Hw 2 - Coms 311

By: Joseph Schmidt

1.

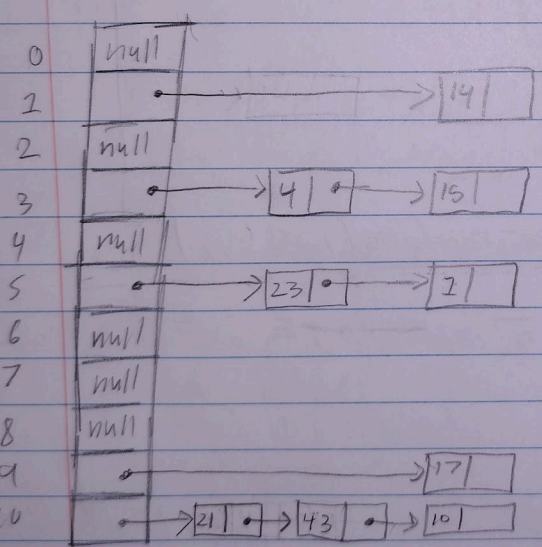
Hw 2

1. $h(x) = 3x+2$, size 11

	1	17	10	15	14	43	4	21	23	24	19	41	42	9	44
h(x)=	5	53	32	47	45	131	14	65	71	74	59	125	128	29	134
% 11	5	9	10	3	1	10	3	10	5	8	4	4	7	7	2

