Fibonacci. Quick sort & Merge sort Gary Cullen

# Practical Lab 04 – Fibonacci, Quick sort & Merge sort (Due 8/3/21)

1. Write a piece of Java code that prints out the first 20 numbers in the Fibonacci sequence. Provide comments where appropriate to explain your code. **Complete in Q1.java**

1. Outline the purpose of Fibonacci numbers and their application in Trend Analysis.

**The purpose of the Fibonacci numbers is to be used as guides in trade. Their application in Trend Analysis is used to draw support lines, identify resistance levels, place stop-loss orders and set target prices.**

1. Quick sort is said to be in the order O (n log n), in the general case, but in the worst case it is said to be O(n2).
   * + Describe this problem, how does this worst case scenario occur? **The problem is when the data is already sorted. The worst case scenario occurs if the rightmost element is selected as the pivot then every element will be compared with every other element in the array.**
     + What fix is available to overcome this problem?  
       **The “Median of 3” Partitioning.**
     + How would you implement this fix (show code)?   
       **Added into Quicksort.java but not complete.**
     + What added benefit, to the overall running of the algorithm does this fix bring? **It helps avoiding the worst case time complexity of O(n2).**
2. Using the code from earlier practical’s, populate 3 random, sorted and inversely sorted arrays with 10,000 integer values. **Completed in Quicksort.java**
3. Implement the Merge sort algorithm – using the code on the vle. **Completed in Mergesort.java**
4. Implement the Quick Sort algorithm – using the code on the vle. **Completed in Quicksort.java**
5. Implement the StopWatch class you had from the practical 03 and test the running time of the standard QuickSort against a sorted array of 10000 integer values. **Completed in Quicksort.java, Mergesort.java, StopWatch.java.**
6. Implement the fix to overcome the problem mentioned in section 3, above. **Added into Quicksort.java but not complete.**
7. Implement the StopWatch class to evaluate your fix, what impact did it have? **Added into Quicksort.java but not complete.**