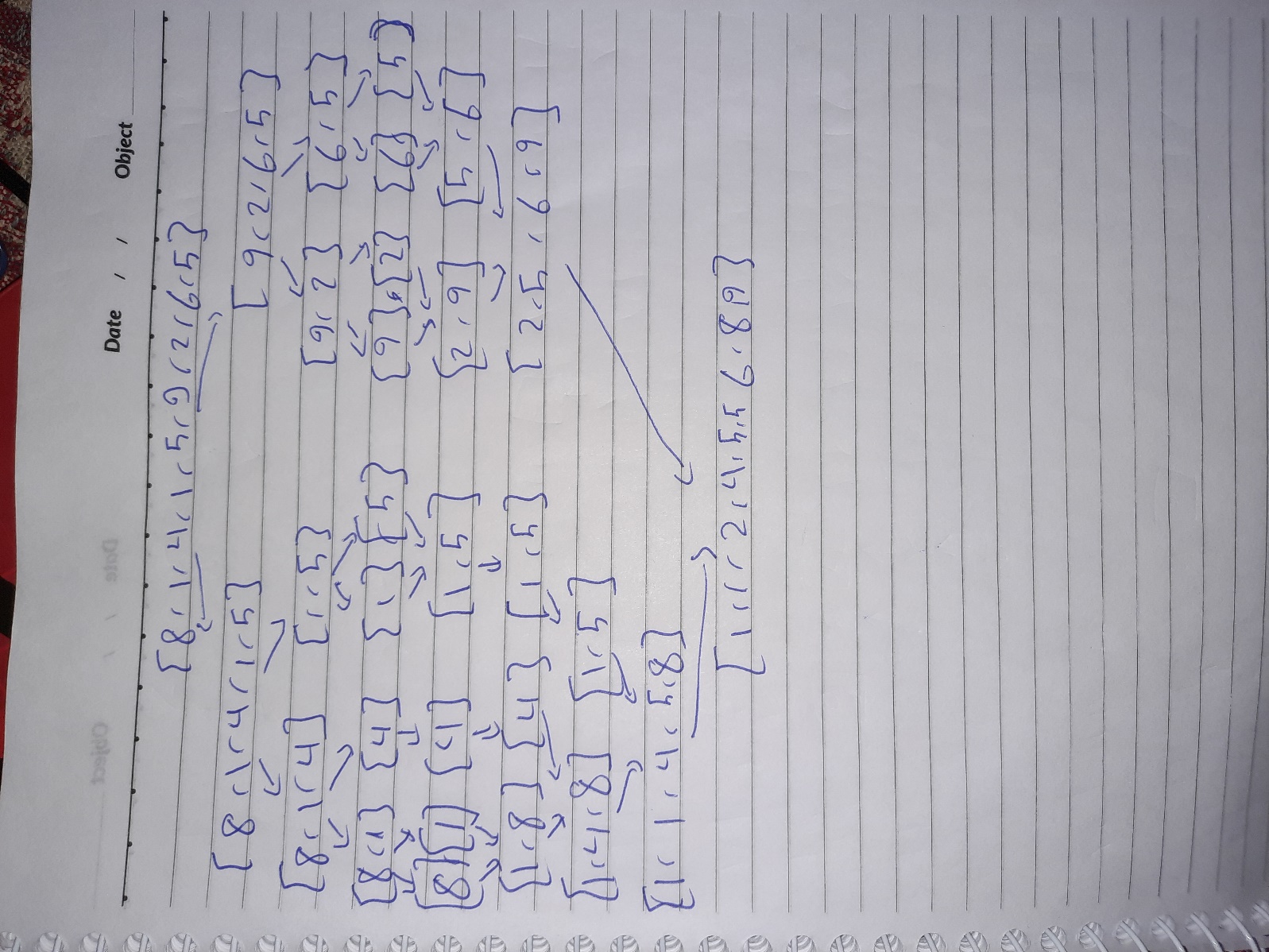
**BY: Fadi Alahmad Alomar 120180049**

# Question 1  
a) Insertion sort:  
[8,1,4,1,5,9,2,6,5]  
swap(arr[0],arr[1])  
[1,8,4,1,5,9,2,6,5]  
swap(arr[1],arr[2])  
[1,4,1,8,5,9,2,6,5]  
swap(arr[0],arr[1])  
[1,1,4,8,5,9,2,6,5]  
swap(arr[4],arr[3])  
[1,1,4,5,8,9,2,6,5]  
swap(arr[6],arr[5])  
[1,1,4,5,8,2,9,6,5]  
swap(arr[5],arr[4])  
[1,1,4,5,2,8,9,6,5]  
swap(arr[4],arr[3])  
[1,1,4,2,5,8,9,6,5]  
swap(arr[3],arr[2])  
[1,1,2,4,5,8,9,6,5]  
swap(arr[7],arr[6])  
[1,1,4,2,5,8,6,9,5]  
swap(arr[6],arr[5])  
[1,1,4,2,5,6,8,9,5]  
swap(arr[8],arr[7])  
[1,1,4,2,5,6,8,5,9]  
swap(arr[7],arr[6])  
[1,1,4,2,5,6,5,8,9]  
swap(arr[6],arr[5])  
[1,1,4,2,5,5,6,8,9]  
  
b) Heap sort:  
max heapify arr  
[9,6,8,5,5,4,2,1,1]  
swap(arr[0],arr[-1])  
[1,6,8,5,5,4,2,1,9]  
heapify arr[:-1]  
[8,6,4,5,5,1,2,1,9]  
swap(arr[0],arr[-2])  
[1,6,4,5,5,1,2,8,9]  
heapify arr[:-2]  
[6,5,4,1,5,1,2,8,9]  
swap(arr[0],arr[-3])  
[2,5,4,1,5,1,6,8,9]  
heapify arr[:-3]  
[5,5,4,1,2,1,6,8,9]  
swap(arr[0],arr[-4])  
[1,5,4,1,2,5,6,8,9]  
heapify arr[:-4]  
[5,2,4,1,1,5,6,8,9]  
swap(arr[0],arr[-5])  
[1,2,4,1,5,5,6,8,9]  
heapify arr[:-5]  
[4,2,1,1,5,5,6,8,9]  
swap(arr[0],arr[-6])  
[1,2,1,4,5,5,6,8,9]  
heapify arr[:-6]  
[2,1,1,4,5,5,6,8,9]  
swap(arr[0],arr[-7])  
[1,1,2,4,5,5,6,8,9]

c) Merge sort  
  