$\alpha = \frac{x}{11} > .75$

$x > 8.25$, once 9 elements put in double size



23, added

now at 9 elements

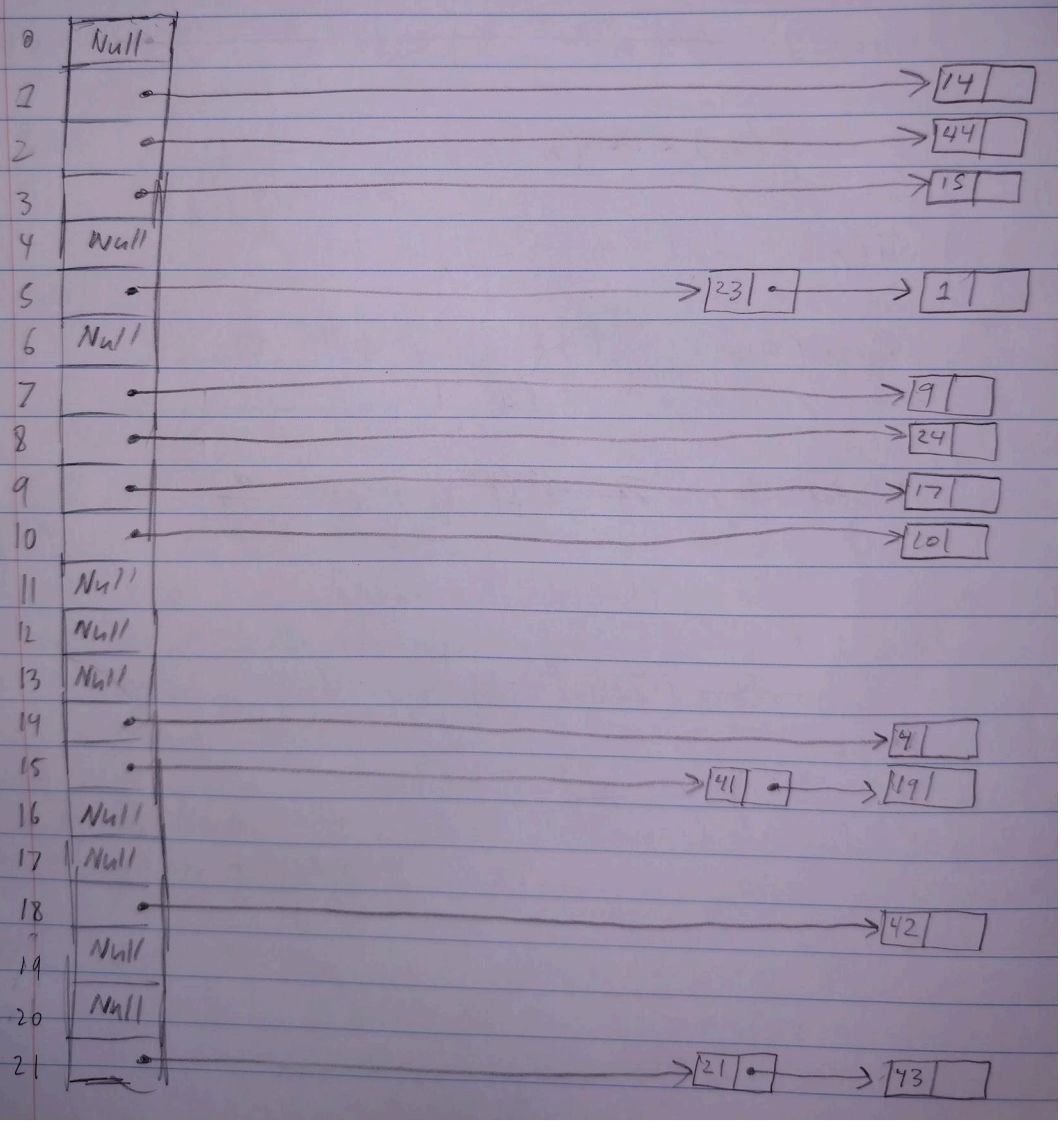
$\alpha = \frac{9}{11} = .81$

Time to resize, must rehash everything or else you can't find items upon look ups

* New elements inserted first

Size doubled, n=22 now

	1	17	10	15	14	43	4	21	23	24	19	41	42	9	44
h(x) =	5	53	32	47	45	131	14	65	71	74	59	125	128	29	134
%22	5	9	10	3	1	21	14	21	5	8	15	15	18	7	2



2.

a)

Assuming n is a power of 2

$$T(n) = 7 * (\frac{n}{2}) + O(1)$$

Case 1: $O(n^{log_2 7 - \varepsilon}), for\ \varepsilon = log_2 7, meaning\ \epsilon > 0, therefore\ T(n) \in \Theta(n^{log_2 7})$

b) Base case:

$$T(1) = 1/6(7* 1^{log_2 7} - 1) = 1/6(7 - 1) = 1$$

Inductive Step:

Take $T(k) = 1/6(7k^{log_2 7} - 1)$ for all k < n we show that

$$T(n) = 7[1/6(7 * \frac{k}{2}^{log_2 7} - 1] + 1$$

$$= 7[1/6(7 * 7^{log_2 \frac{k}{2}} - 1)] + 1$$

$$= 7[1/6(7 * 7^{log_2 k - log_2 2} - 1)] + 1$$

$$= 7[1/6(7 * 7^{log_2 k - 1} - 1)] + 1$$

$$= 7[1/6(7 * 7^{log_2 k} * 7^{-1} - 1)] + 1$$

$$= 7[1/6(7^{log_2 k} - 1)] + 1$$

$$= 7/6(7^{log_2 k} - 1) + 1$$

$$= 7/6(k^{log_2 7} - 1) + 1$$

$$= 7k^{log_2 7}/6 - 7/6 + 1$$

$$= 7k^{log_2 7}/6 - 1/6$$

$$= 1/6(k^{log_2 7} - 1) = T(n)$$

3.

$$\text{(a)}\ T(n) = \begin{cases} 1 & \text{if } n < 2 \\ 4 \cdot T(\frac{n}{2}) + n & \text{if } n \geq 2 \end{cases}$$

Level 0: c*n -- c*n

Level 1: c(n/2) c(n/2) c(n/2) c(n/2) -- 4c*(n/2)
Level 2: c(n/4) … c(n/4) -- 8c*(n/4)
Level …
Level log2n: Theta(1) -- $1 * 4^{log_2 n} = n^{log_2 4} = n^2$

Level	Problem Size	# nodes	Work/node	Total
0	n	1 = 4^0	n	n
1	n/2	4 = 4^1	n/2	2n
2	n/4	16 = 4^2	n/4	4n
…	…	…	…	…
log2n	1	$4^{log_2 n} = n^{log_2 4} = n^2$	$n/2^{log_2 n} = n/n = 1$	n^2

