Homework 2

Due: September 27, 2022 - 14:30 ET

This homework must be typed in LaTeX and handed in via Gradescope.

Please ensure that your solutions are complete, concise, and communicated clearly. Use full sentences and plan your presentation before you write. Except in the rare cases where it is indicated otherwise, consider every problem as asking you to prove your result.

Problem 1

1. Assume that we want to sort sequences of numbers with the following property:
   P: "Each element in the input sequence is placed in the input at most k locations away from its placement in the output ordered sequence".
   Design a modification of Selection sort which runs in time $O(nk)$.

2. Analyze the worst-case running time of Insertion Sort for strings with the **P** property.

3. For input sequences exhibiting property **P**, which of Mergesort, Insertion Sort, or Standard Selection sort would you use for better performance? How does your choice change for different values of k? Motivate your answer.

4. Assume now that we are looking to sort input sequences in which there is a subsequence of size k which is already sorted. Show that using one among Selection, Insertion, or Bubble sort it is possible to sort such inputs in $O(n(n-k))$ steps.

Problem 2

Joanna is in charge of a law firm working on an important case, where she needs to find a specific folder which is stored in one of the company filing cabinets. She assigns this task to Rohit the Paralegal.

Rohit has at most N workers at his disposal to complete this task. The filing cabinet has M shelves, and each shelf has F_i folders. Assume that there are always more shelves than Rohit has workers. Joanna wants to minimize chaos, so she asks Rohit to assign workers to a contiguous sub-array of shelves.

Rohit wants to divide the work as evenly as possible among the workers. Help Rohit come up with an algorithm to find the **minimum number of folders** that each of his workers has to look at in order to find the specific folder.

Since Joanna is a strict boss, please give an algorithm that runs in $O(M \log(S))$, where S is the total number of folders in the entire filing cabinet.

Problem 3

Let S be an array of n unique integers. An inversion in S is a pair of indices i and j such that $i < j$ but $S[i] > S[j]$. Describe an algorithm running in $O(n \log n)$ time for determining the number of inversions in S.