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

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

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

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

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.

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

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

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

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.

12.