$T(n) = n + 2n + 4n + \ldots + n^2 = n \sum_{i=0}^{log_2 n} 2^i$

$= n(\frac{2^{log_2 n+1}-1}{2-1}) = n(2 * n^{log_2 2} - 1) = n(2n - 1) = 2n^2 - n \in O(n^2)$ **--Final answer 3a**

(b) $T(n) = \begin{cases} 1 & \text{if } n < 2 \\ 2 \cdot T(\frac{n}{2}) + n \log n & \text{if } n \geq 2 \end{cases}$

Level 0: c*nlogn -- c * n/logn
Level 1: c*(n/2)log(n/2) c*(n/2)log(n/2) -- 2c * (n/2)log(n/2)
Level 2: c*(n/4)log(n/4) .. c*(n/4)log(n/4) -- 4c * (n/4)log(n/4)
Level …
Level log2n: Theta(1) … Theta(1) -- $1 * 2^{log_2 n} = n^{log_2 2} = n$

Level	Problem Size	# nodes	Work/node	Total
0	n	1 = 2^0	nlogn	nlogn
1	n/2	2 = 2^1	(n/2)log(n/2)	nlog(n/2)
2	n/4	4 = 2^2	(n/4)log(n/4)	nlog(n/4)
…	…	…	…	…

$log_2 n$	1	$2^{log_2 n} = n^{log_2 2} = n$	1	n

$T(n) = nlogn + nlog(n/2) + nlog(n/4) + \ldots + n$

$T(n) = n \sum_{i=0}^{log_2 n} log(n/2^i) = n(\sum_{i=0}^{log_2 n} (log(n) - log(2^i))) = n(\sum_{i=0}^{log_2 n} log(n) - \sum_{i=0}^{log_2 n} log(2^i))$

$T(n) = n(log_2 n * log(n) - \sum_{i=0}^{log_2 n} ilog(2)) = n(log_2 n * log(n) - log_2 n * log(2))$

$T(n) = nlog_2 n(log(n) - log(2)) = nlog_2 n * log(n/2) \in O(n\, log^2 n)$ **-- Final answer 3b**

$$\text{(c)}\ T(n) = \begin{cases} 1 & \text{if } n < 3 \\ T(\frac{n}{3}) + n & \text{if } n \geq 3 \end{cases}$$

Level 0: c*n
Level 1: c*n/3
Level 2: c*n/9
Level …
Level log3(n) -1 1

Level	Problem Size	# nodes	Work/node	Total
0	n	1=3^0	n	n
1	n/3	3=3^1	n/3	n
2	n/9	9=3^2	n/9	n
…	…	…	…	…
$log_3 n - 1$	1	$3^{log_3 n-1} = 3^{log_3 n} * 3^{-1}$ $= n^{log_3 3} * 3^{-1}$ $= n * 3^{-1}$	$n/3^{log_3 n-1}$ $= n/n^{log_3 3} * 3^{-1}$ $= 3n/n$ $= 3$	n

$T(n) = n + \ldots + n = \sum_{i=0}^{log_3 n-1} n = \sum_{i=1}^{log_3 n} n = nlog_3 n \in O(n\, log\, n)$ **-- Final answer 3c**

$$\text{(d)}\ T(n) = \begin{cases} 1 & \text{if } n < 3 \\ 9 \cdot T(\frac{n}{3}) + n^{2.5} & \text{if } n \geq 3 \end{cases}$$

Level 0: c*n^2.5 -- cn^2.5
Level 1: c*(n/3)^2.5 … c*(n/3)^2.5 --9c(n/3)^2.5
Level 2: c*(n/3)^2.5 … c*(n/3)^2.5 --81c(n/9)^2.5
Level …
Level log3(n) - 1 1

