1. In class you learn how to find the number that has rank *r* among *n* elements, in $\Theta(n)$ time, using the classic deterministic Selection algorithm with groups of size 5.

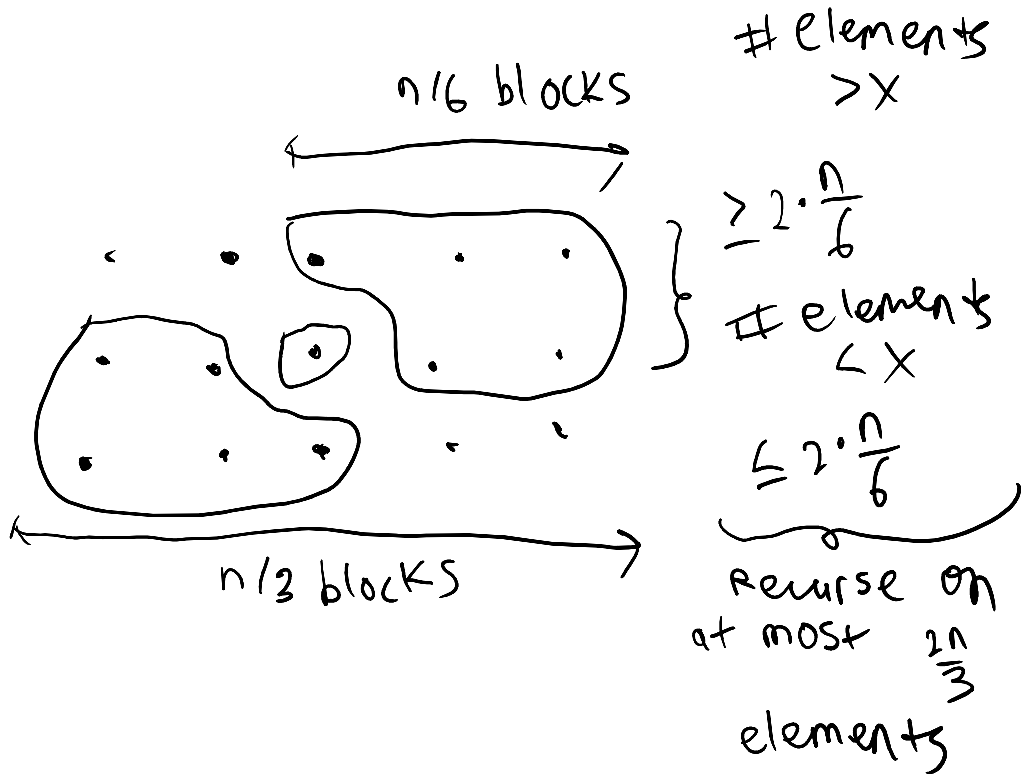
a) Show what happens if we form groups of size 3 instead.

The *n* elements will be segmented into groups of size 3, giving us the expected load factor of n/3. To find the medians for the representatives, we can do a non-recursive search for each block of size 3. For each block of size 3, we can find and copy the median representative in $\frac{n}{3} * \theta(1) = \theta(n)$ time.

To find the median of medians x, the medians would have to be searched recursively which has a time complexity dependent on the size of the groups, providing the time complexity of $T\left(\frac{n}{3}\right)$ to find the median of medians due to the groups of size 3.

Using the median of medians, there is a choice to search left or right of the pivot in order to determine the location of rank k. This search would have to be done recursively with its direction depending on the size of rank k.

