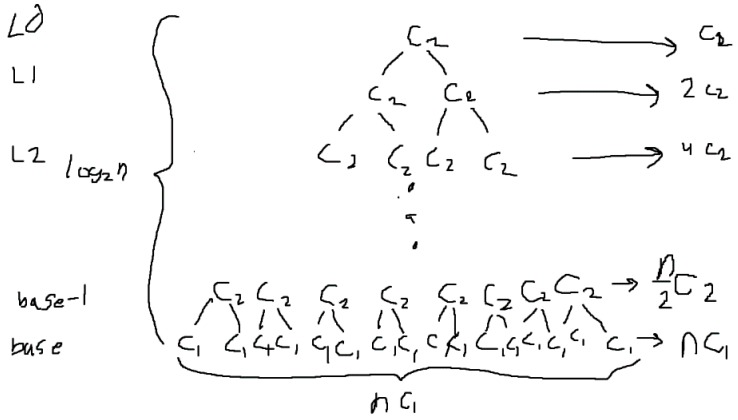
1. a. $f(n) = \theta(g(n))$
   b. $f(n) = \theta(g(n))$
   c. $f(n) = \Omega(g(n))$
   d. $f(n) = \Omega(g(n))$
   e. $f(n) = O(g(n))$ or more accurately $f(n) = \text{Theta}(g(n))$

2. a. Returns the smallest value in the array
   b. $T(1) = \theta(1)$
   $T(n) = 2*T(n/2) + \theta(1)$
   Base case: grab value of i, grab value of j, check i=j, grab value of i, find A[i], grab value of A[i], return a[i], $c_1 = 7$
   Recursive step: grab value of j, grab value of I, subtract j – i, divide by 2, floor, grab value of I, add, set equal to k, grab temp1, grab temp2, calculate <, grab value of temp1 or temp2, return it, $c_2 = 13$
   c.

Level	Level number	Total # of recursive executions at this level	Input size to each recursive execution	Work done by each recursive execution, excluding the recursive calls	Total work done by the algorithm at this level
Root	0	1	n	C_2	C_2
One level below root	1	2	n/2	C_2	$2C_2$
Two levels below the root	2	4	n/4	C_2	$4C_2$
The level just above the base case level	$\log_2 n - 1$	n/2	2	C_2	$n/2*C_2$
Base case level	$\log_2 n$	n	1	C_1	nC_1



   d. $T(n) = \theta(n)$

3.

Level	Level number	Total # of recursive executions at this level	Input size to each recursive execution	Work done by each recursive execution, excluding the recursive calls	Total work done by the algorithm at this level
Root	0	1	n	cn	cn
One level below root	1	7	n/8	cn/8	(7/8)*cn
Two levels below the root	2	49	n/64	cn/64	$(7/8)^2$*cn
The level just above the base case level	$\log_8 n - 1$	7^($\log_8 n$-1)	8	cn/8^($\log_8 n$-1)	(7/8)^($\log_8 n$-1)*cn
Base case level	$\log_8 n$	7^($\log_8 n$) or n^($\log_8 7$)	1	c	c*7^($\log_8 n$) or c*n^($\log_8 7$)

T(n) = $\sum_{i=0}^{\log_4 n-1} (\frac{7}{8})^i cn + \theta(n^{\log_8 7}) \leq \sum_{i=0}^{\infty} (\frac{7}{8})^i cn + \theta(n^{\log_8 7}) = \frac{1}{1-\frac{7}{8}} cn + \theta(n^{\log_8 7}) = 8cn + \theta(n^{\log_8 7})$

T(n) = $O(n)$

4.