Level	Problem Size	# nodes	Work/node	Total
0	n	1 = 9^0	$n^{2.5}$	n
1	n/3	9 = 9^1	$(n/3)^{2.5}$	$3(n/3)^{2.5}$
2	n/9	81 = 9^2	$(n/9)^{2.5}$	$81(n/9)^{2.5}$
…	…	…	…	…
log3(n) - 1	1	$9^{log_3 n-1} = 9^{log_3 n} * 9^{-1}$ $= n^{log_3 9} * 9^{-1} = n^2 * 9^{-1}$	$(n/3^{log_3 n-1})^{2.5}$ $= n^{2.5}/3^{2.5(log_3 n-1)}$ $= n^{2.5}/3^{2.5log_3 n-2.5}$ $= n^{2.5}/3^{2.5log_3 n} * 3^{-2.5}$ $= n^{2.5}/n^{2.5log_3 3} * 3^{-2.5}$ $= n^{2.5}/n^{2.5} * 3^{-2.5}$ $= 3^{2.5}n^{2.5}/n^{2.5}$ $= 3^{2.5}$	$3^{2.5}n^2/9$

$$T(n) = n^{2.5} + 9(\frac{n}{3})^{2.5} + 81(\frac{n}{9})^{2.5} + \ldots = \sum_{i=0}^{log_3 n-1} 9^i(\frac{n}{3^i})^{2.5} = \sum_{i=0}^{log_3 n-1} 9^i(\frac{n}{3^i})^{2.5}$$

$$= \sum_{i=0}^{log_3 n-1} 9^i \frac{n^{2.5}}{3^{2.5i}} = n^{2.5} \sum_{i=0}^{log_3 n-1} (\frac{9^i}{3^{2.5i}}) = n^{2.5} \sum_{i=0}^{log_3 n-1} (\frac{9}{3^{2.5}})^i = n^{2.5} \sum_{i=0}^{log_3 n-1} (\frac{3^3}{3^{2.5}}) = n^{2.5} \sum_{i=0}^{log_3 n-1} (3^{.5})^i$$

$$= n^{2.5} \sum_{i=0}^{log_3 n-1} 3^{.5i} = \frac{3^{log_3 n-1+1}-1}{3-1} = n^{2.5}(\frac{3^{log_3 n}}{2}) = n^{2.5}(\frac{n^{log_3 3}}{2}) = n^{2.5}(\frac{n}{2})$$

$$= \frac{n^{3.5}}{2} \in O(n^{3.5})$$ **-- Final answer 3d**

$$\text{(e)}\ T(n) = \begin{cases} 1 & \text{if } n < 3 \\ T(n-2) + n^2 & \text{if } n \geq 3 \end{cases}$$

Level 0:	c*n^2
Level 1:	c*(n-2)^2
Level 2:	c*(n-4)^2
Level …	
Level n/2 - 1	c*(n-2(n/2 -1))^2

Level	Problem Size	# Nodes	Work/Node	Total
0	n	1	n^2	n^2
1	n-2	1	$(n-2)^2$	$(n-2)^2$
2	n-4	1	$(n-4)^2$	$(n-4)^2$
…	…	…	…	…
n/2-1	1	1	$(n-2((n/2)-1))^2$ $= (n-n-2))^2$ $= 4$	4, makes n <= 3 So n can equal 2 so 2^2 = 4

$$T(n) = n^2 + (n-2)^2 + (n-4)^2 + \ldots + 4 = \sum_{i=0}^{n/2-1} (n-i)^2$$

$$= \sum_{i=0}^{n/2-1} n^2 - 2ni + i^2 = \sum_{i=0}^{n/2-1} n^2 - \sum_{i=0}^{n/2-1} 2ni + \sum_{i=0}^{n/2-1} i^2 = n^2 \sum_{i=1}^{n/2} 1 - 2n \sum_{i=1}^{n/2} i + \frac{(\frac{n}{2}-1)(\frac{n}{2}-1+1)(2\frac{n}{2}+1)}{6}$$

$$= n^2(\frac{n}{2}) - 2n(\frac{\frac{n}{2}(\frac{n}{2}+1)}{2}) + \frac{(\frac{n}{2}-1)(\frac{n}{2}-1+1)(2\frac{n}{2}+1)}{6}$$

$$= n^2(\frac{n}{2}) - n(\frac{n}{2}(\frac{n}{2}+1)) + \frac{(\frac{n}{2}-1)(\frac{n}{2})(n+1)}{6}$$

$$= n^2(\frac{n}{2}) - n(\frac{n^2}{4} + \frac{n}{2}) + \frac{(\frac{n}{2}-1)(\frac{n^2}{2}+\frac{n}{2})}{6}$$

$$= n^2(\frac{n}{2}) - n(\frac{n^2}{4} + \frac{n}{2}) + \frac{(\frac{n^3}{4}+\frac{n^2}{4}-\frac{n^2}{2}-\frac{n}{2})}{6}$$

$$= \frac{n^3}{2} - \frac{n^3}{4} - \frac{n^2}{2} + \frac{n^3}{24} + \frac{n^2}{24} - \frac{n^2}{12} - \frac{n}{12}$$

$$= \frac{12n^3}{24} - \frac{6n^3}{24} - \frac{12n^2}{24} + \frac{n^3}{24} + \frac{n^2}{24} - \frac{2n^2}{24} - \frac{2n}{24}$$

