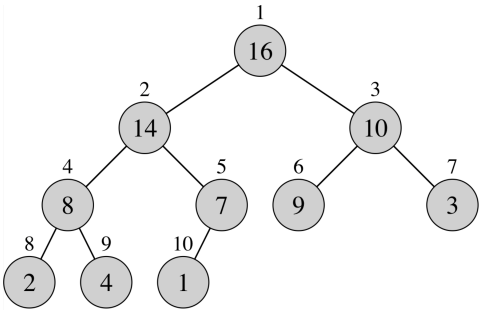
CSCI 200

HW #3: Heaps, heapsort, mergesort, quicksort

Directions: *Complete your work on a separate sheet of paper. Submit the physical copy of your work at the beginning of class on the specified due date. Show your work. You may work in groups of up to 3 students provided that all students participate in each question. Provide a short preliminary explanation of how an algorithm works before running an algorithm or presenting a formal algorithm description, and use examples or diagrams if they are needed to make your presentation clear.*

1. Let T be a (max) heap storing n keys. Give the **pseudocode** for an efficient algorithm for printing all the keys in T that are greater than or equal to a given query key x (which is not necessarily in T). You can assume the existence of a $O(1)$-time *print(key)* function. For example, given the heap below and query key $x = 6$, the algorithm should report $16, 14, 10, 8, 7, 9$. Note that the keys do not need to be reported in sorted order. **Analyze the run time of your algorithm**. It should run in $O(k)$ time, where k is the number of keys reported.



2. Consider the following arrays. Indicate whether they are or are not a max-heap.
   (a) $[23, 17, 14, 6, 13, 10, 1, 5, 7, 12]$
   (b) $[50, 25, 28, 8, 3, 17, 14, 6, 5, 1, 2]$
   (c) $[4, 8, 10, 9, 12, 16, 18, 22, 40]$
   (d) $[15, 14, 13, 10, 12, 9, 8, 5, 7, 3, 2]$
   (e) $[12, 5, 13, 4, 2, 6, 10, 1, 3]$

3. Illustrate the execution of the Build-Max-Heap algorithm on the following array: $[1, 5, 8, 10, 12, 3, 6, 13, 4, 2, 7, 9, 15, 10, 24, 28, 11]$. Show the resulting heap (as a tree) as each level of the tree is processed.

4. Illustrate (using a tree) the execution of heapsort on the same array as above. Use your max-heap data structure constructed above as your starting point. Use Figure 6.4 as a model, showing separately the resulting max-heap after each individual item is removed.

5. Illustrate the execution tree of mergesort on the following array: $[1, 5, 8, 10, 12, 3, 6, 13, 4, 2, 7, 9, 15, 10, 24, 28, 11]$.

6. Using the last element as pivot, illustrate the execution tree of quicksort on the following array: $[5, 8, 2, 4, 3, 7, 6, 9, 10]$

7. Suppose we are given a sequence S of n elements, each of which is colored red or blue. Assuming S is represented by an array, give a linear-time **in-place** algorithm for ordering S so that all the blue elements are listed before all the red elements. What is the running time of your method?

8. Let A and B be two sequences of n integers each. Give an integer m, describe an $O(n \log n)$ time algorithm for determining if there is an integer a in A and an integer b in B such that $m = a + b$.

9. Given an array A of n distinct integers, explain how to find the number of pairs of elements with the **minimum absolute difference** of any two elements. For example, provided the array $A = [4, 2, 6, 1, 3, 10]$, the algorithm would output 3, as there are three pairs with minimum absolute difference: [1,2],[2,3],[3,4].

10. Let $A[1..n]$ be an array of n distinct numbers. If $i < j$ and $A[i] > A[j]$, then the pair (i, j) is called an **inversion** of A.

    (a) List the five inversions of the array $[2, 3, 8, 6, 1]$.

    (b) Describe an algorithm that determines the number of inversions in any permutation of n elements in $O(n \log n)$ worst-case time. *Hint: a divide-and-conquer approach will help to obtain an efficient run-time.*