Homework 3

Released on October 5, 2016

---

1. Consider the following algorithm:

```
PERMUTE(A)
    n = A.length;
    for i = 1 to n do
        swap A[i] with A[RANDOM(1, n)];
```

Does this code produce a uniform random permutation? Why or why not?

2. Prove that in the array P in procedure PERMUTE-BY-SORING, the probability that all elements are unique is at least $1 - 1/n$.

3. Write a non-recursive code for MAX-HEAPIFY(A, i).

4. Argue the correctness of HEAPSORT using the following loop invariant: At the start of each iteration of the **for** loop of lines 2-5, the subarrary $A[1..i]$ is a max-heap containing the i smallest elements of $A[1..n]$, and the subarrary $A[i+1..n]$ contains the $n-i$ largest elements of $A[1..n]$, sorted.

5. Give an $O(n \log k)$ algorithm (pseudocode) to merge k sorted lists into one sorted list, where n is the total number of elements in all the input lists. (Explain the basic idea before writing your code.)
Hint: You may use min-heap and you do not need to write the methods for min-heap.

6. In running time analysis of QUICKSORT using randomized partition, we assumed that all elements have distinct values. What's the problem without this assumption?

7. We modify quicksort as follows: Upon calling quicksort on a subarrray with fewer than k elements, let it simply return without sorting the subarrary. After the top-level call to quicksort returns, run insertion sort (*cf.* Section 2.1) on the entire array to finish the sorting process. Assume that all elements have distinct values.

(1) Show that this sorting algorithm runs in $O(nk + n\log(n/k))$ expected time.
Hint: $H_n = \sum_{i=1}^{n} \frac{1}{i} = \ln n + O(1)$.

(2) How should we pick k, both in theory and in practice?