Sorting Threads Project:

Overview:

Write a multithreaded sorting program in Java which uses the merge sort algorithm. The basic steps of merge sort are: 1) divide a collection of items into two lists of equal size, 2) use merge sort to separately sort each of the two lists, and 3) combine the two sorted lists into one sorted list. Of course, if the collection of items is just a single item then merge sort doesn't need to perform the three steps, just return the single item. Normally, without using threads, merge sort makes a recursive call to itself to sort the first list and then another recursive call to itself to sort the second list. Instead, for this assignment, merge sort will use sorting threads to sort the list.

A sorting thread object receives the list to sort as an array via its constructor. The sorting thread class uses Java Generics in order to sort any type of object which implements the Comparable interface. For example, assume the name of the sorting thread class is Merge Sort and the generic datatype name is Any Type. The sorting thread class has at the beginning of its declaration:

... class MergeSort<AnyType extends Comparable<? super AnyType>>...and the declaration of the array of objects v to sort would look like: AnyType[] v;

A sorting thread object follows the steps of merge sort as follows:

1) divide the array of items it received into two arrays of equal size,

2) create two sorting thread objects where each thread sorts one of the equally divided arrays produced in the previous step, and

3) combine the two sorted arrays (generated by the two sorting threads from the previous step) into one sorted array. A sorting thread needs to know when each of its two sorting threads completed its execution. This is done using the join()method on each of the two sorting threads before the two sorted arrays are merged into one sorted array. Of course, if the array of items is just a single item then merge sort doesn't need to perform the three steps, just return the single item.

Design

1.Create a driver class and make the name of the driver class Assignment1containingonly one method: public static void main(String args[]).The main method itself is fairly short containing code to do the following:

a. Create a 100 element array of Integers.

b. Fill each element of the array with a randomly generated integer in the range of 1 to 10,000.

c. Print the Integer array in its original unsorted order.

d. Create a sorting thread object passing the Integer array into the sorting thread object via its constructor.

e. Start the execution of the sorting thread.

f. Use the join method on the sorting thread to wait until it completes its execution.

g. Print the Integer array in its sorted order.

2.You must declare private the data members in every class you create.

3.Tip: Make your program as modular as possible, not placing all your code in one .java file. You can create as many classes as you need in addition to the classes described above. Methods being reasonably small follow the guidance that "A function do one thing, and do it well. "You will lose a lot of points for code readability if you don't make your program as modular as possible. But, do not go overboard on creating classes and methods. Your common sense guides your creation of classes and methods. Though, in this assignment you probably won't need to create any additional classes.

4.Do NOT use your own **packages** in your program. If you see the keyword package on the top line of any of your .java files then you created a package. Create every .java file in the **src** folder of your Eclipse project.

5.Do NOT use any graphical user interface code in your program!

6.Do NOT type any comments in your program. If you do a good job of programming by following the advice in number 3 above then it will be easy for me to determine the task of your code.

Specification:

1.If your code does not implement the task described in this assignment then it will not be accepted.

2. Don't submit if your program does not compile successfully.

3. If your program produces runtime errors which prevents me from determining if your code works properly then it will not be accepted.

If the program compiles successfully and executes without significant runtime errors then see the next steps:

Followed proper submission instructions:

1.Was the file submitted a zip file.

2.The zip file has the correct filename.

3.The contents of the zip file are in the correct format.

4.The keyword package does not appear at the top of any of the .java files.

Program execution,

- Program input, the program properly generates input for the assignment.
- Program output, the program produces the correct results for the input.

Code implementation,

- The driver file has the correct filename, Assignment1.javaand contains only the method main performing the exact tasks as described in the assignment description.
- The code performs all the tasks as described in the assignment description.
- The code is free from logical errors

Code readability,

- Good variable, method, and class names.
- Variables, classes, and methods that have a single small purpose.
- Consistent indentation and formatting style.
- Reduction of the nesting level in code.