Problem Definition

**Given** a set of "n" unordered numbers we want to find the "k th" smallest number. (k is an integer between 1 and n).
A Simple Solution

A simple sorting algorithm like heapsort will take Order of $O(n \log_2 n)$ time.

<table>
<thead>
<tr>
<th>Step</th>
<th>Running Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sort $n$ elements using heapsort</td>
<td>$O(n \log_2 n)$</td>
</tr>
<tr>
<td>Return the $k^{th}$ smallest element</td>
<td>$O(1)$</td>
</tr>
<tr>
<td>Total running time</td>
<td>$O(n \log_2 n)$</td>
</tr>
</tbody>
</table>
Linear Time selection algorithm

- Also called Median Finding Algorithm.
- Find \( k^{\text{th}} \) smallest element in \( O(n) \) time in worst case.
- Uses Divide and Conquer strategy.
- Uses elimination in order to cut down the running time substantially.
Steps to solve the problem

- **Step 1:** If $n$ is small, for example $n<6$, just sort and return the $k^{th}$ smallest number in constant time i.e; $O(1)$ time.

- **Step 2:** Group the given number in subsets of 5 in $O(n)$ time.
Step3: Sort each of the group in $O(n)$ time. Find median of each group.

Given a set

(.........2,5,9,19,24,54,5,87,9,10,44,32,21,13,24,18,26,16,19,25,39,47,56,71,91,61,44,28.........) having n elements.
Arrange the numbers in groups of five

<table>
<thead>
<tr>
<th></th>
<th>2</th>
<th>54</th>
<th>44</th>
<th>4</th>
<th>25</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>5</td>
<td>5</td>
<td>32</td>
<td>18</td>
<td>39</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>87</td>
<td>21</td>
<td>26</td>
<td>47</td>
</tr>
<tr>
<td></td>
<td>19</td>
<td>9</td>
<td>13</td>
<td>16</td>
<td>56</td>
</tr>
<tr>
<td></td>
<td>24</td>
<td>10</td>
<td>2</td>
<td>19</td>
<td>71</td>
</tr>
</tbody>
</table>
Find median of N/5 groups

Median of each group
Find the Median of each group

Find $m$, the median of medians
Find the sets L and R

- Compare each n-1 elements with the median m and find two sets L and R such that every element in L is smaller than M and every element in R is greater than m.

\[ \frac{3n}{10} < L < \frac{7n}{10} \quad \text{and} \quad \frac{3n}{10} < R < \frac{7n}{10} \]
Description of the Algorithm step

- If n is small, for example n<6, just sort and return the k the smallest number. (Bound time - 7)
- If n>5, then partition the numbers into groups of 5. (Bound time n/5)
- Sort the numbers within each group. Select the middle elements (the medians). (Bound time - 7n/5)
- Call your "Selection" routine recursively to find the median of n/5 medians and call it m. (Bound time - T n/5)
- Compare all n-1 elements with the median of medians m and determine the sets L and R, where L contains all elements <m, and R contains all elements >m. Clearly, the rank of m is r=|L|+1 (|L| is the size or cardinality of L). (Bound time - n)
If $k=r$, then return $m$.

If $k<r$, then return $k^{th}$ smallest of the set $L$. (Bound time $T_{7n/10}$)

If $k>r$, then return $k-r^{th}$ smallest of the set $R$. 
Recursive formula

\[ T(n) = O(n) + T\left(\frac{n}{5}\right) + T\left(\frac{7n}{10}\right) \]

We will solve this equation in order to get the complexity.

We assume that \( T(n) < C \times n \)

\[ T(n) = a \times n + T\left(\frac{n}{5}\right) + T\left(\frac{7n}{10}\right) \]

\[ C \times n \geq T\left(\frac{n}{5}\right) + T\left(\frac{7n}{10}\right) + a \times n \]

\[ C \geq \frac{9C}{10} + a \]

\[ C/10 \geq a \]

\[ C \geq 10a \]

There is such a constant that exists....so \( T(n) = O(n) \)
Why group of 5 why not some other term??

- If we divide elements into groups of 3 then we will have
  \[ T(n) = O(n) + T\left(\frac{n}{3}\right) + T\left(\frac{2n}{3}\right) \] so \( T(n) > O(n) \)…..
- If we divide elements into groups of more than 5, the value of constant 5 will be more, so grouping elements into 5 is the optimal situation.