1. Sort the below numbers using

1. Counting Sort
2. Radix Sort
3. Insertion Sort
4. Bubble Sort
5. Selection Sort

$$[1, 2, 0, -3, 5, -7, 10]$$
$$[0, 2, 3, 8, 9, 16]$$

(1a) Part 1

[1, 2, 0, -3, 5, -7, 10]

add 7 to all elements

[8, 9, 7, 4, 12, 0, 17]

max element is 17

11				1			1	1	11			1					1
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17

WSt count

2				1			2	3	4			5					6
~~1~~	1	1	1	~~2~~	2	2	~~3~~	~~4~~	~~5~~	5	5	~~6~~	6	6	6	6	~~7~~
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17

[0 4 7 8 9 12 17]

0 1 2 3 4 5 6

Subtract all elements by -7

[-7, -3, 0, 1, 2, 5, 10]

resulting array

[-7, -3, 0, 1, 2, 5, 10]

(1a) Part 2

[0 2 3 8 9 16]

max is 16

1	0	1	1	0	0	0	0	1	1	0	0	0	0	0	0	1
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16

Iterated count

0		1	2					3	4							5
~~1~~	1	~~2~~	~~2~~	3	3	3	3	~~4~~	~~5~~	5	5	5	5	5	5	~~6~~
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16

$wsf[i] = wsf[i] + wsf[i-1]$

$1 \leq i < k+1$

[0 2 3 8 9 16]
 0 1 2 3 4 5

resulting array

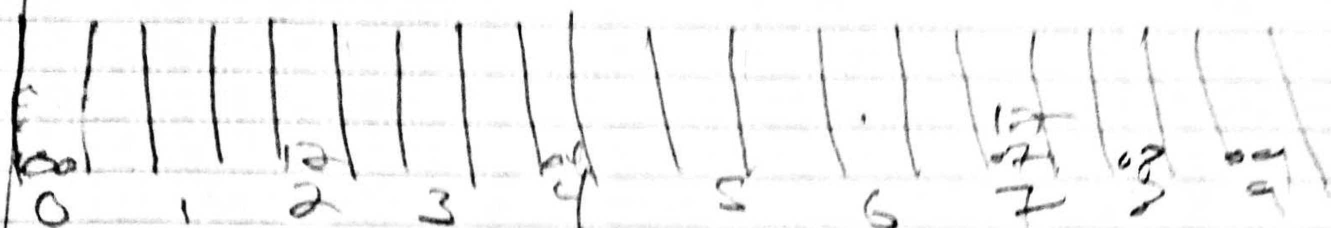
[0 2 3 8 9 16]

11b Part 1

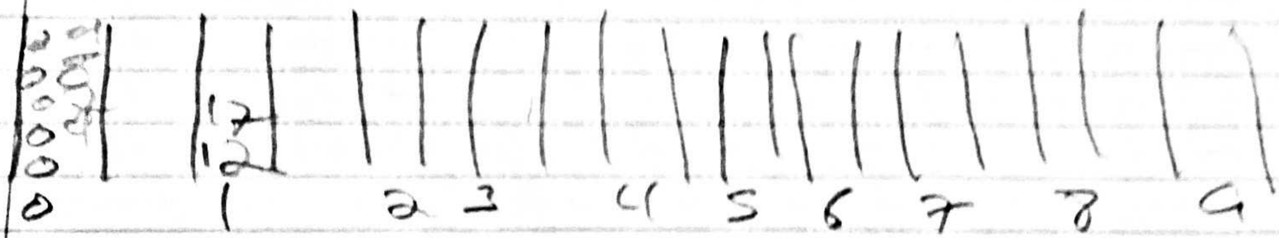
[1, 2, 0, -3, 5, -7, 10]

add 7 to all elements

[8, 9, 7, 4, 12, 0, 17]



0 / 12 / 04 / 07 / 17 / 08 / 09



0 / 04 / 07 / 08 / 09 / 12 / 17

subtracting by -7

-7 / -3 / 0 / 1 / 2 / 5 / 10

resulting array

[-7, -3, 0, 1, 2, 5, 10]

(b) Part 2

[00, 02, 03, 08, 09, 16]