In finding the number that is greater than or less than the rank r, the worst-case scenario would be searching half of the elements greater than the median for each group. For size 3 groups, 2 elements would have to be searched for and discarded when searching for any numbers belonging under rank k.



$$Out\ of \frac{n}{3}\ blocks, they\ would\ have\ to\ discard\ 2\ elements\ in\ the\ worst - case\ scenario$$

$$This\ means\ searching\ for \frac{n}{3}\ elements\ while\ discarding \frac{2n}{3}\ elements\ for\ each\ group$$

$$T\left(\frac{n}{3}\right)(Searching\ for\ an\ element\ based\ on\ x) + T\left(\frac{2}{3}\right)\ (Discarding\ 2\ elements)$$

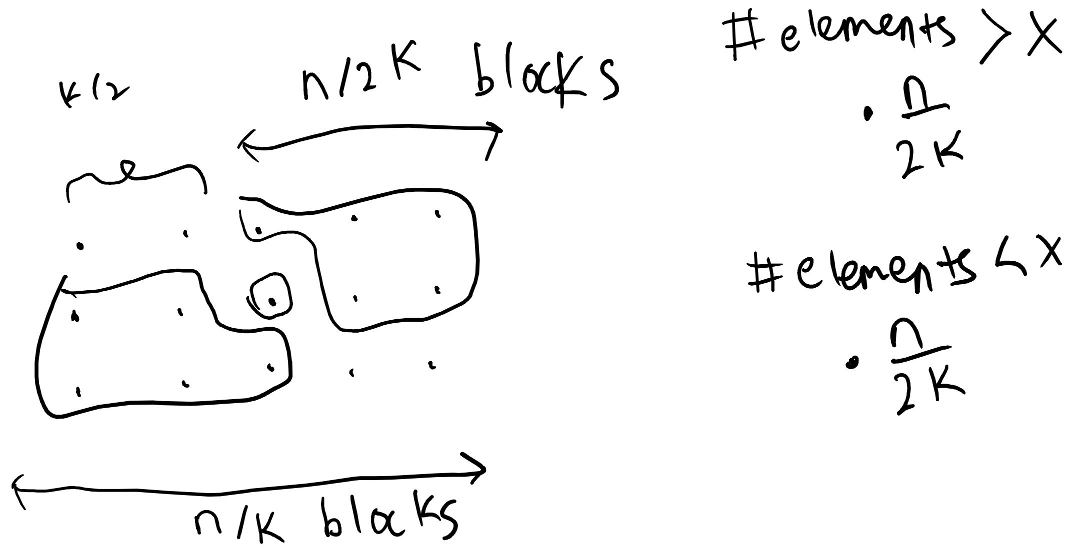
- The time to copy the representatives still remains at θ(n)
- The time to take the median of the representatives is based on the size 3 groups: $T\left(\frac{n}{3}\right)$

$$Prune\ and\ Search\text{: } T(n) = \theta(n) + T\left(\frac{n}{x}\right) + T\left(\frac{n}{b}\right)$$

$$T(n) = \theta(n) + T\left(\frac{n}{3}\right) + T\left(\frac{2n}{3}\right)$$

Using exaggerate and simplify along with the master method, this would be case 2, meaning the deterministic Selection algorithm will be θ(n log n) which is larger than O(n) time.

b) Show what happens if we use groups of size k, where k represents an odd integer greater than 5. Even though k = O(1), try not to remove it from the final calculation.



When initializing a series of representatives for each block of size k, the representatives will be created in O(n) time. The median of representatives will be recursively searched for based on the size of k, meaning that it will take $T\left(\frac{n}{k}\right)$ to calculate the median representatives based on groups of size k.

There are $\frac{n}{4k}$ blocks and the amount of elements discarded will be based on the size of k. Using the median of medians, searching for numbers of rank k would discard one side of the median of medians whether or not k is greater than or less than x. There would be $\frac{k}{2}$ columns that would be discarded when recursively searching the median of medians. For the other $\frac{k}{2}$ columns, we would have to search through $\frac{n}{2k}$ elements that remain in one of the quadrants bordering the median of medians. The worst-case time complexity would mean discarding the lower quadrant and having to search the rest of the quadrants greater than the median of medians.

$$\frac{k}{2} * \frac{n}{2k} \text{ elements would need to be recursively searched through and discarded}$$

$$T(n) = \theta(n) + T\left(\frac{n}{x}\right) + T\left(\frac{n}{b}\right)$$

$$\text{One partition will be} \frac{n}{k} \text{ to account for the search to find rank } k$$

$$n - \frac{n}{4k} \text{ to account for searching the remaining quadrants in the worst} - \text{case scenario}$$

$$T(n) = \theta(n) + T\left(\frac{n}{k}\right) + T\left(n - \frac{n}{2k} * \frac{k}{2}\right)$$

