1. (5) What is the smallest possible depth of a leaf in a decision tree for a comparison sort?

\[ n - 1 \]


(a) A[1] has d children that are kept at 2 \ldots d + 1 and their children are kept at d + 2, \ldots d^2 + d + 1.
(b) D-ary parent A[(i - 2)/d + 1]
(c) D-ary child (i,j) A[d(i - 1) + j + 1]
(d) The height of d-ary heap is \( \Theta(\log_d n) \).
(e) EXTRACT-MAX \( \Theta(d \log_d n) \), since one has to examine d children during heapify.
(f) HEAP-INSERT running time is \( \Theta(\log_d n) \).
(g) HEAP-INCREASE KEY similar to heap insert \( \Theta(\log_d n) \) - proportional to the height of the heap.

3. (10) page 173, problem 8.3-4. (9-3.4 old book) sorting in linear time. Use radix sort on two digits and counting sort for sorting within each digit. In order for the counting sort to run in linear time the digit size should be \( \log n \). The total running time will be \( O(2n) = O(n) \).

4. (20) page 194, problem 9-1 (10-1 old book) largest i numbers in a sorted order.

(a) Sort the numbers using heapsort or merge sort in \( \Theta(n \log n) \) time. Putting the first i elements in the output array will take \( \Theta(i) \) time. Total time is \( \Theta(n \log n + i) \).
(b) Implement priority queue as a heap. BUILD-HEAP takes \( \Theta(n) \) time. Then calling HEAP-EXTRACT-MAX i-times will take \( \Theta(i \log n) \). The worst-case extraction is \( \Theta(i \log n) \) because i extractions from the heap with n elements takes \( O(i \log n) \) and half of these extractions will be from the heap with at least \( n/2 \) elements, so they will take \( (i/2)\Omega(i \log(\frac{n}{2})) = \Omega(i \log n) \). Total time is \( \Theta(i \log n + n) \).
(c) Use the select algorithm to find the ith largest number in \( \Theta(n) \) time. Partition around that number in \( \Theta(n) \) time. Sort the i largest numbers in \( \Theta(i \log i) \) time. The total running time is \( \Theta(n_i \log i) \). The third and second method are asymptotically at least as good as first and third method is as least as good second. So third is at least as good at second and first methods.