1. (20) Given a target value and a sorted array, assuming no duplicates in that array, if the target is found in the array, return its index. If not, return where it should be inserted. **Give a O(log n) time algorithm.** Example: \{2,5,8,10\}, target=5, returns 1. \{2,5,8,10\}, target=6, returns 2.

2. (80) Implement Priority Queue (pq) using max heap, all the elements are stored in the Integer array A, n indicates the size of pq. Let us assume the capacity of the pq is 10, you do not need to use dynamic array to enlarge its size. When n = 10, you can not insert anymore elements. In previous homework we use 0 to represent empty element, this time we use null. You should implement the following functions:
   (a) buildheap(): build a heap from array A
   (b) heapify(i): apply heapify on element A[i]
   (c) insert(int x): insert x into pq
   (d) int maximum(): returns the element of pq with the largest value
   (e) int extract-max(): removes the element of pq with the largest value
   (f) increase-value(i, val): increases the value of A[i] to the new value val
   (g) decrease-value(i, val): decreases the value of A[i] to the new value val

3. (20) Insert 11 to the red-black tree, show all the intermediate steps.