CSCI 1112 Algorithms and Data Structures

Lab 2 – Recursion

Part 1: Merge Sort (8 points)

a) Download MergeSort.java and implement the *merge*() method used in *mergeSort*(). Count the number of element-wise comparisons and the number of copy operations. Report the performance in the table below and compare with the performance of insertion sort.

b) Merge sort can be improved if the algorithm stops before the merge step when the elements of the array are already sorted. If the right-most element in the first subarray is smaller than or equal to the left-most element in the second subarray, then the array is already sorted and there is no need to merge. Implement this step and report the change in counts.

Algorithm	**N=10**		**N=100**		**N=1000**	
	compare	**Swap/copy**	**compare**	**Swap/copy**	**compare**	**Swap/copy**
Insertion sort						
Merge sort [1.a]						
Merge sort [1.b]						

Part 2: Insertion sort (3 points)

a) Implement a recursive version of insertion sort.

Part 3: Find the maximum (4 points)

a) Implement a recursive method for finding the maximum value in an array of integers