CSCI 305, Homework # 5

Caleb Ouellette

Due date: Midnight, May 14

1. Analysis of d-ary heaps (problem 6-2 in the text).

   A **d-ary heap** is like a binary heap, but (with one possible exception) non-leaf nodes have d children instead of 2 children.

   (a) How would you represent a d-ary heap in an array? Use the same strucutre as a binary tree, exect use h instead of 2. So the root is still at one, then its h children. After that is the h children on the left child of the first node and so on.

   (b) What is the height of a d-ary heap of n elements in terms of n and d?

   $$\lfloor log_h n \rfloor$$

   (c) Give an efficient implementation of Extract-Max in a d-ary max-heap. Analyze its running time in terms of d and n.

   ```
   EXTRACT-MAX($A$)
   if $n < 1$
       error "heap underflow"
   $max = A[1]$
   $A[1] = A[n]$
   $n = n - 1$
   MAX-HEAPIFY($A, 1, n$)
   return $max$
   ```

   The Extract-Max function will be dominated by Max-Heapify. Max-Heapify will take $\Theta(log_h n)$ time.

   (d) Give an efficient implementation of Insert in a d-ary max-heap. Analyze its running time in terms of d and n.

   (e) Give an efficient implementation of Increase-Key(A, i, k), which flags an error if $k < A[i]$, but otherwise sets $A[i] = k$ and then updates the d-ary max-heap structure appropriately. Analyze its running time in terms of d and n.