# CS515: Algorithms and Data Structures
# Midterm Exam

**The exam is TWO pages.**

**Problem 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 in increasing order) and $\binom{n}{2}$ (if the array is sorted in decreasing order).

Describe and analyze an algorithm to compute the number of inversions of an $n$-element array in $O(n \log n)$ time. Suppose, to simplify the explanation, that the elements of the array are distinct.

**Problem 2.** Suppose you are given a tree $T$ with root $r$ and positive weights on its edges.

(a) Describe and analyze an algorithm to find a simple path in $T$ with maximum total weight that starts at $r$.

(b) Describe and analyze an algorithm to find a simple path in $T$ with maximum total weight.

For both parts, $O(n)$ time algorithms receive full credit.

**Problem 3.** Recall the job scheduling problem. The input is composed of the starting and finishing times of $n$ jobs. We would like to find the maximum set of pairwise disjoint jobs.

Consider the following alternative greedy algorithms for the job scheduling problem. For each algorithm, either prove or disprove (by presenting a counter example) that it always constructs an optimal schedule.

(a) Choose the job that ends last, discard all conflicting jobs, and recurse.

(b) Choose the job that starts first, discard all conflicting jobs, and recurse.

(c) Choose the job that starts last, discard all conflicting jobs, and recurse.

(d) Choose the job with shortest duration, discard all conflicting jobs, and recurse.

**Problem 4.** Suppose you are given $n$ bolts and $n$ nuts. The bolts have distinct sizes. Also, the nuts have distinct sizes. Each bolt match exactly one nut.

Consider the following randomized algorithm for choosing the **largest** bolt. Draw a bolt uniformly at random from the set of $n$ bolts, and draw a nut uniformly at random from the set of $n$ nuts. If the bolt is smaller than the nut, discard the bolt, draw a new bolt uniformly at random from the unchosen bolts, and repeat. Otherwise, discard the nut, draw a new nut uniformly at random from the unchosen nuts, and repeat. Stop either when every nut has been discarded, or every bolt except the one in your hand has been discarded.

(a) What is the probability that the algorithm discards NO bolts?

(b) What is the exact expected number of discarded bolts when the algorithm terminates?

(c) What is the probability that the algorithm discards exactly one nut?

(d) What is the exact expected number of discarded nuts when the algorithm terminates?

(e) What is the exact expected number of nut-bolt tests performed by this algorithm?

**Problem 5. [Extra Credit]** Suppose you are given an $n \times n$ bitmap, represented by an array $M[1 \ldots n, 1 \ldots n]$ of 0s and 1s. A solid block in $M$ is a subarray of the form $M[i \ldots i', j \ldots j']$ containing only 1-bits. Describe and analyze an algorithm to find the maximum area of a solid block.

The amount of extra credit you earn highly depends on the running time of your algorithm. To earn any extra credit the running time of your algorithm must be $O(n^4)$.