**Project 3: Multithreaded Sorting Application**

**& Fork-Join Sorting Application**

Project 3-1: Multithreaded Sorting Application

实现细节：

一、准备

1、全局变量

两个全局变量，一个存原数组，一个存排序后的数组

2、index类型

声明将排序的开始位置和结束位置传递给线程而生命的index类型

3、quicksort()

依照快速排序算法写函数quicksort()

4、sort()

写快速排序函数的包裹函数sort()，作为快速排序线程的“run”，即写成一个函数指针

5、merge()

依照归并排序算法写函数merge()，直接作为归并线程的“run”,也是一个函数指针

二、main()

1、输入数组大小

2、希望数组大小至少为2，否则提示非法并退出

3、循环输入数组

4、实例化3个index对象，存在一个index类型数组中，为三个排序线程提供排序的开始位置和结束位置。

5、创建与初始化线程

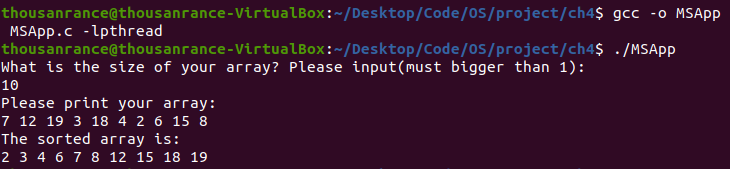
6、依次运行线程

7、输出排序结果

8、释放空间

9、结束返回

运行结果：



Project 3-2: Fork-Join Sorting Application

（一）QuickSort实现细节：

一、配制环境

按照网上找到的资料配制java环境：<https://www.linuxidc.com/Linux/2018-10/155020.htm>

二、类与成员函数

1、代码框架

依照课本Fork Join in Java内容搭建代码框架ForkJoinTask\_Quick类，是RecursiveAction的派生类。

2、设置“small”的标准THRESHOLD。

3、全局变量

设置全局变量：数组大小、数组、起始位置、终止位置。

4、类的构造函数

类的构造函数用于初始化参数。

5、compute()

重载的compute()函数，相当于线程的run函数。当问题规模小于THRESHOLD时，直接调用选择排序算法；当问题规模大于THRESHOLD时，使用快排算法算出pivot的位置，利用pivot将问题分为leftTask与rightTask，将其实例化为新的ForkJoinTask\_Quick类对象，为其创建新线程，并执行。最后将左任务与右任务merge起来，返回。

6、quicksort()

依照快速排序算法写quicksort()函数，但是不需要在最后递归，而是返回pivot的值。

7、selectionsort()

依照选择排序算法写selectionsort()函数，用于问题规模划分至小于THRESHOLD的时候。

8、merge()

依照归并排序算法写函数merge()，用于归并已完成的左任务和右任务。

三、主函数

1、实例化ForkJoinPool类线程池对象。

2、利用随机数生成初始数组。

3、实例化ForkJoinTask\_Quick对象task，是用于排序的任务。

4、将其加入线程池开始运行。

5、最后将排好序的数组输出。

（二）MergeSort实现细节：

基本与QuickSort相同，只有划分左任务和右任务时，划分的标准为mergesort()函数返回，实际上就是对半划分。

（三）运行结果：

