MEDlANS (or SELECTION)

INPUT: A list of $n$ real numbers $S$, an integer $k$.
OUTPUT: The $k$th smallest element of $S$, for some $1 \leq k \leq n$.

e.g. $k = 1 \Rightarrow$ MINIMUM and can be computed in $O(n)$ time.

$k = n \Rightarrow$ MAXIMUM and can be computed in $O(n)$ time.

$k = \lfloor \frac{n}{2} \rfloor \Rightarrow$ MEDIAN ELEMENT

How fast we can compute Median? or in general the $k$th smallest?

Easy Solution: $\text{SORT}(A)$
OUTPUT $k$th element in the sorted list.
This takes $O(n \log n)$ time.

OBJECTIVE: $k$th smallest can be computed in $O(n)$ time.

1973 BLUM Time Bounds for Selection
FLOYD PRATT RIVEST TARJAN (or Median of Median algorithm) based on Divide and Conquer.
Selection in Linear Time

Approach: In \( O(n) \) time find an element \( x \) in \( S \), which partition \( S \) as follows:

\[
\begin{array}{c|c|c}
\# \geq (1-x)n & \# \geq (1-x)n \\
< x & x & > x \\
\hline
\# \leq \alpha n & \# \leq \alpha n \\
L & R
\end{array}
\]

Here,

1. Assume that all elements of \( S \) are distinct (this is to make explanation easier).
2. Set \( L \) consists of all elements in \( S \) that are \(< x\).
3. Set \( R \) consists of all elements in \( S \) that are \(> x\).
4. \( \alpha \) is a constant between \( \frac{1}{2} \leq \alpha \leq \frac{1}{2} + \frac{1}{2}\).

We say that \( x \) partitions \( S \) in a BALANCED WAY.
Suppose we can find such an \( x \) that partitions \( S \) in a Balanced Way.

We are looking for \( k^{th} \) smallest element of \( S \).

Question: Is where is the \( k^{th} \) smallest element? Is it \( =x \), or in set \( L \), or in set \( R \).

Answer: That is easy to find.

(a) If \( |L| \geq k \), then \( k^{th} \) element is the \( k^{th} \) element in set \( L \).

(b) If \( |L| = k-1 \), then \( x \) is the \( k^{th} \) element.

(c) If \( |L| < k-1 \), then \( x \) is the \( \frac{k-|L|}{1} \)th element of set \( R \).
Either we find the $k$th-element or we have to recursively find an appropriate element in either the set $L$ or $R$.

If $T(n) =$ time to find the $k$th smallest element then

$$T(n) = T(\alpha n) + O(n)$$

Since $|L| \leq \alpha n$

$|R| \leq \alpha n$

Time to find an element $x$ that partitions $S$ in a balanced way + Partitioning $S$ in sets $L$ and $R$

Then,

$$T(n) = \frac{1}{1-\alpha} n = O(n)$$

Since $\alpha$ is a constant.

What remains is to show that such an element $x$ can be found in $O(n)$ time.
INPUT: Let \( S = \{a_1, a_2, \ldots, a_n\} \) be \( n \)-numbers.

OUTPUT: \( k \)-th smallest element.

Assumption: Assume all elements of \( S \) are distinct (this is just to simplify the discussion)

**Pseudo code**

Step 1: Divide the input into \( \lceil \frac{n}{5} \rceil \) groups, where each group contains 5-elements, but the last one may have fewer than 5.

\[
\begin{array}{c}
\frac{a_1, a_2, a_3, a_4, a_5}{\text{group 1}} & \frac{a_6, a_7, \ldots, a_{10}}{\text{group 2}} & \frac{a_{11}, a_{12}, a_{13}}{\text{group 3}} \\
\end{array}
\]

Step 2: For each group \( j \), \( (1 \leq j \leq \lceil \frac{n}{5} \rceil) \), compute the median \( \text{m}_j \) of the \( j \)-th group.

[Since each group has 5-elements, we can sort them and compute its median in \( O(1) \) time.]

Step 3: \( x = \text{median of } m_1, m_2, \ldots, m_{\lceil \frac{n}{5} \rceil} \).

↑ That's what we were trying to find.
Claim: At least \( \frac{3n}{10} \) elements of \( S \) are \( \leq x \).

Equivalently, at most \( \frac{7n}{10} \) elements of \( S \) are \( > x \).

