1. Compute

1. $\sum_{j=10}^{n}\sum_{i=-5}^{j} 2$

$$\begin{aligned}
\sum_{j=10}^{n}\sum_{i=-5}^{j} 2 &= \sum_{j=10}^{n} 2\cdot(j+5+1)\\
&= \sum_{j=10}^{n} 2\cdot j + \sum_{j=10}^{n} 6\\
&= 2\cdot\sum_{j=10}^{n} j + 6\cdot(n-10+1)\\
&= 2\cdot(\sum_{i=1}^{n} j - \sum_{i=1}^{9} j) + 6\cdot(n-10+1)\\
&= 2\cdot(\frac{n\cdot(n+1)}{2} - \frac{9\cdot 10}{2}) + 6\cdot(n-10+1)\\
&= \Theta(n^2)
\end{aligned}$$

2. $\sum_{i=100}^{n^2} 6^i$

$$\begin{aligned}
\sum_{i=100}^{n^2} 6^i &= \sum_{i=0}^{n^2} 6^i - \sum_{i=0}^{99} 6^i\\
&= \frac{6^{n^2+1}-1}{6-1} - \frac{6^{100}-1}{6-1}
\end{aligned}$$

2. Sort the below numbers using Quicksort

2, 8, 10, 5, 4, 1, 9.

M(2, 5, 9) = 5

2, 8, 10, 9, 4, 1, 5

2 1 10 9, 4, 8, 5

2 1 4 9 10 8 5

2 1 4 5 10 8 9

sort

sort

1	2	4	5	8	9	10

3. Find the 7^{th} least element using the reandom find statistics algorithm. Choose the pivot as the last element in each iteration.

k=7 2, 0, 12, 6, 17, 10, 4, 3, 9, 8

2 0 3 6 17 10 4 12 9 8

2 0 3 6 4 10 17 12 9 8

2 0 3 6 4 8 17 12 9 10

k=7 17, 12, 9, 10

9 12 17 10

9 10 17 12

Randomized

9 is the last element

4. Use the formula you learned in this class to determine the asymptotic growth of:

$$T(n) = 10 \cdot T(\frac{n}{25}) + \sqrt{n}$$

$$\begin{aligned}
\sum_{i=0}^{k-1}(\frac{n}{25^i})^{\frac{1}{2}} + 10^k &= \sqrt{n} \cdot \sum_{i=0}^{k-1}(\frac{1}{5})^i + n^{log_{25}(10)} \\
&= \sqrt{n}\Theta(1) + n^{log_{25}(10)} \\
&= \sqrt{n} \cdot \Theta(1) + n^{log_{25}(10)} \\
&= \Theta(n^{log_{25}(10)})
\end{aligned}$$

