序模块；划分模块调用交换模块。

（3）输入输出数据说明

输入数据：无。

输出数据：排序结果。

（4）总体流程

1）获取一段随机数组成的整数序列，转2。

2）输出原序列，转3.

4）使用快速排序排序原序列，转4。

5）输出排序后的原序列，并输出结果。若已执行足够次数，则转5，否则转1。

6）结束，返回。

**4. 各功能模块/函数的设计说明：**

（1）void swap( int \*a, int \*b)

函数名称: swap

函数功能: 交换两个个指针指向的对象的内容

参数: int \*a( 操作数1 ), int \*b( 操作数2 )

返回值: void

执行流程：

1）交换指针中对象，返回。

可能结果：

1）交换完成

（2）void insertSort( int nums[], int start, int end)

函数名称: insertSort

函数功能: 插入排序

参数: int nums[]( 待排序数组首地址 ), int start( 开始排序的位置 ),

int end( 结束排序的位置加1 )

返回值: void

执行流程：

1）插入排序，返回。

可能结果：

1）排序完成

（3）int partiton( int nums[], int start, int end )

函数名称: partiton

函数功能: 划分数组

参数: int nums[]( 待划分数组首地址 ), int start( 开始划分的位置 ),

int end( 结束划分的位置加1 )

返回值: int( 表示大小位于中间的数的位置 )

执行流程：

1）使得第一个元素是左中右中大小中等的，转2。

2）执行划分，转3。

3）把首元素填入对应位置，转4.

4）返回现在首元素的位置。

可能结果：

1）完成划分

（4）void quick\_sort( int nums[], int start, int end )

函数名称: quick\_sort

函数功能: 快速排序

参数: int nums[]( 待排序数组首地址 ), int start( 开始排序的位置 ),

int end( 结束排序的位置加1 )

返回值: void

执行流程：

1）若待排序列大小小于1则返回，否则转2。

2）若待排序列大小小于10转3，否则转4。

3）执行插入排序，返回。

4）执行划分，分别用快速排序排序前、后半段，返回。

可能结果：

1）完成排序

**5. 源程序：**

（1）quick\_sort\_test.c

#include<stdio.h>

#include<time.h>

#include<stdlib.h>

#include"quick\_sort.h"

int main(){

srand( time( NULL ) );

int org\_num[20];

for( int i=0; i<5; ++i ){

for( int j=0; j<20; ++j ){

org\_num[j] = rand();

}// 获取20个随机数

printf("未排序的序列为：\n");

for( int j=0; j<20; ++j ){

printf("%d ", org\_num[j] );

}// 显示未排序的序列

printf("\n");

quick\_sort( org\_num, 0, 20 );

printf("排序后的序列为\n");

for( int j=0; j<20; ++j ){

printf("%d ", org\_num[j] );

}// 显示已排序的序列

printf("\n\n");

}

return 0;

}

（2）quick\_sort.h

#ifndef \_\_QUICK\_SORT\_H

#define \_\_QUICK\_SORT\_H

#include<malloc.h>

/\*

\* 函数名称：swap

\* 函数功能：交换两个个指针指向的对象的内容

\* 参数: int \*a( 操作数1 ), int \*b( 操作数2 )

\* 返回值：void

\*/

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

int temp = \*a;

\*a=\*b;

\*b=temp;

return;

}

/\*

\* 函数名称：insertSort

\* 函数功能：插入排序

\* 参数: int nums[]( 待排序数组首地址 ), int start( 开始排序的位置 ),

\* int end( 结束排序的位置加1 )

\* 返回值：void

\*/

void insertSort( int nums[], int start, int end){

for( int i=start+1; i<end; i++ ){

if( nums[i]<nums[i-1] ){

int temp = nums[i];

int j;

for( j=i-1; j>=0; j-- ){

if( nums[j]>temp ){

nums[j+1]=nums[j];

}

else{

break;

}

}

nums[j+1] = temp;

}

}

}

/\*

\* 函数名称：partiton

\* 函数功能：划分数组

\* 参数: int nums[]( 待划分数组首地址 ), int start( 开始划分的位置 ),

\* int end( 结束划分的位置加1 )

\* 返回值：int( 表示大小位于中间的数的位置 )

\*/

int partiton( int nums[], int start, int end ){

int mid= (start+end-1)/2;

if( nums[start]>nums[end-1] ){

swap( &nums[start], &nums[end-1] );

}

if( nums[mid]>nums[end-1] ){

swap( &nums[end-1], &nums[mid] );

}

if( nums[mid]>nums[start] ){

swap( &nums[start], &nums[mid] );

}// 使得第一个元素是左中右中大小中等的, 这样有助于提升性能

int flag = nums[start];

int left=start;

int right=end-1;

while( left<right ){

while( left<right && nums[right]>=flag ){ right--; }

nums[left]=nums[right];

while( left<right && nums[left]<=flag ){ left++; }

nums[right]=nums[left];

}//执行划分

nums[left]=flag;//把首元素填入对应位置

return left;

}

/\*

\* 函数名称：quick\_sort

\* 函数功能：快速排序

\* 参数: int nums[]( 待排序数组首地址 ), int start( 开始排序的位置 ),

\* int end( 结束排序的位置加1 )

\* 返回值：void

\*/

void quick\_sort( int nums[], int start, int end ){

if( start<end-1){

if( end-start>10 ){

// 若排序范围大于10, 则选择快速排序

int mid = partiton( nums, start, end );//划分待排序列

quick\_sort( nums, start, mid);// 排序划分后的左半部分

quick\_sort( nums, mid+1, end);// 排序划分后的右半部分

}

else{

// 否则选择插入排序

insertSort( nums, start, end);

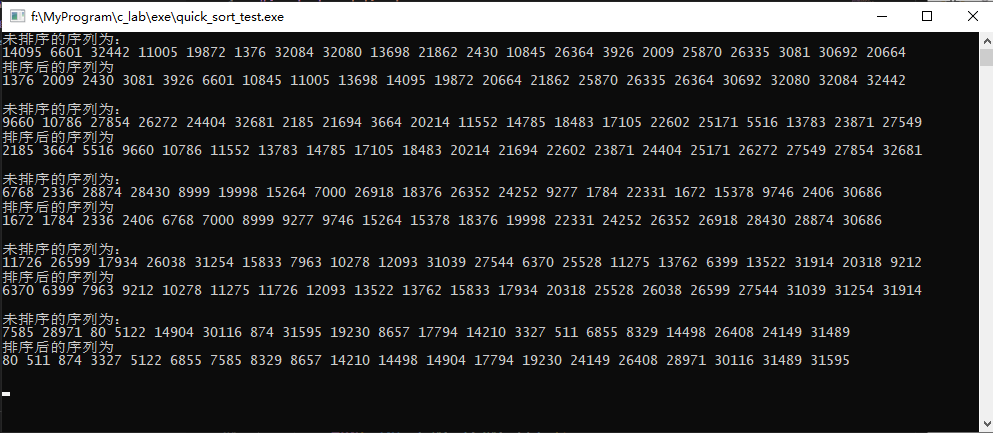
}

}

}

#endif

**6. 测试数据（输入、输出）：**



**7. 小结：**

本次实验圆满完成，实验结果符合实验要求。

实验过程中，我复习到了c语言中数组、快速排序算法等重要知识，收货颇丰。