0	1	2	3	4	5	6	7	8	9
00		02	03			16		08	09

00 / 02 / 03 / 16 / 08 / 09

00	1	2	3	4	5	6	7	8	9
09 08 03 02 00	16								

00 / 02 / 03 / 08 / 09 / 16 ✓

resulting array

[0, 2, 3, 8, 9, 16]

1c Part 1

1 2 0 -3 5 -7 10

1 2 0 -3 5 -7 10

0 1 2 -3 5 -7 10

-3 0 1 2 5 -7 10

-3 0 1 2 5 -7 10

-7 -3 0 1 2 5 10

-7	-3	0	1	2	5	10

✓

12 Part 2

6 2 3 8 9 16

0 2 3 8 9 16

0 2 3 8 9 16

0 2 3 8 9 16

0 2 3 8 9 16

0	2	3	8	9	16

1D Part 1

[1 2 0 -3 5 -7 10]

[1 2 0 -3 5 -7 10]

[1 0 2 -3 5 -7 10]

[1 0 -3 2 5 -7 10]

[1 0 -3 2 5 -7 10]

[1 0 -3 2 -7 5 10]

[1 0 -3 2 -7 5 10]

[0 1 -3 2 -7 5 10]

[0 -3 1 2 -7 5 10]

[0 -3 1 2 -7 5 10]

[0 -3 1 -7 2 5 10]

[0 -3 1 -7 2 5 10]

[0 -3 1 -7 2 5 10]

[-3 0 1 -7 2 5 10]

[-3 0 1 -7 2 5 10]

[-3 0 1 -7 2 5 10]

[-3 0 -7 1 2 5 10]

[-3 0 -7 1 2 5 10]

[-3 0 -7 1 2 5 10]

[-3 0 -7 1 2 5 10]

[-3 0 -7 1 2 5 10]

[-3 -7 0 1 2 5 10]

[-3 -7 0 1 2 5 10]

[-3 -7 0 1 2 5 10]

[-3 -7 0 1 2 5 10]

[-3 -7 0 1 2 5 10]

[-7 -3 0 1 2 5 10]

[-7 -3 0 1 2 5 10]

[-7 -3 0 1 2 5 10]

[-7 -3 0 1 2 5 10]

[-7 -3 0 1 2 5 10]

[-7 -3 0 1 2 5 10]

[-7 -3 0 1 2 5 10]

[-7 -3 0 1 2 5 10]

[-7 -3 0 1 2 5 10]

[-7 -3 0 1 2 5 10]

[-7 -3 0 1 2 5 10]

[-7 -3 0 1 2 5 10]

[-7 -3 0 1 2 5 10]

no swaps so stop

resulting array

[-7 -3 0 1 2 5 10]

(1D) Part 2:

[0 2 3 8 9 16]

[0 2 3 8 9 16]

[0 2 3 8 9 16]

[0 2 3 8 9 16]

[0 2 3 8 9 16]

No swaps where needed
so stop

resulting array

[0 2 3 8 9 16]

① e part 1

1	2	0	-3	5	-7	10

idx

new min idx   min new   min new   min new

swap current idx and min idx

-7	2	0	-3	5	1	10

set idx and min idx equal to 1

-7	2	0	-3	5	1	10

idx   min idx   min idx

& min idx

swap current idx and min idx

-7	-3	0	2	5	1	10

idx and min idx equal to 2

-7	-3	0	2	5	1	10

idx

min idx

swap idx and min idx

-7	-3	0	2	5	1	10

idx and min idx equal to 3

-7	-3	0	2	5	1	10

idx

min idx

min idx

swap idx and min idx values

-7	-3	0	1	5	2	10

idx and min idx equal to 4

-7	-3	0	1	5	2	10

idx min idx

min idx

Swap idx and min idx value

-7	-3	0	1	2	5	10

idx and min idx equal to 5

-7	-3	0	1	2	5	10

idx

min idx

Swap idx and min idx values

-7	-3	0	1	2	5	10

idx and non idx equal to 6

-7	-3	0	12	5	10

idx

non idx

swap idx and non idx

-7	-3	0	12	5	10

resulting array

-7	-3	0	1	2	5	10

① e part 2

Set idx and min idx to 0

0	2	3	8	9	16

idx

min idx

Swap idx and min idx values

0	2	3	8	9	16

Set idx and min idx to 1

0	2	3	8	9	16

idx

min idx

Swap idx and min idx values

0	2	3	8	9	16

Set idx and min idx to 2

0	2	3	8	9	6

idx

min idx

Swap idx and min idx.