What to prove: T(n) = O(n), T(n) $\leq$ cn - 3

Base Case: T(1) = 5 $\leq$ c*1 – 3 where c $\geq$ 4

Inductive Hypotheses: T(n/3) $\leq$ cn/3 - 3

Inductive Step: T(n) = 3T(n/3) + 5 $\leq$ 3*cn/3 – 9 + 5 = cn – 4 $\leq$ cn - 3

Value of c: c = 4

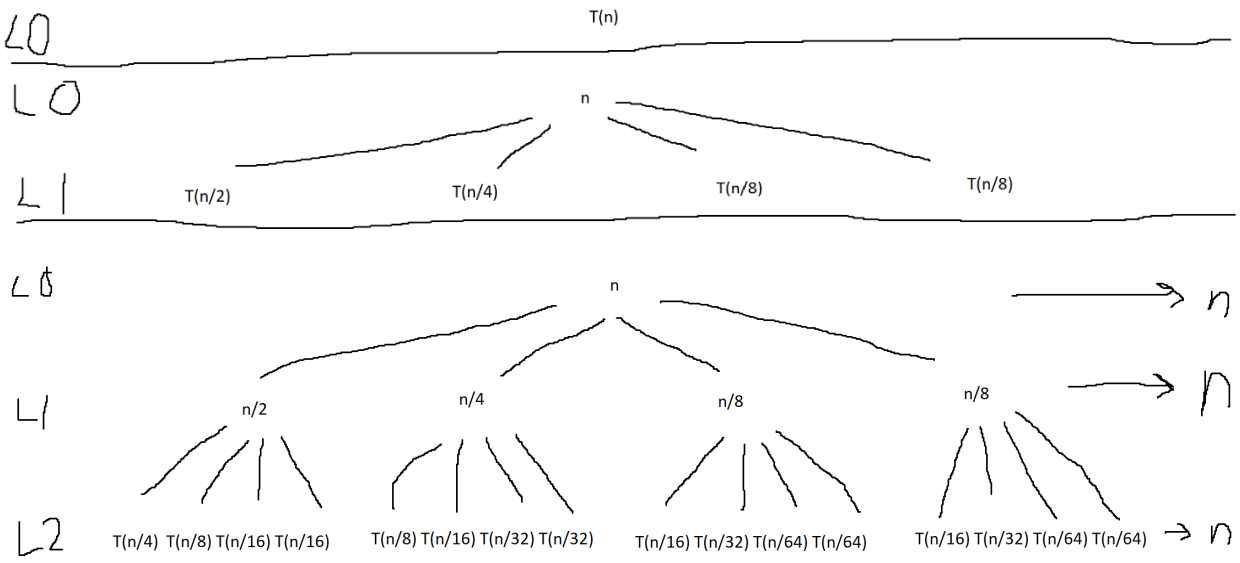
5. f(n) = 2n, s(n) = n^2 + 1, g(n) = n, r(n) = n^2

f(n) – g(n) = 2n – n = n, s(n) – r(n) = n^2 + 1 – n^2 = 1

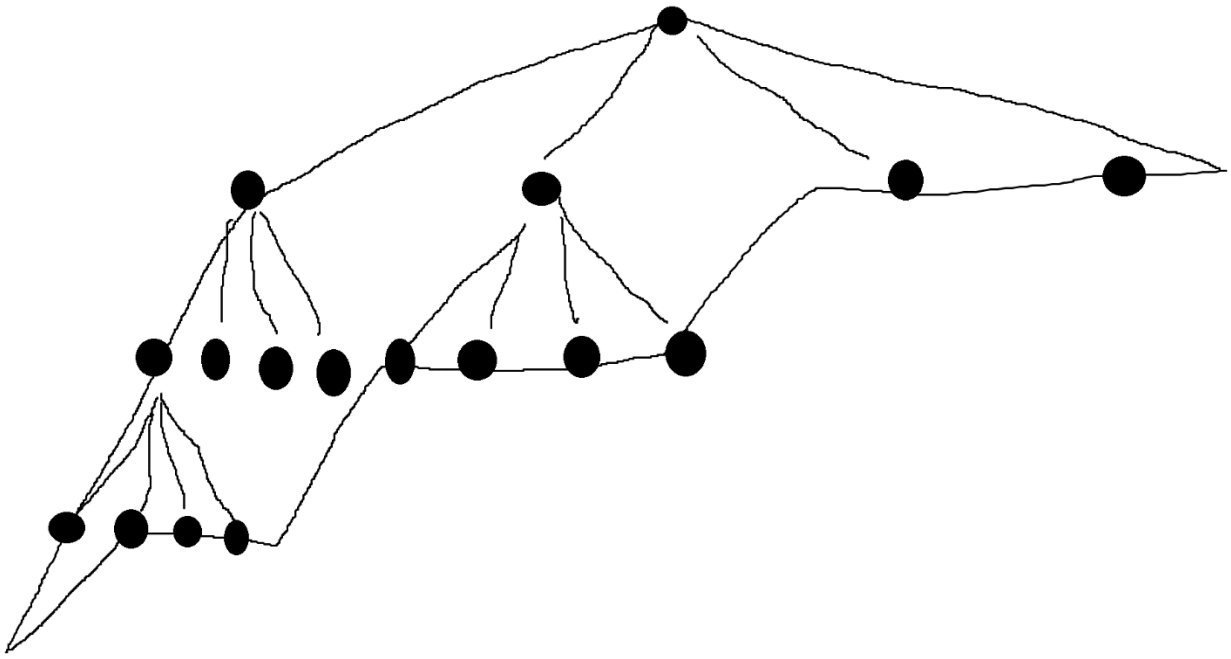
n > 1 for a sufficiently large n, so f(n) – g(n) > s(n) – r(n) for large n, therefore f(n) – g(n) $\neq$ O(s(n) – r(n))

