(1) **Sorting in Linear Time (40 points)**

In lecture, we discussed how we can sort an array of n integers in the range \([0, k]\) in \(O(n + k)\) time. In this question, we’ll show that we can actually expand the range of our values without incurring too much cost.

(a) (30 points) Given an array \(A\) of \(n\) pairs of integers \((a_1, b_1), (a_2, b_2), \ldots, (a_n, b_n)\), where \(a_i, b_i \in [0, n]\) for each \(i\).

We say two pairs \((a_i, b_i), (a_j, b_j)\) are in lexicographic order if and only if one of the following holds:

- \(a_i < a_j\)
- \(a_i = a_j\) and \(b_i \leq b_j\)

(e.g., the ordered pairs \((1, 2), (1, 2)\), \((1, 2), (1, 3)\), and \((1, 2), (2, 1)\) are in lexicographic order, while the ordered pairs \((1, 2), (1, 1)\) and \((2, 1), (1, 2)\) are not.) Give an algorithm that sorts \(A\) according to lexicographic order in \(O(n)\) time. **Formally prove** the correctness and runtime of your algorithm.

(b) (10 points) **Informally explain** how part (a) can be used to sort \(n\) integers in the range \([0, n^2]\) in \(O(n)\) time.

(2) **Repetition Detection (50 points)**

Given a sorted integer array \(A\) of size \(n\), where \(n\) is a multiple of 4. Propose an algorithm that decides whether not there exists an integer that repeats at least \(n/4\) times in the array:

(a) (20 points) In \(O(n)\) time.

(b) (30 points) In \(O(\log n)\) time.

**Formally prove** the correctness and runtime of both algorithms.

(3) **Binary Search Trees (30 points)** Problem 3 will require you to draw trees. To cut down on the amount of drawing you have to do, you will only be required to show the final tree for full credit. This means you will only have to draw (at most) five trees total: one for each part of 3a that you answer yes to, and one for each part of 3b. If you want you may draw your trees by hand and scan them into Gradescope.

(a) (15 points) For each of the following unlabeled binary trees, state whether or not it can be the structure of a red-black tree. If yes, label the nodes red or black. If no, state which of the red-black tree invariants cannot be satisfied and why.

(b) (15 points) Perform the following operations on this binary search tree, and draw the final result for each part. Each part of this problem is independent of each other – the operations in part 2 must be performed on the original tree, and not on the output of part 1.

(i) (5 points) Use the Insert procedure in the textbook/lecture notes to insert 14 into this tree.

![Figure 1. Trees for part 3a(i) (7 points) and (ii) (8 points)](image-url)
(ii) (5 points) Use the Delete procedure in the textbook/lecture notes to delete 6 from this tree.

(iii) (5 points) Use the Delete procedure in the textbook/lecture notes to delete 15 from the tree.

Figure 2. Binary tree for part 3b

(4) (2 points) Don’t forget to fill out the feedback form, located here: http://goo.gl/forms/GxcPClhWm0x1R7kR2

Note that the feedback form extra credit will be given at the end of the class, after grade cutoffs have already been determined, so that the extra credit is actually extra.