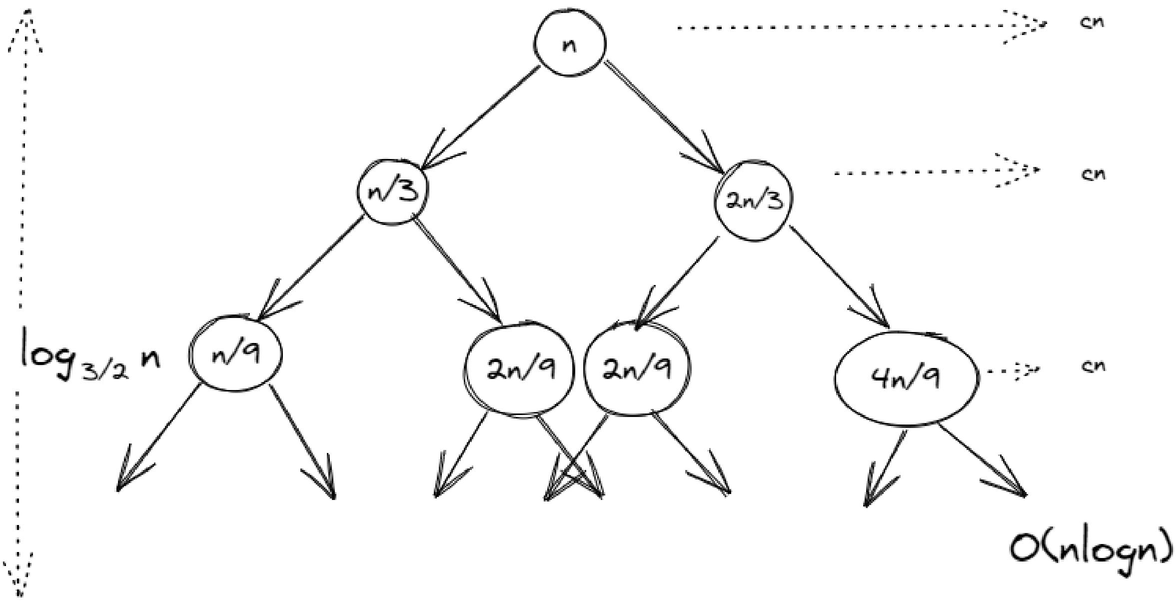
Q1

(a)

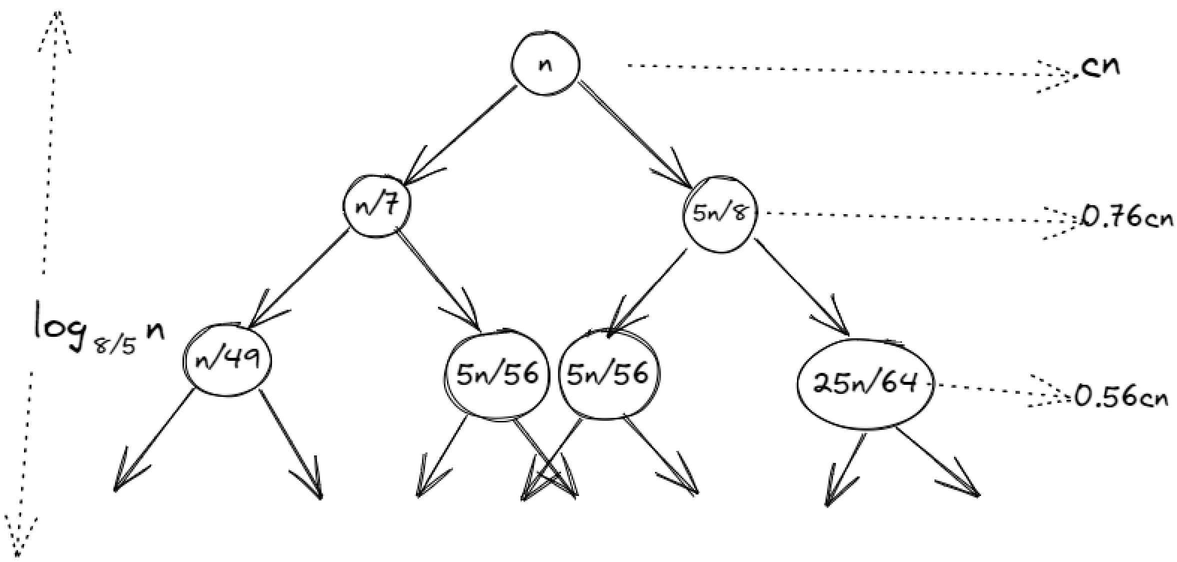
$T(n) = T(\frac{n}{3}) + T(\frac{2n}{3} + 4) + O(n)$



