CSci 231 Homework 2

1 Practice problems

These are intended to help you study. You do not need to turn them in.

1. What are the minimum and maximum number of elements in a heap of height h? Note: the height of a heap is the number of edges on the longest root-to-leaf path.
2. Where in a min-heap might the largest element reside, assuming that all elements are distinct?
3. Is an array that is in sorted order a min-heap?
4. What is the effect of calling MIN-HEAPIFY(A, i) for $i > size[A]/2$?
5. What is the running time of QUICKSORT when all elements of array A have the same value?
6. Briefly sketch why the runnning time of QUICKSORT is $\Theta(n^2)$ when the arrary A contains distinct elements and is sorted in decreasing order.
7. Argue that for any constant $0 < \alpha \leq 1/2$, the probability is approximately $1 - 2\alpha$ that on a random input array, PARTITION produces a split more balanced that $(1-\alpha)$-to-α.
8. Professors Dewey, Cheatham, and Howe have proposed the following "elegant" sorting algorithm:

   STOOGE-SORT(A, i, j)
   if $A[i] > A[j]$
       then exchange $A[i] \leftrightarrow A[j]$
   if $i + 1 \geq j$
       then return
   $k \leftarrow \lfloor (j - i + 1)/3 \rfloor$
   STOOGE-SORT$(A, i, j - k)$
   STOOGE-SORT$(A, i + k, j)$
   STOOGE-SORT$(A, i, j - k)$

   a. Argue that STOOGE-SORT$(A, 1, length[A])$ correctly sorts the input array $A[1..n]$, where $n = length[A]$.

   b. Give a recurrence for the worst-case running time of STOOGE-SORT and a tight asymptotic (Θ-notation) bound on the worst-case running time.

   c. Compare the worst-case running time of STOOGE-SORT with that of insertion sort, merge sort, heapsort, sock sort, and quicksort. Do the professors deserve praise?

9. Which of the following sorting algorithms are stable: insertion sort, merge sort, quicksort? Give a simple scheme that makes any sorting algorithm stable. How much additional time and space does your scheme entail?

10. Suppose that you have a "black-box" worst-case linear-time median subroutine. Give a simple, linear-time algorithm that solves the selection problem SELECT(i) for an arbitrary i.

11. Illustrate the operation of BUCKET-SORT on the array

$$A = [.79, .13, .16, .64, .39, .20, .89, .53, .71, .42]$$

2 Homework

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

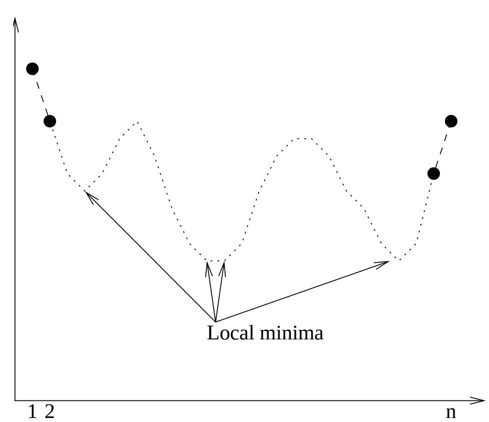
2. Given a set of n numbers, we wish to find the i largest in sorted order using a comparison-based algorithm. Find the algorithm that implements each of the following methods with the best asymptotic worst-case running time, and analyze the running times of the algorithms on terms of n and i.

   (a) Sort the numbers, and list the i largest.

   (b) Build a max-priority queue from the numbers, and call EXTRACT-MAX i times.

   (c) Use a SELECT algorithm to find the ith largest number, partition around that number, and sort the i largest numbers.

3. Consider an array A of length n for which we know that $A[1] \geq A[2]$ and $A[n-1] \leq A[n]$. We say that $A[x]$ is a *local minimum* if $A[x-1] \geq A[x]$ and $A[x] \leq A[x+1]$. Note that A must have at least one local minimum.



   We can obviously find a local minimum in $O(n)$ time by scanning through A. Describe an $O(\log n)$ algorithm for finding a local minimum.

4. Describe an $O(n)$ algorithm that, given a set S of n distinct numbers and a positive integer $k \leq n$, determines the k numbers in S that are closest (in value) to the median of S.

5. Let A be a list of n (not necessarily distinct) integers. Describe an $O(n)$-algorithm to test whether any item occurs more than $\lceil n/2 \rceil$ times in A. Your algorithm should use $O(1)$ additional space.

6. Give an $O(n \lg k)$ algorithm to find the $k-1$ elements in a set that partition the set into (approx.) k equal-sized sets $A_1, A_2, \ldots A_k$ such that all elements in A_i are smaller than all elements in A_{i+1}. Assume k is a power of 2.

7. Show how to sort n integers in the range 1 to n^2 in $O(n)$ time.