CRT Chapter 12

1. a) Start: 4, 6, 2, 10, 9
   First iteration: **2**, 6, 4, 10, 9. - 4 is swapped with 2, the smallest number
   Second loop iteration : 2, **4**, 6, 10, 9 - 6 swaps with 4
   Third loop iteration :2, 4, **6**, 10, 9 - 6 stays in the same spot because it is correctly ordered
   Fourth iteration : 2, 4, 6, **9,** 10 - the 10 swapped with the 9
   End : 2, 4, 6, 9, 10 - the iterations have finished.

   b) Start: 4, 6, 2, 10, 9
   First loop iteration : **4, 6**, 2, 10, 9 - relative to 6, 4 is in the correct spot
   Second loop iteration : **2**, 4, 6, 10, 9 - 2 should be before the 4 so it is inserted there.
   Third loop iteration : 2, 4, 6, **10**, 9 - 6 is less than 10, so the 10 is in the correct order relative to the 6.
   Forth loop iteration : 2, 4, 6, 9, 10 - The 9 and 10 swap positions as 9 is smaller than 10.
   End - 2, 4, 6, 9 10 - the iterations have finished.

2. To be able to use a binary search a list or array must be sorted into ascending order.

3. a)
   Output is:
   Starting 13
   Starting 4
   Starting 1
   Starting 0
   Middle 0
   Middle 1
   Middle 4
   Middle 13
   b)
   Starting 3
   Starting 1
   Starting 0
   Middle 0
   Middle 1
   Middle 3
   c)
   Starting 0

7. Merge Sort is stable. Selection sort is not stable.
E.g. If we sort an array including Ann(20), Jon(19), and Tom(19). It can become Tom(19), Jon(19), Ann(20) after selection sort which would flip Jon and Tom making this not a stable sorting method.

8.

a) False. Sorting allows the program to either sort high to low, or low to high, in any order, it depends on what the program's comparison is doing.
b) True. The faster an algorithm can sort a set amount of data the more efficient it is.
c) False. Insertion sort is faster so it is more efficient.

e) True. it is the most efficient sorting algorithm and the time taken doesn't scale exponentially as number of terms increase.

f) False. A binary search is used to search a sorted list, its not the algorithm that sorts.

g) False. Binary search splits the list in half, and examines the middle term first. Depending on that, if the goal is mid term the program ends, otherwise it continues splitting into halves, and does that until the program finds the goal value.

h) False. Efficiency is dependent on how fast the algorithm works and runs, not dependent on how many lines of code. It depends on the time complexity of the code.

i) True. If there is not a base case which stops the recursion at some point, the code will keep recursively running indefinitely.

k) False. This description is describing merge sort, not insertion sort. Insertion sort builds the list working on one item at a time, not multiple at a time.