$$
\begin{aligned}
T(n) &> c\lceil \frac{n}{3} \rceil + c(\frac{2n}{3} + 2) + an \\
&> c\frac{n}{3} + c + 2c\frac{2n}{3} + 2c + an \\
&= cn + 3c + an \\
&> cn \\
&= \omega(n)
\end{aligned}
$$

(b)

$T(n) = T(\frac{n}{7}) + T(\frac{5n}{7} + 8) + O(n)$



$$
\begin{aligned}
T(n) &\le c\lceil \frac{n}{7} \rceil + c(\frac{5n}{7} + 8) + an \\
&\le c\frac{n}{7} + c + 5c\frac{n}{7} + 8c + an \\
&= 6c\frac{n}{7} + 9c + an \\
&= cn + (-c\frac{n}{7} + 9c + an) \\
&\le cn \\
&= O(n)
\end{aligned}
$$

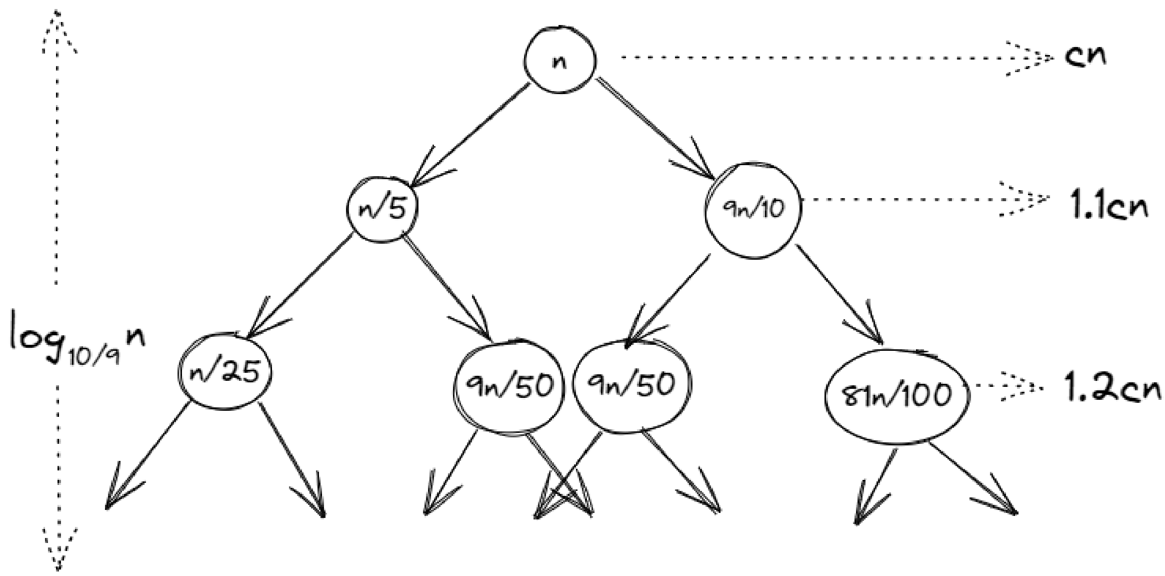
Q2

(a)

Because the pivot is the median of $A[1 \ldots n/5]$, there are at least $n/10$ elements of A greater than the pivot and $n/10$ less than the pivot.