Proof: Since \( x \) is median of \( m_1, m_2, \ldots, m_{[n/5]} \), then 50\% of \( m_j \)'s are \( \leq x \).

\[ \Rightarrow \frac{n}{10} \text{ of } m_j \text{'s are } \leq x. \]

In each of the groups, where \( m_j \) came from, at least 3 elements in that group are \( \leq x \).

\[ \Rightarrow \text{At least } \frac{3n}{10} \text{ elements in } S \text{ are } \leq x. \]

\[ \Rightarrow \text{At most } \frac{7n}{10} \text{ elements in } S \text{ are } > x. \]

Corollary: At most \( \frac{7n}{10} \) elements in \( S \) are \( < x \).

Proof: Repeat the argument of the claim by using the fact that 50\% of \( m_j \)'s are \( \geq x \).
So \( x = \frac{7}{10} \) in Figure 1, for this value of \( x \).

\[
\begin{array}{ccc}
 & L & R \\
< x & x & > x \\
\leq \frac{7}{10} & \leq \frac{7}{10} \\
\end{array}
\]

**Step 4:** Partition elements of \( S \) into \( L \) and \( R \).

\( L = \{ u \mid u \in S \text{ such that } u < x \} \)

\( R = \{ u \mid u \in S \text{ such that } u > x \} \)

**Step 5:**

- If \( |L| = k-1 \): Return \( x \)
  
  If \( |L| \geq k-1 \): Recursively, compute \( k^{th} \) smallest in \( L \).

  If \( |L| < k-1 \): Recursively, compute \((k-|L|-1)^{th}\) smallest in \( R \).
\[ T(n) = O(n) \quad \text{[Step 1]} \]
\[ + O(n) \quad \text{[Step 2]} \]
\[ + T\left(\frac{n}{5}\right) \quad \text{[Step 3]} \]
\[ + O(n) \quad \text{[Step 4]} \]
\[ + T\left(\frac{7}{10}n\right) \quad \text{[Step 5]} \]

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

\text{Claim: } T(n) = O(n).

\text{Proof: } By \text{ induction on } n.

Base case: Easy to verify.

Need to show

\[ T(n) \leq T\left(\frac{n}{5}\right) + T\left(\frac{7}{10}n\right) + cn \leq dn \]

\[ \Rightarrow d \frac{n}{5} + d \frac{7n}{10} + cn \leq dn \]

\[ \frac{1}{I.H} \quad \frac{10}{I.H} \]

\[ \Rightarrow 10c \leq d \]

Since we can always find a constant \( d = 10c \), the claim holds.
Taking care of the fact that the last group may have anywhere from 1 to 5 elements, the true recurrence relation is

\[ T(n) = \begin{cases} 
O(1) & \text{if } n \leq 140 \\
T(\left\lfloor \frac{n}{5} \right\rfloor) + T\left( \frac{7n}{10} + 6 \right) + O(n) & \text{if } n > 140
\end{cases} \]

And by induction one can show that

\[ T(n) \leq d \left\lfloor \frac{n}{5} \right\rfloor + d \left( \frac{7n}{10} + 6 \right) + cn \leq dn \]

\[ \Leftrightarrow \quad d \frac{n}{5} + d + d \left( \frac{7n}{10} + 6 \right) + cn \leq dn \]

\[ \Leftrightarrow \quad \frac{9d}{10} n + 7d + cn \leq dn \]

\[ \Leftrightarrow \quad -\frac{d}{10} n + 7d + cn \leq 0 \]

\[ \Leftrightarrow \quad d \geq \frac{10cn}{n-70}, \quad \text{since } n \geq 140 \quad \Rightarrow \quad d \geq 20c \quad \text{will satisfy } \square. \]
Are there other ways to find $x$.

What if we choose $x$ randomly.

It is obvious that sometimes we may be unlucky (e.g. $x=\text{MIN}$ or $x=\text{MAX}$) or we may be lucky (e.g. $x$ is an element that partitions in a balanced way).

Call $\text{CHOICE of } x$ \textbf{GOOD} if

the element $x$ leaves at least \( \frac{|S|}{4} \) elements $< x$ and at least \( \frac{|S|}{4} \) elements $> x$.

What is the probability of picking such an element $x$?

Easy $= \frac{1}{2}$

Why? Smallest 25\% and largest 25\% elements are not good choice - except that
all elements are good.

⇒ Given a random $x$, it has 50% chance of being good.

⇒ Within two trials we can find on the average we can find a good $x$.

(= How many times we have to toss a coin till we get "Head")

Hence

$$T(n) = \text{Expected Running Time}$$

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

Time to reduce the set to $\frac{3}{4}$ of its size (including on the average two trials).

$$= O(n).$$
The above Expected Running Time Analysis was very sloppy.

We need to use linearity of expectation in a proper way.

In this course we will mainly stay with deterministic algorithms.