5. Use induction to prove that $1 + 2 + 2^2 + \ldots + 2^h = 2^{h+1} - 1$.

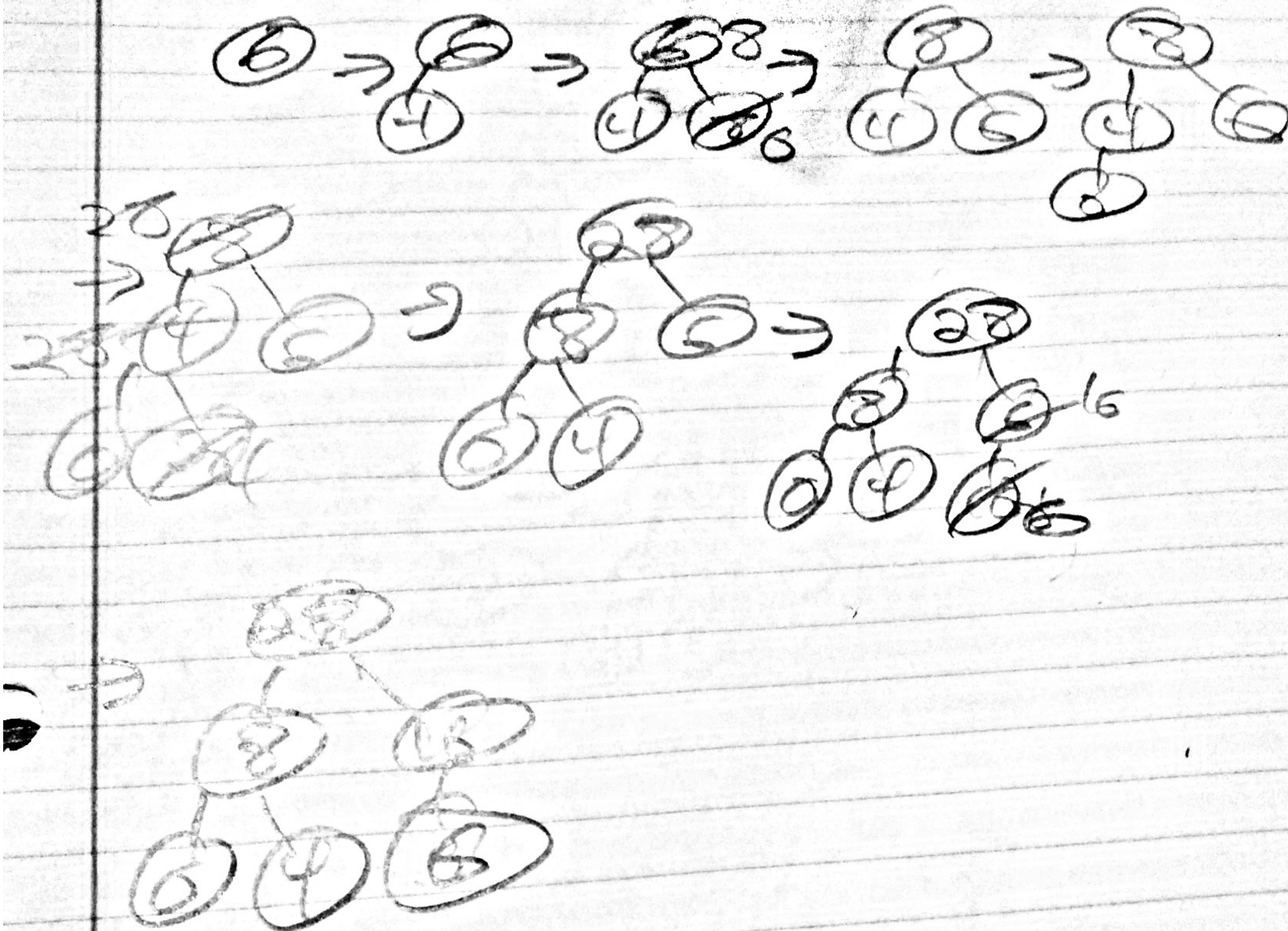
$P(0)$ trivially true. We assume it holds for $P(k)$ for some positive integer k and we show it must hold for the $P(k+1)$ integer.

$$\begin{aligned}
\sum_{i=0}^{k+1} 2^i &= \sum_{i=0}^{k} 2^i + 2^{k+1} \\
&= 2^{k+1} - 1 + 2^{k+1} \\
&= 2^{(k+1)+1} - 1
\end{aligned}$$

6. Make the max heap by successive insertions into an initially max heap. Re-draw the heap each time an insertion causes one or more swaps.

$$6, 14, 8, 0, 28, 16$$

6, 4, 8, 0, 28, 16



7. How many leaves does a binary heap of height 10 have?

$$\lceil \frac{10}{2} \rceil \quad = \quad 5$$

8. What is the assumption of max-heapify algorithm?

That the left and right subtrees are maxheaps

9. Explain when the worst-case running time of Quicksort happens and calculate the time complexity for this case.

$$\begin{aligned} T(n) &= T(n-1) + O(n) \\ &= O(n^2) \end{aligned}$$

10. Use a recursion tree for the following algorithm to find the running time.

1. $T(n) = 4 \cdot T(\frac{n}{2}) + n^3$

$$\begin{aligned} \sum_{i=0}^{k-1} 4^i (\frac{n}{2^i})^3 + 4^k &= n^3 \cdot \sum_{i=0}^{k-1} (\frac{4}{8})^i + n^{log_2(4)} \\ &= n^3 \Theta(1) + n^2 \\ &= O(n^3) \end{aligned}$$

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

$$\begin{aligned} \sum_{i=0}^{k-1} log(\frac{n}{2^i}) + 1 &= \sum_{i=0}^{k-1} log(n) - \sum_{i=0}^{k-1} i + 1 \\ &= \Theta(log^2(n)) - \frac{log^2(n) - log(n)}{2} + 1 \\ &= \Theta(log^2(n)) \end{aligned}$$

11.

