Homework 3

Due Monday, July 23rd at 11:59pm

Name: Zachary McNulty
Student ID: 1636402

Problem 2: Solving Recurrences

(i)
Master Theorem Parameters: $d = 1, a = 7, b = 2, c = 2$
Thus, we can see that $log_b(a) = log_2(7) > 2 = c$. As such, $log_b(a) > c$ and thus:
$T(n) \in \Theta(n^{log_b(a)}) = \Theta(n^{log_2(7)})$

(ii)
Master Theorem Parameters: $d = 1, a = 4, b = 2, c = 2$
Thus, we can see that $log_b(a) = log_2(4) = 2 = c$. As such, $log_b(a) = c$ and thus:
$T(n) \in \Theta(n^c log(n)) = \Theta(n^2 log(n))$

(iii)
Master Theorem Parameters: $d = 1, a = 2, b = 2, c = \frac{1}{2}$
Thus, we can see that $log_b(a) = log_2(2) = 1 > 1/2 = c$. As such, $log_b(a) > c$ and thus:
$T(n) \in \Theta(n^{log_b(a)}) = \Theta(n^{log_2(2)}) = \Theta(n)$

(iv)
Master Theorem Parameters: $d = 1, a = 4, b = 2, c = 3$
Thus, we can see that $log_b(a) = log_2(4) = 2 < 3 = c$ As such, $log_b(a) < c$ and thus:
$T(n) \in \Theta(n^c) = \Theta(n^3)$

(v)
Master Theorem Parameters: $d = 1, a = 3, b = 2, c = 1$
Thus, we can see that $log_b(a) = log_2(3) > 1 = c$. As such, $log_b(a) > c$ and thus:
$T(n) \in \Theta(n^{log_b(a)}) = \Theta(n^{log_2(3)})$

b)

$T(n)$
$= T(n^{0.5}) + T(n^{0.5}) + log(n)$
$= (T((n^{0.5})^{0.5}) + T((n^{0.5})^{0.5}) + log(n^{0.5})) + T((n^{0.5})^{0.5}) + T((n^{0.5})^{0.5}) + log(n^{0.5})) + log(n)$

$i = 0$:	$log(n)$
$i = 1$:	$log(n^{0.5})$ $log(n^{0.5})$
$i = 2$:	$log((n^{0.5})^{0.5}))$ $log((n^{0.5})^{0.5}))$ $log((n^{0.5})^{0.5}))$ $log((n^{0.5})^{0.5}))$

# nodes at level i:	2^i
input size at level i:	$n^{0.5^i}$
work per node at level i:	$log(size) = log(n^{0.5^i}) = 0.5^i * log(n)$
total work at level i:	$work_i * nodecount_i = (2^i)((0.5^i)log(n)) = log(n)$.
level base case:	$n^{0.5^i} = 2 \rightarrow 0.5^i * log_2(n) = log_2(2) = 1$ $\rightarrow log_2(0.5^i * log_2(n)) = log_2(1) = 0$ $\rightarrow i * log_2(0.5) = -log_2(log_2(n))$ $\rightarrow i = log_2(log_2(n))$
number nodes base case:	$2^i = 2^{log_2(log_2(n))} = log_2(n)$
expression for recursive work:	$\sum_{i=0}^{log_2(log_2(n))-1} log(n)$
expression for non-recursive work:	$log_2(n) * 1$
closed form for total work:	$recursive: log(n) + log(n) + ... + log(n)$ $= log(n) * log_2(log_2(n))$ total $= log(n) * log_2(log_2(n)) + log_2(n)$
simpliest big Θ for total work:	$\Theta(log(n) * log(log(n)))$