(b)

$T(n) \le T(\lceil \frac{n}{5} \rceil) + T(\frac{9n}{10}) + O(n)$



(c)

$$
\begin{aligned}
T(n) &= T(\frac{n}{5}) + T(\frac{9n}{10}) + O(n) \\
&\le d(\frac{n}{5})lg(\frac{n}{5}) + d(\frac{9n}{10})lg(\frac{9n}{10}) + an \\
&= d(\frac{11n}{10})lgn - d(\frac{n}{5}) + d(\frac{9n}{10})lg9 - d(\frac{9n}{10})lg10 + an \\
&< d(\frac{11n}{10})lgn - d(\frac{n}{5})lg5 + an \\
&= nlgn
\end{aligned}
$$

Q3

(a)

$$T(n) = 4T(n/2) + n^2$$

We have $a = 4, b = 2, f(n) = n^2$. We get:

$$f(n) = n^2 = \Theta(n^{\log_b a}) = \Theta(n^2)$$

Thus, according to the Case (b) of the **Master Theorem**, we have $T(n) = \Theta(n^2 \log n)$.

(b)

$$T(n) = 2T(n/2) + \log n$$

We have $a = 2, b = 2, f(n) = \log n$. We get:

$$f(n) = \log n = \mathcal{O}(n^{\log_b a}) = \mathcal{O}(n)$$

Thus, according to the Case (a) of the **Master Theorem**, we have $T(n) = \Theta(n)$.[**Akra–Bazzi method**]

(c)

$$T(n) = 0.2T(n/2) + n$$

We have $a = 0.2, b = 2, f(n) = n$. $a < 1$ cannot have less than one sub problem. So, it cannot use Master Theorem.

(d)

$$T(n) = 8T(n/2) + 2^n$$

We have $a = 8, b = 2, f(n) = 2^n$. $c = \log_b a = 3$. $f(n)$ isn't a polynomial and $f(n) \notin \Theta(n^3 \log^{k+1} n)$.

So, it can't use MT.

(e)

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

We have $a = 2, b = 2, c = \log_b a = 1, f(n) = \frac{n}{\log n}$.

The difference between $f(n)$ and $n^c = n$ can be expressed with the ratio:

$$\frac{f(n)}{n^{\log_b a}} = \frac{n/\log n}{n} = \frac{1}{\log n}$$

the difference is not polynomial and the basic form of the Master Theorem does not apply.

Q4

(a)

We maintain l,r to represent the current left and right intervals. Each time we divide it so that $[l, i]$ is less than or equal to $pivot$ and $[i+1, r]$ is greater than or equal to $pivot$. By determining the size of the left interval and thus deciding whether to operate on the left half of the interval or on the right half of the interval.

Thus the time complexity of the algorithm is.

$$N + \frac{N}{2} + \frac{N}{4} + \cdots 1 = 2N - 1$$
$$\mathcal{O}(Alg) = \mathcal{O}(N)$$

The pseudocode is as follows.

```
function qselect (arr, k, l, r) -> 1Darray:
    i <- l
    j <- r
    while i < j:
        while (i < j and arr[j] >= arr[l]) j <- j - 1
        while (i < j and arr[i] <= arr[l]) i <- i + 1
        swap the arr[i], arr[j].

    swap the arr[i], arr[j]

    if i > k:
        return qselect(arr, k, l, i - 1)
    if i < k:
        return qselect(arr, k, i + 1, r)

    initial the `res` array
    `res` is equal to the first `k` element in arr
    return res
```

(b)

Use part (a) to find all the *k*th smallest elements. Which take $\mathcal{O}(n)$.

Then we change from considering the size of the left interval to considering the size of the right interval.

If the size of the right interval is larger than $k - l + 1$, we recurse the right interval, otherwise we recurse the left interval, with the same main flow as the algorithm in (a).

In this way, we actually call the algorithm in (a) twice, so the time complexity is $\mathcal{O}(n)$.