6.

(1)



(2)



(3) shallowest depth = $\log_8 n$

(4) deepest depth = $\log_2 n$

(5) $O(n\log_2 n)$

7.

What to prove: $T(n) = \theta(n*\log_2 n)$, $T(n) \leq cn\lg n$

Base Case: $T(2) = c_0 \leq cn\lg n$, where $c \geq c_0/2$

Inductive Hypotheses: $T(n/2) \leq cn/2\lg(n/2) = cn/2\lg(n) - cn/2\lg(2) = cn/2\lg(n) - cn/2$

$T(n/4) \leq cn/4(\lg n/4) = cn/4\lg(n) - cn/4\lg(4) = cn/4\lg(n) - cn/2$

$T(n/8) \leq cn/8(\lg n/8) = cn/8\lg(n) - cn/8\lg(8) = cn/8\lg(n) - cn3/8$

Inductive Step: $T(n) = T(n/2)+T(n/4)+2T(n/8) + n \leq cn/2\lg(n) - cn/2 + cn/4\lg(n) - cn/2 + 2*cn/8\lg(n) - 2*cn3/8 + n$

$= cn\lg(n) - 7/4cn + n = cn\lg(n) + n(1-7/4c) \leq cn\lg n$, since $n(1-7/4c) < 0$ for $c \geq 1$

Value of c: $c = 2$

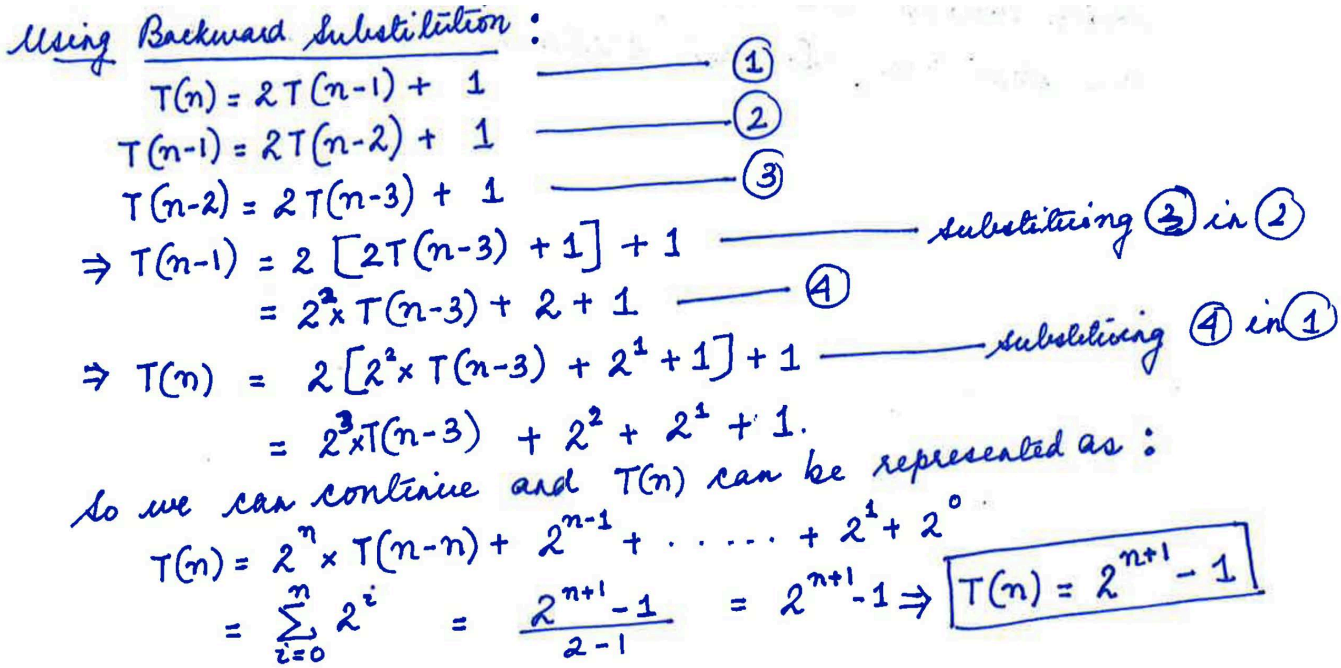
8.a. $a = 2$, $b = 100/99$, $n^{(\log_{100/99}2)} \sim= n^{69}$, $100n = O(n^{69-\varepsilon})$, $\varepsilon = 1$, $T(n) = \theta(n^{(\log_{100/99}2)})$

