**Elementary Sorts:**

**Which of the following arrays would insertion sort process the fastest?**(a) {1, 2, 3, 4, 7, 5, 6}  
(b) {2, 3, 4, 6, 1, 5, 7}  
(c) {6, 7, 5, 4, 3, 2, 1}

(d) They will be processed the same

**Which sorting algorithm will take least time when all elements of input array are identical? Consider typical implementations of sorting algorithms.**

(a) Selection Sort

(b) Merge Sort

(c) Insertion Sort

(d) Shell Sort

**Merge Sort :**

**What is the auxiliary space complexity of standard merge sort for arrays?**(a) O(1)  
(b) O(log n)  
(c) O(n)  
(d) O(n log n)

**If you got to pick a data structure on which you would be performing merge sort, which one would you choose?**(Hint: Think about space complexity)

(a) Array

(b) Linked Structure

(c) Stack

(d) Queue

**Priority Queues:**

**Consider this array representation of a heap, what is the right child of node 75? Assume the zeroth index is null.**

**A = {\_, 100, 75, 50, 51, 40, 30, 3, 25, 10}**

(a) 30

(b) 51

(c) 40

(d) 100

**As mentioned in class, a max heap would be a great data structure when we need to extract the maximum element.**

**Let us say that you are given a few processes that require CPU time which are stored in a heap structure. The process with minimum priority needs to be executed first. Example, a process with priority 0 needs to be picked up before a process with priority 1. We need to select the task with minimum priority.**

**What would be the complexity to simply read the minimum element in a min heap?**

(a) O(1)

(b) O(logn)

(c) O(nlogn)

(d) O(n)