0	2	3	8	9	6

Set idx and min idx to 3

0	2	3	8	9	6

idx — min idx

Swap idx and min idx values

0	2	3	8	9	6

Set idx and min idx to 4

0	2	3	8	9	6

idx — min idx

Swap idx and min idx values

0	2	3	8	9	6

see idx and min idx to 8

0	2	3	8	9	6

idx

min idx

swap idx and min idx

0	2	3	8	9	6

resulting array

0	2	3	8	9	6

2. What is the running time of Insertion sort if all elements are equal?

Algorithme 1 Insertion Sort

```
Insertion-Sort(a) :
for j = 1 :len(A)
    key = a[j]
    i = j - 1
    while i > 0 and A[i] > key
        A[i + 1] = A[i]
        i = i - 1
    A[i + 1] = key
```

Since the array is sorted the algorithm never enters the while loop and thus does a constant amount of work per iteration, we denote this constant c. $\sum_{i=1}^{n} c = n \cdot c = \Theta(n)$.

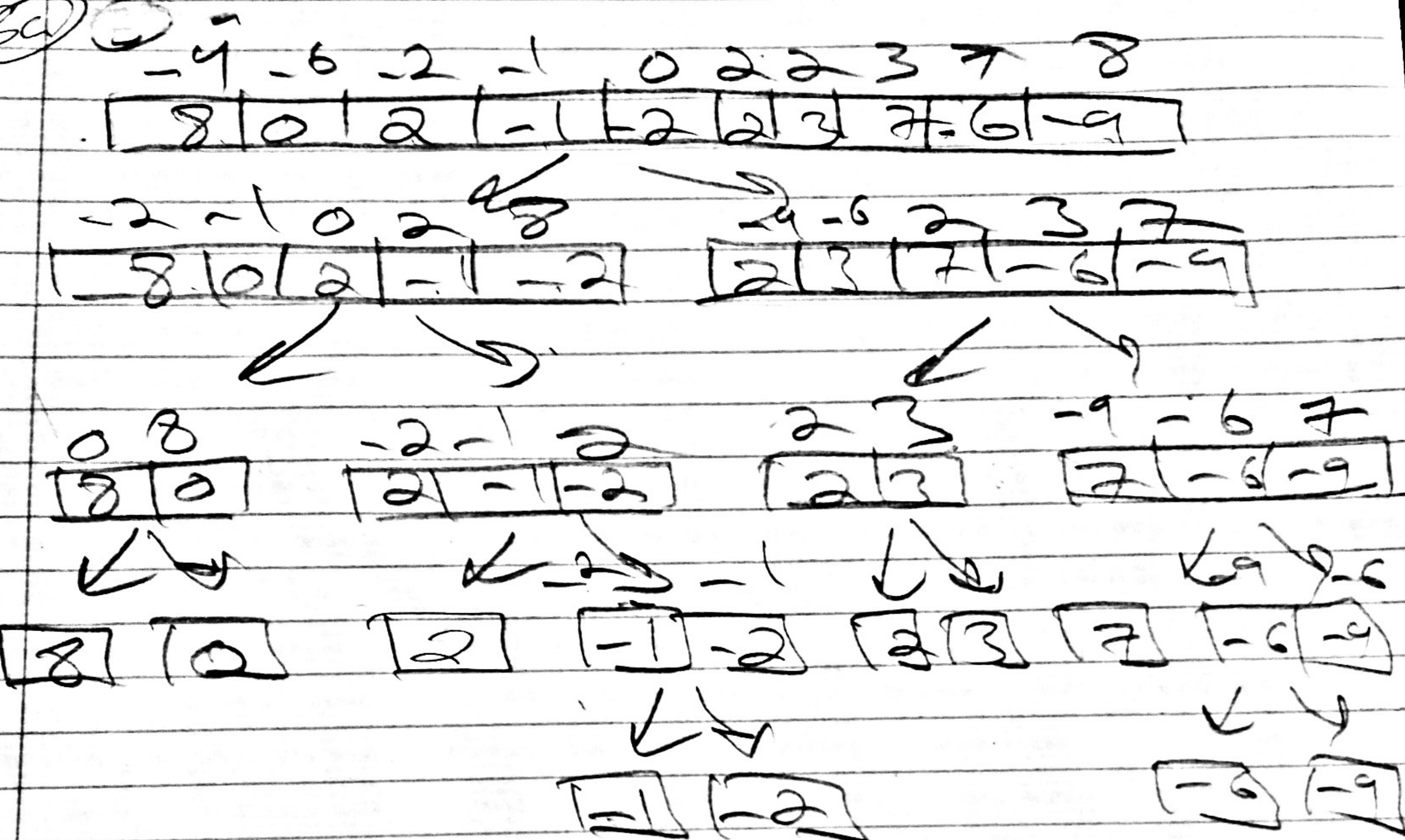
3. Sort the below numbers using:

1. Merge Sort
2. QuickSort

$$[8, 0, 2, -1, -2, 2, 3, 7, -6, -9]$$
$$[19, 7, 6, 3, 2, -1, -7, -18]$$

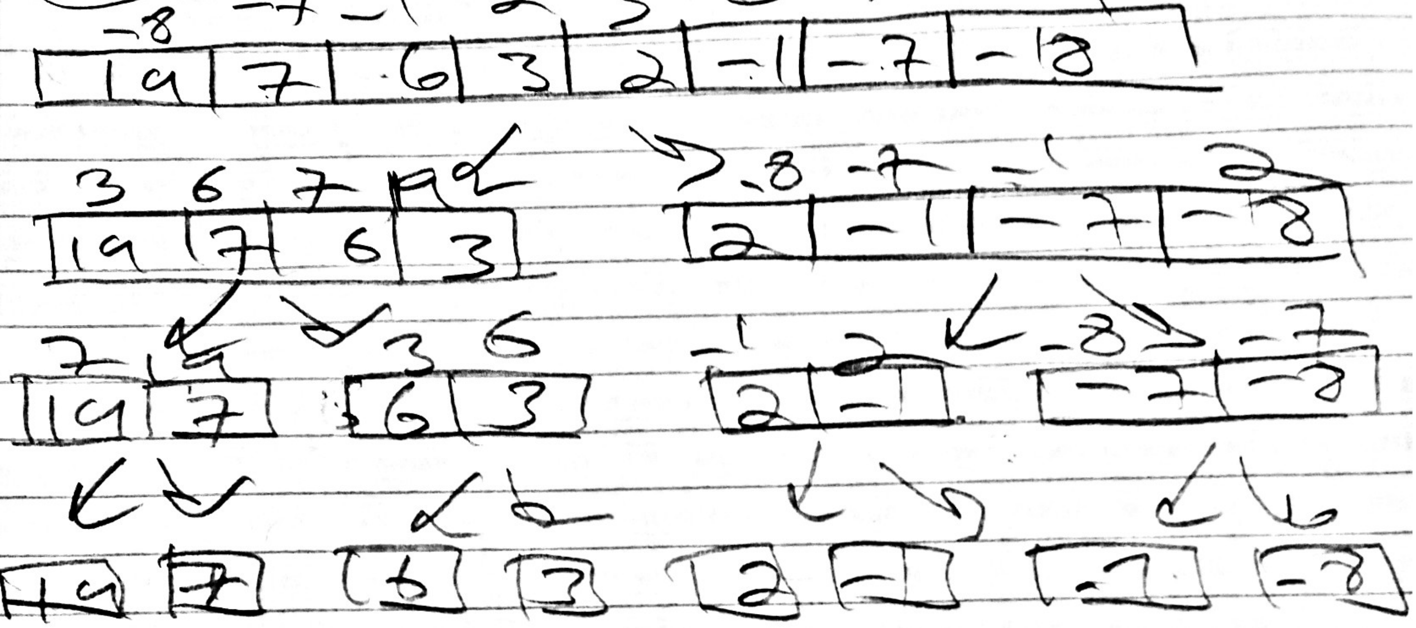
Part 1

3a


-9 -6 -2 -1 0 2 2 3 7 8
8 0 2 -1 -2 2 3 7 -6 -9
-2 -1 0 2 8
8 0 2 -1 -2
-9 -6 2 3 7
2 3 7 -6 -9
0 8
8 0
-2 -1 2
2 -1 -2
2 3
2 3
-9 -6 7
7 -6 -9
8
0
2
-2 -1
-1 -2
2 3
7
-9 -6
-6 -9
-1
-2
-6
-9