$= \frac{7n^3}{24} - \frac{13n^2}{24} - \frac{2n}{24}$

$T(n) = \frac{1}{24}(7n^3 - 13n^2 - 2n) \in O(n^3)$ **-- Final answer 3e**

4.

$$\text{(a)}\ T(n) = \begin{cases} 1 & \text{if } n < 2 \\ 4 \cdot T(\frac{n}{2}) + n & \text{if } n \geq 2 \end{cases}$$

Case 1: $O(n^{log_2 4 - \varepsilon}),\ for\ \varepsilon = 1,\ meaning\ \epsilon > 0,\ therefore\ T(n) \in \Theta(n^{log_2 4}) = \Theta(n^2)$

$$\text{(b)}\ T(n) = \begin{cases} 1 & \text{if } n < 2 \\ 2 \cdot T(\frac{n}{2}) + n \log n & \text{if } n \geq 2 \end{cases}$$

Case 1: $O(n^{log_2 2 - \epsilon}),\ no\ such\ \epsilon\ satisfies$

Case 2: $\Theta(n^{log_2 2} log^k n) = \Theta(n\ log^k n),\ take\ k = 1,\ n\ log\ n \in \Theta(n\ log^2 n)$

$$\text{(c)}\ T(n) = \begin{cases} 1 & \text{if } n < 3 \\ 3 \cdot T(\frac{n}{3}) + n & \text{if } n \geq 3 \end{cases}$$

Case 1: $O(n^{log_3 3 - \epsilon}),\ no\ such\ \epsilon\ satisfies$

Case 2: $\Theta(n^{log_3 3} log^k n) = \Theta(n\ log^k n),\ take\ k = 0, T(n) \in \Theta(n\ log\ n)$

$$\text{(d)}\ T(n) = \begin{cases} 1 & \text{if } n < 2 \\ 2 \cdot T(\frac{n}{4}) + \sqrt{n} & \text{if } n \geq 2 \end{cases}$$

Case 1: $O(n^{log_4 2 - \epsilon}),\ no\ such\ \epsilon\ satisfies$

Case 2: $\Theta(n^{log_4 2} log^k n) = \Theta(n^{1/2}\ log^k n),\ take\ k = 0, T(n) \in \Theta(n^{1/2}\ log\ n)$

5.

Algorithm 1 Wacky(A)

```
1: if A.length == 1 then
2:     A[0] += 1;
3: else
4:     int m = ⌊A.length/2⌋;
5:     Wacky (A.sub(0, m));
6:     Wacky (A.sub(m, m));
7:     Wacky (A.sub(0, 1));
```

a)

$T(n) = 2 * T(\frac{n}{2}) + O(n + 1)$ //This is from wacky dividing up the array in two, everytime it is divided the work is divided up evenly with it. The O(n+1) comes from the fact that I assumed substring runs in O(n) time to copy all elements into a new array. The constant +1 comes from line 7 which makes a substring of size 1 which will instantly go into the base case and do a constant operation of changing it's value to += 1.

O(n+1) belongs to O(n) so it can be simplified to:

$T(n) = 2 * T(\frac{n}{2}) + O(n)$

b)

$a = 2\ b = 2$

Case 1:

$f(n) = n^{log_2 2-\epsilon} => n = n^{1-\varepsilon}, \varepsilon\ has\ to\ be\ > 0\ which\ breaks\ case\ 1$

Case 2:

$f(n) = \Theta(n^{log_b a} log^k n)\ for\ k >= 0,\ f(n) = n = \Theta(n^1 * log^k n)\ for\ k = 0$

$Therefore,\ T(n) = \Theta(n\ log\ n)$