CS 4349.501 Homework 3

Due Wednesday September 20th, in class

September 13, 2017

Please answer each of the following questions. Each student must write their solutions in their own words and submit their solutions on paper at the beginning of class. ***Include your name and/or Net ID at the top of each page.***

1. An *inversion* in an array $A[1 .. n]$ is a pair of indices i, j such that $i < j$ and $A[i] > A[j]$. The number of inversions in an n-element array is between 0 (if the array is sorted) and $\binom{n}{2}$ (if the array is sorted backward). Describe and analyze an algorithm to count the number of inversions in an n-element array in $O(n \log n)$ time. ***Hint***: Modify mergesort.

2. You are spending your summer working at wonderful Funtime Resorts and Money Sink, a magical place that provides your guests with hours of entertaining rides, games, and excuses to spend their hard earned cash. Unfortunately, the park rules require all guests to exchange their normal currency for Funtime FunDollars before making any purchases. Funtime FunDollars come in only three denominations: 1FD bills, 4FD bills, and 6FD bills. As an employee of Funtime, you will need to make change (in FunDollars) for guests.

   (a) Consider the following *greedy* algorithm for making change for k FunDollars. Hand over the largest bill less than or equal to k, and then recursively make change for the amount that remains. Give an example where this strategy forces you to hand over more bills than the minimum possible.

   (b) Design a recursive algorithm that computes, given a value k, the minimum number of bills needed to make change for k FunDollars. You do not need to worry about making your algorithm efficient, but it should be correct. You should express the running time of your algorithm as a recurrence in k, but you do not have to solve the recurrence.

   (c) Design and analyze an efficient algorithm that computes, given a value k, the minimum number of bills needed to make change for k FunDollars. Your analysis should give the asymptotic running time of your algorithm in terms of k. This algorithm should be fast.