Part 2

3a


-8 -7 -1 2 3 6 7 19
19 7 6 3 2 -1 -7 -8
3 6 7 19
19 7 6 3
-8 -7 -1 2
2 -1 -7 -8
7 19
19 7
3 6
6 3
-1 2
2 -1
-8 -7
-7 -8
19 7 6 3 2 -1 -7 -8

3b part 1

8	0	2	-1	-2	2	3	7	-6	-9

$M(8, -2, -9) = -2$

8	0	2	-1	-9	2	3	7	-6	-2

Swap 8 and -6

-6	0	2	-1	-9	2	3	7	8	-2

Swap 0 and -9

-6	-9	2	-1	0	2	3	7	8	-2

Swap 2 and -2

-6	-9

-2

-1	0	2	3	7	8	2

sort manually

quicksort

-9	-6

-1	0	2	3	7	8	2

M(-1, 3, 2) = 2

-1	0	2	3	7	8	2
↑	↑	↑	↑	↑	↑	
✓	✓	✗	✓	✓	✓	

Swap 2 and 2

-1	0		{2}		3	7	8	2

sort naively

quick sort

-1	0

3	7	8	2

M(3, 7, 2) = 3

2	7	8	3
↑	↑	↑	
✓	✗	✓	

Swap ~~7~~ and 3

| 2 |

{3}

| 8 | 7 |

Sort manually

Sort manually

| 2 |

| 7 | 8 |

| -9 | -6 | -2 | -1 | 0 | 2 | 2 | 3 | 7 | 8 |

Combining all the work

3b part 2

19	7	6	3	2	-1	-7	-18

M(19, 3, -18) → 3

19	7	6	-18	2	-11	-7	3

Swap 19 and -7

-7	7	6	-18	2	-11	19	3

Swap 7 and -11

-7	-11	6	-18	2	-7	19	3

Swap 6 and 2

-7	-1	2	-18	6	7	19	3

end partition

-7	-1	2	-18	3	7	19	6

quicksort

sort recursively

6	7	19

-7	-1	2	-18

MC(-7, -1, -18) = -7

-18	-1	2	-7

Stop partitioning

-18	-7	2	-1

sort recursively

sort recursively

-18

-1	2

-18	-7	-1	2	3	6	7	19

Combining work

4. Perform the partitioning algorithm on the below array using the median-of-three heurisitic

$$[1, 2, 6, -3, 20, -61, 7, 8, 19, 100]$$
$$[0, 7, -6, 23, 12, 30, -71, 19]$$

(4)

(a)

1	2	6	-3	20	-61	7	8	19	100
0	1	2	3	4	5	6	7	8	9

M(1, 20, 100) = 20

1	2	6	-3	100	-61	7	8	19
✓	✓	✓	✓	✗				✗

Swap 100 and 19

1	2	6	-3	19	-61	7	8	100	20
				✓	✓	✓	✓	✗	

Swap 100 and 20

1	2	6	-3	19	-61	7	8	20	100

Final partitioned array

4 b

0	7	-6	23	12	30	-7	19
0	1	2	3	4	5	6	7

M(0, 23, 19) = 19

0	7	-6	23	12	30	-7	19
✓	✓	✓	✗			✗	

Swap 23 and -7

0	7	-6	-7	12	30	23	19
			✓	✓	✗	✓	

Swap 30 and 19

0	7	-6	-7	12	19	23	3

final partitioned array

5. What is the worst-case running time of Quicksort if the pivot is randomly chosen as the first element in the array in each recursive call?

We assume that the array is sorted. If so then the first element chosen is the smallest. This leads to the left-subproblem having no elements and the right subarray having $n-1$ elements. This is of the form $T(n) = T(n-1) + O(n)$, which solves to $O(n^2)$.