b. $a = 16$, $b = 2$, $n^{(\log_2 16)} = n^4$, $n^3\lg n = O(n^{4-\varepsilon})$, $\varepsilon = 0.1$, $T(n) = \theta(n^4)$

c. $a = 16$, $b = 4$, $n^{(\log_4 16)} = n^2$, $n^2 = \theta(n^2)$, $T(n) = \theta(n^2\lg n)$

9.

Backward:



10.

Backward:

T(n) = T(n-1) + n/2, T(n-1) = T(n-2) + (n-1)/2, T(n-2) = T(n-3) + (n-2)/2

T(n) = T(n-2) + (n-1)/2 + n/2

T(n) = T(n-3) + (n-2)/2 + (n-1)/2 + n/2

Forward:

T(1) = 1

T(2) = T(1) + 2/2 = 1 + 2/2 = 4/2

T(3) = T(2) + 3/2 = 3/2 + 2/2 + 1 = 7/4

T(4) = T(3) + 4/2 = 4/2 + 3/2 + 2/2 + 1 = 11/4

T(n) = $\sum_{i=2}^{n} \frac{i}{2} + 1 = \frac{1}{2}\sum_{i=2}^{n} i + 1 = \frac{1}{2}\left(\frac{n(n+1)}{2} - 1\right) + 1 = \frac{n(n+1)}{4} + \frac{1}{2}$

11.

What to prove: T(n) = O(nlog^{2}n), T(n) $\leq$ cnlog^{2}n

Base Case: T(2) = 4 $\leq$ c2log^{2}2, where c $\geq$ 2

Inductive Hypotheses: T(n/2) $\leq$ cn/2log^2(n/2)

Inductive Step: T(n) = 2T(floor(n/2)) + 2nlog$_2$n $\leq$ 2T(n/2) + 2nlog$_2$n $\leq$ cnlog^2(n/2) + 2nlog$_2$n

= cn(log$_2$n-log$_2$2)^2 + 2nlog$_2$n = cn(log$_2$n-1)^2 + 2nlog$_2$n = cnlog^{2}n – 2cnlog$_2$n + 1 + 2nlog$_2$n

= cnlog^{2}n + 2nlog$_2$n(1-c) + 1 $\leq$ cnlog^{2}n, since 2nlog$_2$n(1-c) < -1 for c $\geq$ 2

Therefore, since T(n) $\leq$ cnlog^{2}n, T(n) = O(nlog^{2}n)

12. O(g(n)) is used as an upper bound, saying that f(n) is bounded above by n^2. The term “at least” is used to describe a lower bound. Using an upper bound as a lower bound gives no actual information about where our function really lies. This is similar to saying that a function is at most Ω(g(n)).