Problem Set 1

Due date: Oct 12, 2021, end of the day

For calculation questions, you should give the steps you take to arrive at your solution. Your final solution does not need to be in closed form, but simplify it as much as you can.

You may get full marks without doing bonus questions.

1. (7 points) A priority queue is a data structure that, among other operations, has a pointer to the maximum element currently stored in the data. Suppose we have n distinct integers, arriving in an ordering that is uniformly at random, to be stored in a priority queue. Each time an integer larger than all previous ones arrives, the pointer is updated to point to the new element. Compute the expected number of times the pointer is updated throughout the procedure.

2. (7 points) Given an undirected graph $G = (V, E)$, let n be $|V|$, the number of vertices, and m be $|E|$, the number of edges. For $S \subseteq V$, the subgraph *induced* on S is the graph with vertex set S and all the edges in E whose two endpoints are both in S. Give a randomized algorithm to pick $S \subseteq V$ with $|S| = k$, such that the expected number of edges in the subgraph induced on S is at least $\frac{mk(k-1)}{n(n-1)}$. Your algorithm should run in polynomial time. Show that your algorithm is correct.

3. (7 points) We count 50 noodles and cook them in a pot. The pot is not large enough, so the 100 ends of the noodles stick out. As the noodles soften, we pick two ends uniformly at random and tie them together. Then we pick another two ends uniformly at random, tie them, and repeat, until there are no loose ends. How many loops are formed in expectation?

4. A binary tree is said to be *height balanced* if for each node x, the heights of the left and right subtrees of x differ by at most 1. An *AVL tree* is a binary search tree that is height balanced. To maintain an AVL tree, we maintain an extra field in each field: $h[x]$ is the height of node x. Let $root[T]$ point to the root node of a tree T.

   (a) (4 points) Prove that an AVL tree with n nodes has height $O(\log n)$. (Hint: Prove that in an AVL tree of height h with $h \geq 1$, there are at least F_h nodes, where F_h is the h-th Fibonacci number: $F_0 = 1, F_1 = 1, F_2 = 2, F_3 = 3, F_4 = 5, \cdots, F_n = F_{n-1} + F_{n-2}$. You may use the fact $F_n \geq (3/2)^n$.)

   (b) (6 points) To insert into an AVL tree, a node is first placed in the appropriate place in binary search tree order. After this insertion, the tree may no longer be height balanced. Specifically, the height of the left and right children of some node may differ by 2. Describe a procedure BALANCE(x), which takes a subtree rooted at x whose left and right children are each height balanced and have heights that differ by at most 2, i.e., $|h[right[x]| - h[left[x]| \leq 2$, and alters the subtree rooted at x to be height balanced. (Hint: Use tree rotations.)

(c) (5 points) Using part (4b), describe a procedure AVL-INSERT(x, z), which takes a node x within an AVL tree and a newly created node z (whose key has already been filled in), and adds z to the subtree rooted at x. Thus, to insert the node z into the AVL tree T, we call AVL-INSERT$(root[T], z)$. Show that AVL-INSERT, run on an n-node AVL tree, takes $O(\log n)$ time.

(d) (bonus question, 3 points) Show that AVL-INSERT performs only $O(1)$ rotations for each insertion.