1. Build maximum heap with required methods. It should support the behaviors like adding element, getting maximum element, extracting maximum element, count number of elements and to check the method to test the heap order property.
2. Modify Q1 to sort the input in ascending order.
3. Build minimum heap with required methods. It should support the behaviors like adding element, getting minimum element, extracting minimum element, count number of elements and to check the method to test the heap order property.
4. Modify Q3 to sort the input in descending order.
5. Build priority queue to handle real time tasks. It is assumed that all tasks arrive at same time. The attributes of tasks are task-id, priority and execution time. Compute waiting time, turnaround time for each job. It is treated that 10 is maximum priority and 1 is least priority.
6. Modify Q5 to include deadline for each job.
7. Modify Q6 to include the 'Start Time'. Start Time is the time of creating heap object. User can specify the 'End Time' which indicates time after which tasks are not scheduled. Provide the task schedule which includes the tasks that can be completed without missing deadline within End Time. Scheduling is based on priority.
8. Modify Q5 which gives time range and prepare the schedule of tasks which can be completed within given time range. Tasks which arrives before and after time range are not considered for scheduling.
9. An airport is developing a computer simulation of air-traffic control that handles events such as landings and takeoffs. Each event has a time stamp that denotes the time when the event will occur with additional information like, plane number, takeoff or landing. The simulation program needs to efficiently perform the following two fundamental operations: 1. Insert an event with a

given time stamp, aircraft number, takeoff / landing (add a future event). 2. Extract the event with smallest time stamp (that is, determine the next event to process). Design and implement suitable data structures that work efficiently in terms of time.