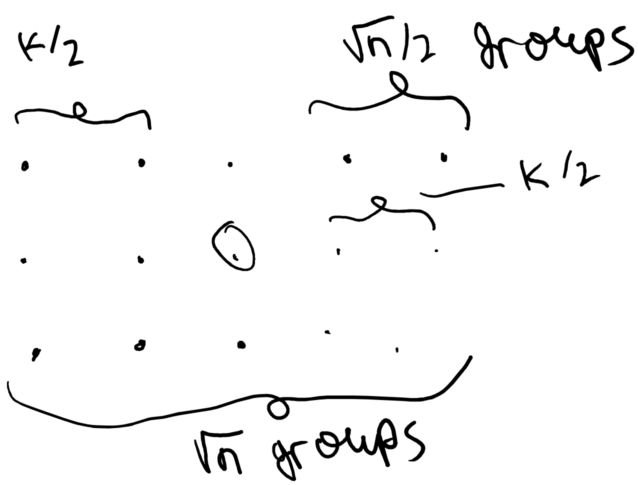
$$T(n) = \theta(n) + T\left(\frac{n}{k}\right) + T\left(n - \frac{n}{4}\right)$$

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

$$\text{Compared to groups of size } 5\text{: } T(n) \leq \theta(n) + T\left(\frac{n}{5}\right) + T\left(\frac{3n}{4}\right)$$

If k > 5, then $T\left(\frac{n}{k}\right)$ will be greater than size 5's $T\left(\frac{n}{5}\right)$ which will lower the needed to compute the deterministic selection sort algorithm.

c) See what happens if we use √ n groups of size √ n, or 5 groups of size n/5. For these cases, you might want to adapt the algorithm slightly when trying to optimize.



1. Finding x to form the partitions out of n elements

Time to run a selection on n elements means drawing a set of representatives for each block of size √n.

We can't find the representatives in $\theta(1)$ time because we're dividing n into a variable amount √n groups. We would have to use a form of sorting algorithm to search for and find the representatives of the medians, but sorting through an array of partitions would take $O(n \log n)$ time for an array of size n. For

√n groups each, it would take $O(\sqrt{n}\ log\ \sqrt{n})$ time to sort through √n groups which becomes $O(\sqrt{n}\ log\ n)$ time for a group of size √n. This makes non-recursive work take more time for √n groups.

Attempting to gather the medians through recursive searching, with √n groups, we're searching through each group θ($\sqrt{n}$) * $\sqrt{n}$ times = $\sqrt{n} * T(\sqrt{n})$ times to gather the representatives of the medians.

2. Copying the representatives of each block to the list and partitioning x

The process of copying the median representatives is based on overall size of n, meaning that it would take O(n) times to copy the representatives from the list of n.

3. Recursively searching left or right

When using a large enough n, the recursion to find a rank in n begins at $\frac{n}{4}$ elements and goes up to a maximum possible amount of $\frac{3n}{4}$ elements. With the worst-case scenario, recursive searching will discard $\frac{n}{4}$ elements and have to search up to $T\left(\frac{3n}{4}\right)$ elements.

$$T(n) = \theta(n) + T\left(\frac{n}{x}\right) + T\left(\frac{n}{b}\right)$$

$$T(n) \leq \sqrt{n} * T\left(\sqrt{n}\right) + T\left(\frac{3n}{4}\right) + \theta(n)$$

Exaggerating $\sqrt{n} * T\left(\sqrt{n}\right)$ to $n * T(n)$

$$T(n) \leq n * T(n) + T\left(\frac{3n}{4}\right) + \theta(n)$$

The time complexity of this equation does not match O(n) time

In all cases, either prove that we still get O(n) time or prove that we cannot get O(n) time. You may assume that division always produces an integer value.