$T(n) = T(n-1) + O(n)$

level	size	tree
0	n	$n-1$
1	$n-1$	$n-1$
2	$n-2$	$n-2$
$\vdots$		
i	$n-i$	$n-i$
$\vdots$		
k	1	-1

$n-k=1$

$k=n-1$

$$\sum_{i=0}^{k} i = \frac{(k+1)(k)}{2} =$$

$$\frac{(n)(n-1)}{2} = \boxed{O(n^2)}$$

6. Show that the average of a_{left} is $\frac{(n-1)}{2}$ when the input to Quicksort is n distinct elements and the median M is randomly chosen from one of the elements?

$$\frac{\overbrace{(n-1)+(n-2)+(n-3)+\ldots+(n-n)}^{a_{left}\,possible\,sizes}}{\underbrace{n}_{total\,cases}} = \frac{\sum_{i=0}^{n-1} i}{n}$$
$$= \frac{\frac{(n-1)\cdot n}{2}}{n}$$
$$= \frac{(n-1)}{2}$$

7. Calculate the running time of a divide-and-conquer algorithm that requires three recursive calls (each with input-size $\frac{n}{2}$) and $5{\cdot}n^2$ steps that include dividing the input, and using the three solutions to obtain the final solution.

$$\begin{aligned} T(n) &= 3 \cdot T(\frac{n}{2}) + 5 \cdot n^2 \\ &= 3 \cdot T(\frac{n}{2}) + \Theta(n^2) \end{aligned}$$

$T(n) = 3T(\frac{n}{2}) + n^2$

level	size	tree
0	n	n^2
1	$\frac{n}{2}$	$(\frac{n}{2})^2$
$\vdots$	$\vdots$	$\vdots$
k	1	1

$k = \log_2(n)$

$\sum_{i=0}^{k-1} 3(\frac{n}{2^i})^2 + 3^k$

$n^2 \sum_{i=0}^{k-1} (\frac{3}{4})^i + 3^k$

$n^2 \Theta(1) + n^{\log_3(2)}$

$\Theta(n^2)$

$\boxed{T(n) = \Theta(n^2)}$

8. Use a recursion tree for the following algorithms to find the running time.

8

(a) $T(n) = T(\frac{n}{2}) + 1$

Level	Size	Tree
0	n	1
1	$\frac{n}{2}$	1
2	$\frac{n}{2^2}$	1
$\vdots$	$\frac{n}{2^i}$	1
k	1	1

$\frac{n}{2^k} = 1$

$\log_2(n) = k$

$\sum_{i=0}^{k} 1 = (k+1) = (\log_2(n))$

$= \Theta(\log_2(n))$

$\boxed{\Theta(\log_2(n))}$

(b) $T(n) = T(n-1) + 1$

level	size	cost
0	n	1
1	$n-1$	1
2	$n-2$	1
⋮		
k	1	1

$n-k = 1$
$k = n-1$

$$\sum_{i=0}^{k} 1 = (k+1) = (n-1+1) = n$$
$$= \Theta(n)$$

$$\boxed{T(n) = \Theta(n)}$$

(c)

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

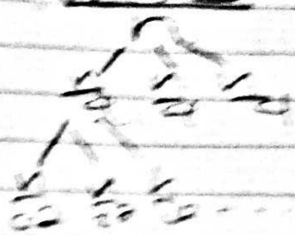
Level	size	tree
0	n	1
1	$n-1$	1 1
2	$n-2$	1 1 1 1
⋮		
k	1	1

$k = n-1$

$$\sum_{i=0}^{k=n-1} 2^i = 2^n - 1 = \Theta(2^n)$$

$$\boxed{T(n) = \Theta(2^n)}$$

1) $T(n) = 3T(\frac{n}{4}) + n$

Level	Size	tree
0	n	
1	$\frac{n}{4}$	
2	$\frac{n}{4^2}$	
k	1	

$k = \log_4(n)$

$$\sum_{i=0}^{k-1} 3^i \frac{n}{4^i} + 3^k$$

$$n \sum_{i=0}^{k-1} \left(\frac{3}{4}\right)^i + n^{\log_4(3)}$$

$$n\,\Theta(1) + n^{\log_4(3)}$$

$$\Theta(n)$$

$$\boxed{T(n) = \Theta(n)}$$

(e) $T(n) = 3T(\frac{n}{3}) + \sqrt{n}$

Level	size	tree
0	n	$\sqrt{n}$
1	$\frac{n}{3}$	$\sqrt{\frac{n}{3}}$ $\sqrt{\frac{n}{3}}$ $\sqrt{\frac{n}{3}}$
2	$\frac{n}{3^2}$	$\sqrt{\frac{n}{3^2}}$
i	$\frac{n}{3^i}$	$\sqrt{\frac{n}{3^i}}$
k	1	1