d) Quick sort  
[8, 1, 4, 1, 5, 9, 2, 6, 5] p= 5 l= 0 r= 8  
[8, 1, 4, 1, 5, 9, 2, 6, 5] array part the algorithm is working on  
[5, 1, 4, 1, 2, 5, 9, 6, 8] array after modification  
  
[5, 1, 4, 1, 2, 5, 9, 6, 8] p= 6 l= 6 r= 8  
[9, 6, 8] array part the algorithm is working on  
[5, 1, 4, 1, 2, 5, 6, 9, 8]  
  
[5, 1, 4, 1, 2, 5, 6, 9, 8] p= 9 l= 7 r= 8  
[9, 8] array part the algorithm is working on  
[5, 1, 4, 1, 2, 5, 6, 8, 9] array after modification  
  
[5, 1, 4, 1, 2, 5, 6, 8, 9] p= 4 l= 0 r= 4  
[5, 1, 4, 1, 2] array part the algorithm is working on  
[2, 1, 1, 4, 5, 5, 6, 8, 9] array after modification  
  
[2, 1, 1, 4, 5, 5, 6, 8, 9] p= 1 l= 0 r= 2  
[2, 1, 1] array part the algorithm is working on  
[1, 1, 2, 4, 5, 5, 6, 8, 9] array after modification  
  
[1, 1, 2, 4, 5, 5, 6, 8, 9] p= 1 l= 0 r= 1  
[1, 1] array part the algorithm is working on  
[1, 1, 2, 4, 5, 5, 6, 8, 9] array after modification  
  
[1, 1, 2, 4, 5, 5, 6, 8, 9] Final result

# Question 3  
transfer the two strings into arrays then sort them using heap sort, each character has an ascii value thus it can be treated as a number when comparing it to other characters  
going through both the arrays and if they are the same then the words are an anagram if the arrays are not the same then the words are not an anagram

# Question 5