Due on January 29, 2018
30 points total

General Directions: If you are asked to provide an algorithm, you should clearly define each step of the procedure, establish its correctness, and then analyze its overall running time (for this assignment, this means arguing why your algorithm achieves the target running time specified by the question). There is no need to write pseudo-code; an unambiguous description of your algorithm in plain text will suffice.

Please make sure that you **START EARLY**. Some of these questions might take you several hours to solve. Since this is a mathematical/theoretical class, the ratio between how much time you spend thinking and how long it takes you to write your solution will be high (at least higher than what you might be used to in coding assignments).

All the answers must be typed, preferably using LaTeX. If you are unfamiliar with LaTeX, you are strongly encouraged to learn it. However, answers typed in other text processing software and properly converted to a pdf file will also be accepted. Before submitting the pdf file, please make sure that it can be opened using any standard pdf reader (such as Acrobat Reader) and your entire answer is readable. **Handwritten answers or pdf files that cannot be opened will not be graded and will not receive any credit.**

Finally, please read the detailed collaboration policy given on the course website. You are **not** allowed to discuss homework problems in groups of more than 3 students. **Failure to adhere to these guidelines will be promptly reported to the relevant authority without exception.**

Problem 1 (20 points)
Professor Dumbledore (a long lost brother of the beloved headmaster) teaches algorithms in Magicland. After the first midterm exam, he wants to carefully scrutinize the students' performance, but unfortunately, the TAs handed him an unsorted array of scores $L = x_1, \ldots, x_n$ and left on vacation in Spring Break. Having come to know of your prowess in algorithms after taking COMPSCI 330, he recruits you to come up with an algorithm to sort these scores. If there were only k distinct scores among all n scores, can you sort L in $O(n \log k)$ time? Note that this does not necessarily imply that the set of possible scores is $\{1, \ldots, k\}$. Also note that k is much smaller than n; therefore, an $O(n \log n)$ time algorithm will not suffice. (An $O(n \log k + k^2)$ solution will get partial credit.)

Problem 2 (10 points)
Alarmed at the performance of students in the first midterm, Professor Dumbledore decided to allow limited consultation in the second midterm. He first arranged students in an arbitrary order and then stipulated that every student is only allowed to discuss answers with her two neighbors (one neighbor for the students at the two ends ... tough luck!). Not surprisingly, students who discussed among themselves ended up obtaining very similar scores: in fact, the scores of any pair of adjacent students turned out to differ by at most 1. Formally, if $X = x_1, x_2, \ldots, x_n$ is an array of students' scores in the order in which they were arranged, then $|x_i - x_{i+1}| \leq 1$ for $1 \leq i \leq n - 1$. Also, suppose $x_1 \leq x_n$. Given a particular number $x_1 \leq b \leq x_n$, can you help the professor find at least one student who got a score of b in $O(\log n)$ time? Note that there may be multiple students with a score of b but you only need to find one. You can assume that all the scores are integers and b is an integer as well.