$k = \log_3(n)$

$$\sum_{i=0}^{k-1} 3^i \sqrt{\frac{n}{3^i}} + 3^k$$

$$\sqrt{n} \sum_{i=0}^{k-1} \left(\frac{1}{3}\right)^i + n$$

$$\sqrt{n}\, \Theta(1) + n$$

$$\Theta(n)$$

$$\boxed{T(n) = \Theta(n)}$$

(f)

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

level	size	tree
0	n	n^2
1	$\frac{n}{2}$	$(\frac{n}{2})^2$
2	$\frac{n}{4}$	$(\frac{n}{4})^2$
$\vdots$		$\vdots$
k	$\frac{n}{2^k}$	$(\frac{n}{2^k})^2$

$k = \log_2(n)$

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

$n^2 \sum_{i=0}^{k-1} (\frac{1}{4})^i + 1$

$n^2 \Theta(1) + 1$

$\Theta(n^2)$

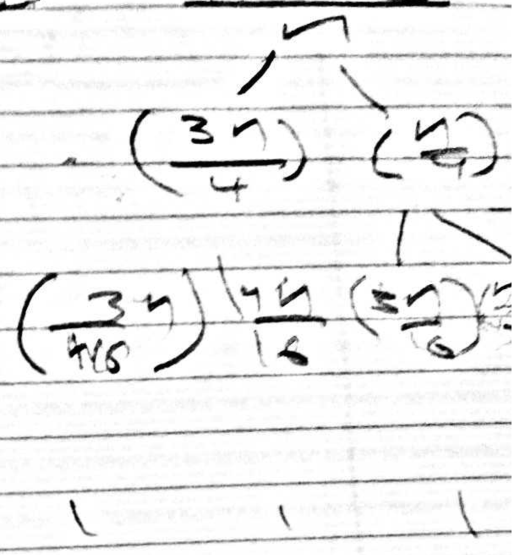
$\boxed{T(n) = \Theta(n^2)}$

(9) $T(n) = T\left(\frac{3n}{4}\right) + T\left(\frac{n}{4}\right) + n$

finding lower bound

Level	Size of subproblems
0	n
1	$\frac{n}{4}$
2	$\frac{n}{4^2}$
$\vdots$	$\vdots$
k	1

tree



$k = \log_4(n)$

$$n \sum_{i=0}^{k} 1 = n(k+1)$$

$$= n(\log_4(n) + 1)$$

$$= \cancel{\Theta}(n \log_4(n))$$

$$\boxed{T(n) = \Omega(n \log(n))}$$

$T(n) = T(\frac{3n}{4}) + T(\frac{n}{4}) + n$

finding upper band

level	size of leftmost	tree
0	$n/4$	n
1		$(\frac{3n}{4})$ $(\frac{n}{4})$
k	1	1 1

$n(\frac{3}{4})^k = 1$

$n = (\frac{4}{3})^k$

$\log_{\frac{4}{3}}(n) = k$

$\sum_{i=0}^{k} n = n(k+1) = n(\log_{\frac{4}{3}}(n)+1)$

$= \Theta(n\log_{\frac{4}{3}}(n))$

$T(n) = O(n\log(n))$

$\boxed{T(n) = O(n\log(n)) \text{ and } \Omega(n\log(n)) \Rightarrow T(n) = \Theta(n\log(n))}$

(h) $T(n) = 4 + (\lfloor \frac{n}{2} \rfloor + 2) + n^2$

Level	Size	tree
0	n	n^2
1	$\lfloor \frac{n}{2} \rfloor + 2$	$(\lfloor \frac{n}{2} \rfloor + 2)^2$
2	$\lfloor \frac{n}{2} \rfloor + 3$	$(\lfloor \frac{n}{4} \rfloor + 3)^2$
3	$\lfloor \frac{n}{2^3} \rfloor + 3$	$(\lfloor \frac{n}{2^3} \rfloor + 3)^2$
k	1	1

$$\lfloor \frac{n}{2^k} \rfloor + 3 = 4$$

$$\frac{n}{2^k} = 1$$

$$\log_2(n) = k$$

$$4^0 n^2 + (\lfloor \frac{n}{2} \rfloor + 2)^2 4^1 + (\lfloor \frac{n}{2^2} \rfloor + 3)^3 + \cdots + 4^k$$

$$= O\left(\sum_{i=0}^{k-1} 4^i \left(\lfloor \frac{n}{2^i} \rfloor + 3\right) + 4^k\right)$$