1. Suggest an algorithm to find the contigous sub-array with the maximum sum

   The maximum sum sub-array problem could be solved with kadane's algorithm

2. Calculate the time complexity of your answer

   The time complexity of the algorithm is $\Theta(n)$

3. Find the MSS of the below array using your suggested algorithm.

   - 4,-6,1,5,-5,2,-1,3

   The mss is 6.

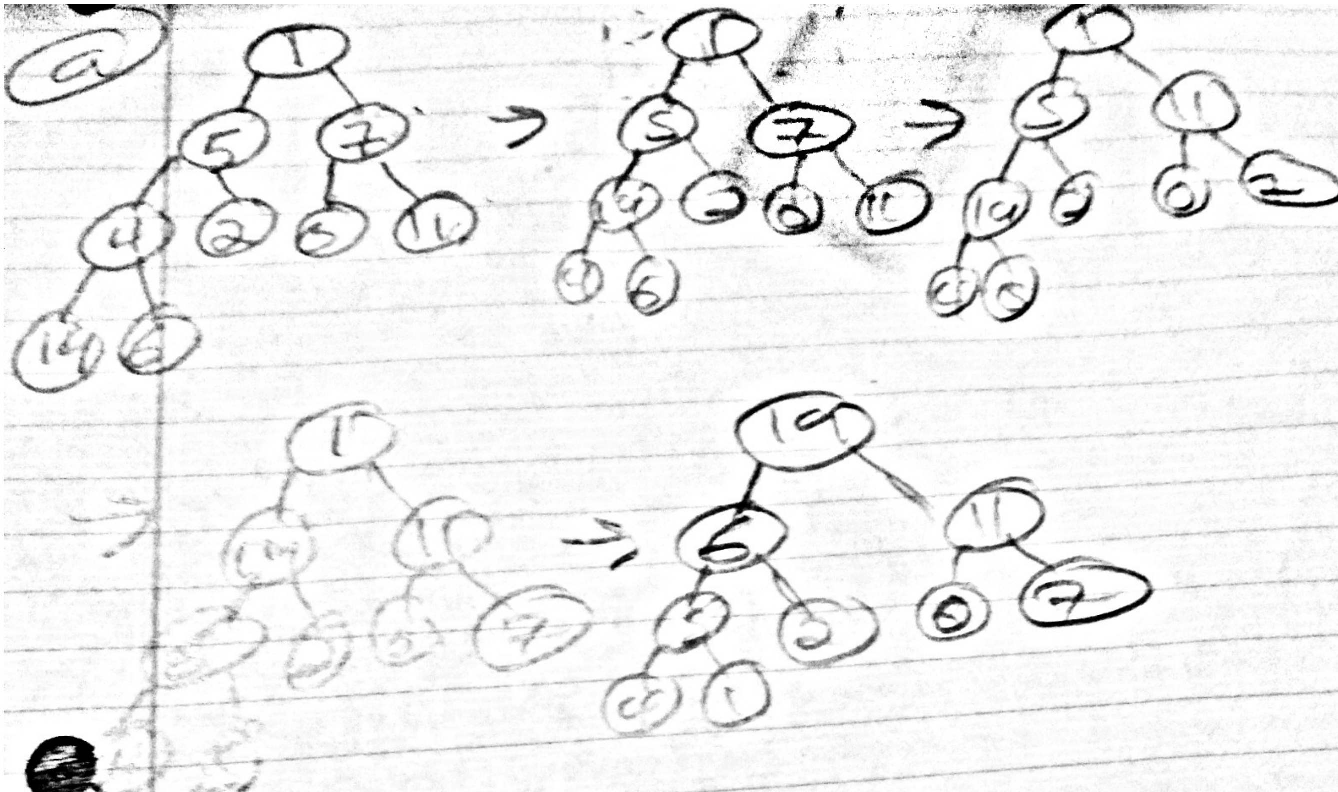
12.

