1. Find the

1. 2^{nd} least element
2. 4^{th} least element

Using the random find statisics algorithm (Quickselect)

2. Calculate the worst-case running time of Quickselect.

An example of the worst-case of Quickselect is looking for the n^{th} least element in a sorted array with a length of n. The Quickselect would select the leftmost valid index then perform the partition and if the index is the not the n^{th} element would result in needing to do a recursive call on a subproblem of one less the original size. The recurrence is of the form $T(n) = T(n-1) + O(n)$ and evaluates to $T(n) = O(n^2)$.

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

Lvl	Size	Cost
0	n	n
1	$n-1$	$n-1$
2	$n-2$	$n-2$
k	1	1

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

$$= \Theta(n^2)$$

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

3. Calculate the average-case running time of Quickselect algorithm. Explain.

The recurrence is of the form $T(n) = T(\frac{n}{2}) + O(n)$. This evaluates to $O(n)$.

$T(n) = T(n/2) + O(n)$

level	size	tree
0	n	n
1	$\frac{n}{2}$	$\frac{n}{2}$
2	$\frac{n}{4}$	$\frac{n}{4}$
⋮	⋮	
1	1	1

$k = \log_2(n)$

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

$$n \sum_{i=0}^{k-1} \left(\frac{1}{2}\right)^i + 2^{\log_2(n)} =$$

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

$$O(n)$$

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

4. Explain an algorithm to return the max k numbers from an unsorted array.

5. Suppose a student wrote the below code for the Maximum Subsequence Sum problem. What is the running time of this algorithm?

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

6. Suppose another student who is a better programmer wrote the below code for the Maximum Subsequence Sum Problem. What is the running time of this algorithm?

$$
\begin{aligned}
\sum_{i=0}^{n-1}\sum_{j=i}^{n-1} c &= \sum_{i=0}^{n-1} c \cdot (n-i) \\
&= c \cdot \sum_{i=0}^{n-1} n - c \cdot \sum_{i=0}^{n-1} i \\
&= c \cdot n \cdot n - c \cdot \frac{(n-1)(n)}{2} \\
&= c \cdot \frac{n^2}{2} - c \cdot \frac{n}{2} \\
&= \Theta(n^2)
\end{aligned}
$$

7. Suppose another students decides to solve the Maximum Subsequence problem using the divide and conquer technique. what is the running time fo this algorithm.

The recurrence of the algorithm will be of the form $T(n) = 2 \cdot T(\frac{n}{2}) + O(n)$, this evaluates to $O(n \cdot log(n))$.

$$T(n) = 2 \cdot T\left(\frac{n}{2}\right) + O(n)$$

level	size	tree	nodes	row sum
0	n	n	1	n
1	$\frac{n}{2}$	$\frac{n}{2}$ $\frac{n}{2}$	2	n
$\vdots$				
i	$\frac{n}{2^i}$	$\frac{n}{2^i}$	2^i	n
$\vdots$				
k	1	1	2^k	n

$k = \log_2(n)$

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

$$= n(\log_2(n)+1)$$

$$= O(n\log_2(n))$$

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

8. Suppose an extraordinary student decides to solve the Maximum Subsequence Sum problem. using the below code. What is the running time of this algorithm.

$$\begin{aligned}\sum_{i=0}^{n-1} c &= c \cdot (n - 1 + 1) \\ &= c \cdot n \\ &= \Theta(n)\end{aligned}$$