$$= O\left(\sum_{i=0}^{k-1} 4^i \left(\frac{n}{2^i}\right) + 4^k\right)$$

$$= O\left(n \sum_{i=0}^{k-1} 2^i + 4^k\right)$$

$$= O(n[2^k - 1] + 4^k)$$

$$= O(n \cdot 2^{\log_2(n)} - 1 + n^{\log_2(4)})$$

$$= O(n^2 + n^2)$$

$$= O(n^2)$$

9. Use the formula you learned in class to determine the aymptotic growth of $T(n)$.

9

a

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

$k = \log_2(n)$

$\sum_{i=0}^{k-1} 1 + n^{\log_2(1)} =$

$\sum_{i=0}^{k-1} 1 + 1 =$

$\sum_{i=0}^{k} 1 = (k+1) = (\log_2(n) + 1)$

$= \Theta(\log_2(n))$

$\boxed{T(n) = \Theta(\log_2(n))}$

b $T(n) = 3T(\frac{n}{3}) + n$

$k = \log_3(n)$

$\sum_{i=0}^{k-1} \cancel{3^i \frac{n}{3^i}} + 3^{\log_3(n)} =$

$\sum_{i=0}^{k-1} n + n =$

$nk + n =$

$n \log_3(n) + n =$

$\boxed{\Theta(n \log_2(n))}$

$$T(n) = 4T\left(\frac{n}{3}\right) + n$$

$$k = \log_3(n)$$

$$n \sum_{i=0}^{k-1} 4^i \left(\frac{1}{3}\right)^i + 4^k =$$

$$n\left[\frac{\left(\frac{4}{3}\right)^k - 1}{\left(\frac{4}{3}\right) - 1}\right] + n^{\log_3(4)} =$$

$$n\,\Theta\left(\left(\frac{4}{3}\right)^k\right) + n^{\log_3(4)} =$$

$$n \cdot n^{\log_3\left(\frac{4}{3}\right)} + n^{\log_3(4)}$$

$$n^{\log_3(4) - \log_3(3) + 1} + n^{\log_3(4)} =$$

$$n^{\log_3(4)} + n^{\log_3(4)} =$$

$$\boxed{\Theta\left(n^{\log_3(4)}\right)}$$

(d) $T(n) = 3T\left(\frac{n}{4}\right) + \sqrt{n}$

$k = \log_4(n)$

$$\sum_{i=0}^{k-1} 3^i \sqrt{\frac{n}{4^i}} + 3^k =$$

$$\sqrt{n} \sum_{i=0}^{k-1} \left(\frac{3}{2}\right)^i + n^{\log_4(3)}$$

$$\sqrt{n}\left[\frac{\left(\frac{3}{2}\right)^k - 1}{\left(\frac{3}{2}\right) - 1}\right] + n^{\log_4(3)}$$

$$\sqrt{n}\,\Theta\left(\left(\frac{3}{2}\right)^k\right) + n^{\log_4(3)}$$

$$\sqrt{n} \cdot n^{\log_4\left(\frac{3}{2}\right)} + n^{\log_4(3)} =$$

$$n^{\log_4(3) - \frac{1}{2} + \frac{1}{2}} + n^{\log_4(3)} =$$

$$n^{\log_4(3)} + n^{\log_4(3)} =$$

$$\boxed{\Theta\left(n^{\log_4(3)}\right)}$$

(e)

$T(n) = 5T(\frac{n}{7}) + n^2$

$k = \log_7(n)$

$\sum_{i=0}^{k-1} 5^i (\frac{n}{7^i})^2 + n^{\log_7(5)}$

$n^2 \sum_{i=0}^{k-1} (\frac{5}{49})^i + n^{\log_7(5)}$

$n^2 \Theta(1) + n^{\log_7(5)} =$

$\boxed{\Theta(n^2)}$

(f) $T(n) = 6T(\frac{n}{5}) + n^3$

$k = \log_5(n)$

$\sum_{i=0}^{k-1} 6^i (\frac{n}{5^i})^3 + n^{\log_5(6)}$

$n^3 \sum_{i=0}^{k-1} (\frac{6}{125})^i + n^{\log_5(6)}$

$n^3 \Theta(1) + n^{\log_5(6)}$

$\boxed{\Theta(n^3)}$