6331 - Algorithms, Spring 2014, CSE, OSU
Midterm
Instructor: Anastasios Sidiropoulos

Problem 1. Sorting:

(a) What is the running time of Heap-Sort, when the input is an array of size n with all entries equal? For example, $A = [1, 1, \ldots, 1]$. Justify your answer.

(b) What is the running time of Heap-Sort, when the input is an array of n distinct elements sorted in decreasing order? For example, $A = [n, n-1, \ldots, 1]$. Justify your answer.

(c) Suppose that you modify the Heap-Sort algorithm so that instead of a Binary Max-Heap, you use a Fibonacci Max-Heap. What is the worst case running time of the resulting algorithm? Justify your answer.

(d) We can sort a given set of n numbers by first building a Binary Search Tree containing these numbers (using Tree-Insert repeatedly to insert the numbers one by one) and then printing the numbers by an inorder tree walk. What are the worst-case and best-case running times for this sorting algorithm? Justify your answer.

Problem 2. Indiana Jones and the temple of doom: Let $A[1 \ldots n, 1 \ldots n]$ be an integer array with n columns and n rows. The array encodes the map of a room of dimension $n \times n$. We have $A[i, j] = 1$ if there is a gold coin at location (i, j) in the room, and otherwise $A[i, j] = 0$. Indiana Jones is initially located at position $(1, 1)$, which is the north-west corner of the room, and has to reach the exit at location (n, n), which is the south-east corner. Indiana Jones is in a hurry, so he can move only south, or east. That is, at each step, starting at location (i, j) he can either move to location $(i + 1, j)$, or $(i, j + 1)$. Note that he cannot move diagonally.

Design an algorithm that given the array A as input, outputs a sequence of moves for Indiana Jones, that allows him to collect the maximum possible number of gold coins before leaving the room. The running time of your algorithm should be polynomial in n.

Hint: Use dynamic programming.