1. Is this array max-heap? If not, change it to the max-heap by putting the elements on a binary tree and applying the max-heapify function.

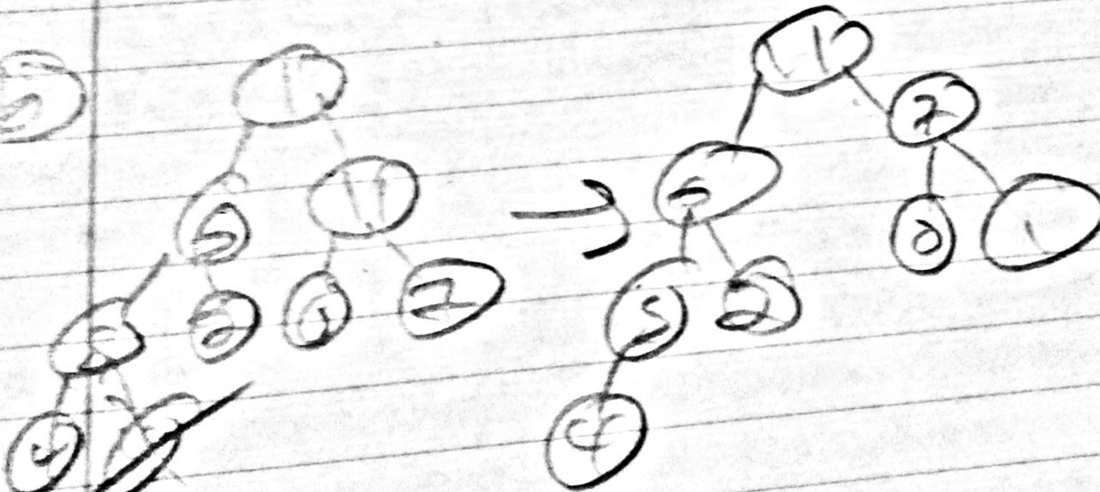
2. Delete the root only once from the max-heap.

   - 1,5,7,4,20,11,19,6

a



b



13. Calculate the running time to find the

1. Minimum

   $\Omega(n)$. It chooses the first element as the pivot, partitions then returns the solution.

2. Median

   $O(n^2)$. $T(n) = O(n) + T(n-1) = O(n^2)$

In a sorted array using find statistics algorithm when the pivot is chosen as the first element in each iteration.s

14. Prove that a binary heap with n elements has height $\lfloor log(n) \rfloor$.

$$\begin{aligned} 2^h \leq & \quad n & \leq 2^{h+1} - 1 \\ 2^h \leq & \quad n & < 2^{h+1} \\ h \leq & \quad log(n) & < h + 1 \end{aligned}$$

we conclude $h = \lfloor log(n) \rfloor$.

15. Use a recursion tree for the following algoritms to find the running time.

$$T(n) = 2T(\lfloor \frac{n}{3} \rfloor + 1) + n^3$$

size	nodes	cost
n	1	n^3
$\frac{n}{3}$	2	$2 \cdot (\lfloor \frac{n}{3} \rfloor + 1)^3$
$\vdots$	$\vdots$	$\vdots$
$\frac{n}{3^i}$	2^i	$2^i \cdot (\lfloor \frac{n}{3^i} \rfloor + 1)^3$
$\vdots$	$\vdots$	$\vdots$
$\frac{n}{3^k}$	2^k	$O(1)$

$k = log_3(n)$.

$$\begin{aligned} n^3 + \sum_{i=1}^{k-1} 2^i \cdot (\lfloor \frac{n}{3^i} \rfloor + 1)^3 + 2^k & \leq \sum_{i=0}^{k-1} 2^i \cdot (\lfloor \frac{n}{3^i} \rfloor + 1)^3 + n^{log_3(2)} \\ & = O(n^3 \cdot \sum_{i=0}^{k-1} (\frac{2}{3})^i + n^{log_3(2)}) \\ & = O(n^3) \end{aligned}$$