EE602: ALGORITHMS

HOMEWORK #3 Due: 10/12

Hualiang Li

(1) Exercise 6.1-2

The number of internal nodes is a complete tree, which has 2^h-1 nodes, where h is th height of the tree. A heap of height h has at least one additional node (otherwise it would be a heap of length h−1) and at most 2^h additional nodes (otherwise it would be a heap of length h+1). Therefore, $2^h < n < 2^h$-1. So, h = floor(lg n).

(2) Exercises 6.1-3

This follows from the max-heap property. If the ith element is the root of the subtree, then both its children would be less or equal to it. Since this properties holds for their children too and it is transitive, all of the descendents will be less or equal to the root, making it the largest value.

(3) Exercises 6.3-3

Prove by induction:

Base: h=0. The number of leaves is $\lceil n/2 \rceil = \lceil n/2^{0+1} \rceil$

Step: Let's assume it holds for nodes of height h−1. Let's take a tree and remove all it's leaves. We get a new tree with $n-\lceil n/2 \rceil = \lfloor n/2 \rfloor$ elements. Note that the nodes with height h in the old tree have height h−1 in the new one.

We will calculate the number of such nodes in the new tree. By the inductive assumption we have that T, the number of nodes with height h−1 in the new tree, is:

$$T=\lceil \lfloor n/2 \rfloor /2h-1+1 \rceil < \lceil (n/2)/2h \rceil = \lceil n2h+1 \rceil$$

As mentioned, this is also the number of nodes with height h in the old tree.

(4) Exercises 7.2.2 What is the running of Quicksort if all the elements have the same value?

It is $\Theta(n^2)$, since one of the partitions is always empty. In each partition, we only can confirm the position for last element. We have to do n partitions, which mean the running time is n^2. This can be showed in the recursion tree:

Index :	0……r	
Partition 1	0……r-1, r	n
Partition 2	0……r-2, r-1, r	n-1
	.	
	.	
	.	
Partition n	0, 1, 2,…, r-1, r	1

$T = n + n-1 + n-2 + \ldots + 2 + 1 = n(n-1)/2$, which is $\Theta(n^2)$.

(5) Exercises 7.3.1.
The worst-case running time is not triggered by a specific output, but occurs randomly. We're not interested in it, since we cannot reproduce it reliably. Instead, it is factored in the analysis of the expected running time.

(6) Exercises 7.4.2 Show that the best-case running time of Quicksort is THETA(n lg n).
The best case happens when the partition is even, that is:

$T(n)=2T(n/2)+\Theta(n)$

Using the master method, a = 2, b = 2, $f(n) = \Theta(\log_b^a) = \Theta(n)$, apple to master theory 2, we get the solution $\Theta(nlgn)$.