Problem 1. We want to sort the following list, $[70, 50, 110, 60, 90, 40, 100, 80]$, of numbers using heap sort algorithm that we covered in class. It will involve two steps, first, building a max heap, second, sorting, by extracting the root and removing any max-heap violations. Apply this algorithm to the given array and answer the following questions.

1. Show the max-heap as a tree. Show the array corresponding to this max-heap. Exactly how many comparisons did it take to build the max-heap?
2. Starting with the max-heap built in the previous step, show the array after each max-heapify. How many comparisons does each max-heapify use? What is the total number of comparisons for just this part, excluding the number of comparisons in Part(a).

Problem 2. In class we counted the number of comparisons to sort an arbitrary array of size, n using heap sort. Solve for the exact number of exchanges that take place in sorting this array using heap sort algorithm. Show your work for exact analysis in the worst case.

Problem 3. Design an efficient algorithm to construct a single heap that contains all the elements of two given heaps of sizes n and m, respectively. What is the asymptotic runtime of your algorithm in the worst case? You may explain in English or in pseudo-code. Explain the runtime.