CpSc 8400: Design and Analysis of Algorithms

Instructor: Dr. Brian Dean
Webpage: `http://www.cs.clemson.edu/~bcdean/`
Handout 15: Quiz #2.

Spring 2016
TTh 12:30-1:45
McAdams 119

Some of the following questions ask you to design an algorithm. Please do so *clearly* and also *concisely*, in English (pseudocode is generally not necessary). For each algorithm, please also briefly justify its correctness and running time. Standard results from class or the textbook can be used as "black boxes" without extra elaboration. Partial credit will be awarded for valuable insight or slightly slow but correct solutions, as long as they are not overly complicated. Use the RAM model of computation unless otherwise stated.

This is a take home quiz. By the start of class on Thursday, April 7, you must email electronic solutions to all problems to the instructor (just like with the homework assignments). **Until the beginning of class on Thursday, you are not permitted to collaborate with any other human being about the quiz problems. Furthermore, you may not consult any algorithmic reference material except for materials handed out in this course (e.g., the textbook, homework solutions, and lecture slides).**

1. Number Guessing. Prof. Dean has a secret number in the range $1 \ldots n$, which you would like to determine. You are allowed to make several guesses. For each guess that is too low, your final grade in CpSc 8400 drops by x points. For each guess that is too high, your grade drops by y points. You may assume x and y are integers in the range $1 \ldots n$. Please describe an efficient algorithm that computes the smallest possible number z such that you can *guarantee* finding Prof. Dean's secret number with a loss of no more than z points.

2. Embedding. You are given an array $A[1 \ldots n]$ containing integers as input. Recall that a *subsequence* of A is a subset of the elements of A appearing in the same order as in A. For example, [4 1 7] is a subsequence of the larger array [1 8 4 2 1 7 9]. Please describe an algorithm that determines the minimum possible length of an array B that is *not* a subsequence of A. You may only use an integer as an element in B if it appears in A.

3. Random Stuff. Please answer the following:

(a) A *local max* in a 2D array is an element that is larger than all of its 4 neighbors (or fewer neighbors, if it is on the boundary of the array). If an $n \times n$ array is filled with a random permutation of the integers $1 \ldots n^2$, what is the expected number of locally maximal elements?

(b) In an infinite-length random binary string (each character equally likely to be 0 or 1), what is the expected length of a palindrome centered on some particular character? Recall that a palindrome is a string that reads the same forward or backward. In this case, we are only interested in an odd-length palindrome, having a distinct central character.

(c) Suppose you generate n intervals $[a_1, b_1], \ldots, [a_n, b_n]$ by choosing the a_i's to be a random permutation of $1, 2, 3 \ldots n$, and the b_i's to be a random permutation of $n+1, n+2, \ldots, 2n$. Please argue that with high probability, the maximum *nesting depth* of over all intervals is $O(\log n)$. The nesting depth of an interval I is either zero if no other interval contains I, or otherwise one plus the maximum nesting depth over all intervals containing I.