Analysis of Algorithms: Assignment 4

Due date: September 28 (Tuesday)

Problem 1 (3 points)
Give an efficient implementation of a Heap-Increase-Key(A, i, k) algorithm, which sets $A[i] \leftarrow \max(A[i], k)$ and updates the heap structure appropriately. Determine its time complexity and briefly explain your answer.

Problem 2 (3 points)
Using Figure 8.1 (page 155) in the textbook as a model, illustrate the operation of the Partition algorithm (which is a subroutine of Quick-Sort) on the following array:

4 5 1 4 2 1 8 3

Problem 3 (4 points)
Briefly describe how to adapt (a) Merge-Sort and (b) Quick-Sort to sort elements stored in a linked list, without copying them into an array. Give the time complexity of your algorithms; is it the same as the complexity of sorting an array?

Problem 4 (bonus)
This problem is optional, and it allows you to get 2 bonus points toward your final grade for the course. You cannot submit this bonus problem after the deadline.

A *d-ary heap* is like a binary heap, but instead of 2 children, nodes have d children.

(a) How would you represent a d-ary heap in an array? What are the expressions for determining the parent of a given element, Parent(i), and a j-th child of a given element, Child(i, j), where $1 \leq j \leq d$?

(b) What is the height of a d-ary heap of n elements in terms of n and d? You need to give an *exact* expression for the height, without using the Θ-notation.