CSL 252- Design and Analysis of Algorithms
Indian Institute of Technology Bhilai
Tutorial Sheet 3

1. What are the minimum and maximum numbers of elements in a heap of height h?
2. Show that an n-element heap has height $\lfloor \log n \rfloor$.
3. Show that in any subtree of a max-heap, the root of the subtree contains the largest value occurring anywhere in that subtree.
4. Where in a max-heap might the smallest element reside, assuming that all elements are distinct?
5. Is an array that is in sorted order a min-heap?
6. Is the array with values $\{23, 17, 14, 6, 13, 10, 1, 5, 7, 12\}$ a max-heap?
7. Show that, with the array representation for storing an n-element heap, the leaves are the nodes indexed by $\lfloor \frac{n}{2} \rfloor$, $\lfloor \frac{n}{2} \rfloor + 1$, $\lfloor \frac{n}{2} \rfloor + 2$, $\cdots$, n.
8. Starting with the procedure MAX-HEAPIFY, write pseudocode for the procedure MIN-HEAPIFY(A, i), which performs the corresponding manipulation on a min-heap. How does the running time of MIN-HEAPIFY compare to that of MAX-HEAPIFY?
9. What is the effect of calling MAX-HEAPIFY$(A, i)$when the element $A(i)$ is larger than its children?
10. Show that the worst-case running time of MAX-HEAPIFY on a heap of size n is $\Omega(\log n)$. (Hint: For a heap with n nodes, give node values that cause MAX-HEAPIFY to be called recursively at every node on a simple path from the root down to a leaf.)
11. Show that there are at most $\lceil \frac{n}{2^{h+1}} \rceil$ nodes of height h in any n-element heap.
12. Let A and B be two arrays of integers of size n and $2n$, respectively. Design an $O(n \log n)$ algorithm that returns "TRUE" if $A \subseteq B$, otherwise, returns "FALSE".
13. Describe an $O(n \log n)$ time algorithm that, given an array A of n integers and another integer x, determines whether or not there exist two integers in A whose sum is exactly x.
14. Why do we want the loop index i in line 2 of BUILD-MAX-HEAP to decrease from $\lfloor \frac{A.length}{2} \rfloor$ to 1 rather than increase from 1 to $\lfloor \frac{A.length}{2} \rfloor$?
15. Illustrate the operation of PARTITION on the array $\{13, 19, 9, 5, 12, 8, 7, 4, 21, 2, 6, 11\}$.
16. What value of q does PARTITION return when all elements in the array $A[p \cdots r]$ have the same value?
17. How would you modify QUICKSORT to sort into non-increasing order?
18. What is the running time of QUICKSORT when all elements of array A have the same value? Show that the running time of QUICKSORT is $O(n^2)$ when the array A contains distinct elements and is sorted in decreasing order.

19. Suppose that the splits at every level of quicksort are in the proportion $1 - \alpha$ to α, where $0 < \alpha < \frac{1}{2}$ is a constant. Show that the minimum depth of a leaf in the recursion tree is approximately $-\frac{\log n}{\log \alpha}$ and the maximum depth is approximately $-\frac{\log n}{\log(1-\alpha)}$ (Dont worry about integer round-off.)

20. Give an $O(n \log k)$-time algorithm to merge k sorted lists into one sorted list, where n is the total number of elements in all the input lists.

21. Let A be an array of n integers. Describe an algorithm that finds the maximum and minimum of the array A with at most $3\lceil \frac{n}{2} \rceil$ comparisons.

22. A sorting algorithm is called stable, if for any input array A, the relative position of two equal keys will not change before and after the sorting. For example, if $a[2]$ and $a[5]$ have the same value 9 in the input array, then the key 9 of $a[2]$ will appear before the key 9 of $a[5]$ in the sorted array. Example of some stable sorting algorithms are Quick sort, Merge Sort.
    (a) Is Heap sort stable? Explain your answer.
    (b) Show that any sorting algorithm can be made stable without increasing its asymptotic time complexity.