Select(S,k)
S a set of n keys, k less than n is the rank of item (kth smallest) to be selected and returned.

0. if \(|S| \leq 5\) return direct solution for kth elt of S.

1. Divide keys into sets of five each, find median of each. Call set of medians M. (See Fig (a)).

2. \(m^* = Select(M, \lceil |M| / 2 \rceil)\)
m* the median of medians (Fig. (b)).

3. Partition: Compare each key in sections A and D of Fig (b) to m*.
   \(S_1 = C \cup \{\text{keys from } A \cup D \text{ smaller than } m^* \}\)
   \(S_2 = B \cup \{\text{keys from } A \cup D \text{ larger than } m^* \}\)
4. Divide and Conquer:
if \((k = |S_1| + 1)\) return \(m^*\) as kth-smallest.

elseif \((k \leq |S_1|)\) return Select\((S_1, k)\).

else return Select\((S_2, k - |S_1| - 1)\).
ANALYSIS of Selection

W(n) is number of key comparisons in worst case with n keys. Assume \( n = 5(2r + 1) \) for some \( r \). Counts per step:

1. Medians of all sets of five keys: \( 6(n/5) \), since if you’re clever can find median of 5 with 6 comps.

2. Recursion: \( W(n/5) \) comparisons.

3. Compare all section A and D keys to \( m^* \): 4r comparisons.

4. Divide and conq. \( W(7r+2) \).
In worst case, all 4r keys in A and D will be on same side of $m^*$ (all $> m^*$ or $< m^*$). B and C have $3r + 2$ elements.

$n = 5(2r + 1)$, so $r \approx n/10$. Thus

$$W(n) \leq 1.2n + W(0.2n) + 0.4n + W(0.7n)$$

$$= 1.6n + W(0.2n) + W(0.7n).$$
\[ W(n) = 1.6n + W(0.2n) + W(0.7n) \]

Unequal-sized subproblems so Master theorem no good. But recursion tree shows row-sums are decreasing geometric series whose ratio is 0.9. Total is \( \Theta \) of the largest term, so \( \Theta(n) \).

16n minus a small number is correct then for this algorithm. The original presentation of the algorithm had improvements that dropped it to 5.5n and the best median-finding algorithm now does about 3n comparisons worst case.
LOWER BOUND FOR MEDIAN-FINDING

$E$ a set of $n$ distinct keys, $n$ odd. we want $(n + 1)/2$th key. Algorithm must know relation of all other key to median. It needs to establish relations as in Figure.
THE GRAPH

Has $n$ nodes, so $n - 1$ arcs, so $n - 1$ comparisons must be done. But can an adversary hurt us worse?

A Crucial Comparison for $x$ is a comparison involving key $x$ if it is the first comparison for $x > y$ for some $y \geq \text{median}$, or $x < y$ for some $y \leq \text{median}$. Comparison of $x$ and $y$ where $x > \text{median}$ and $y < \text{median}$ are noncrucial (tell us nothing).

The relation of $y$ to $\text{median}$ is not necessarily known at time of comparison with $x$. Crucial comps establish the relation of $x$ to $\text{median}$. 
The adversary wants us to make noncrucial comparisons. She chooses some value (not a particular key) to be median. She assigns a value to a key when the algorithm first uses that key in a comparison, and as long as possible she’ll assign values so as to put keys on opposite sides of median, so we learn nothing. She can’t assign values > median for more than \((n – 1)/2\) keys, ditto with smaller.
HER ADVERSARIAL STRATEGY

Let status of key be:
L: assigned a value > median.
S: assigned a value < median.
N: not yet been in comparison.

COMPARANDS ADVERSARY ACTION

(N, N) Make one key > median, other <.
(L, N), (N, L) Assign a value < median to N key.
(S, N), (N, S) Assign a value > median to N key

(L,L), (S,S) Correct response based on values.
(S,L), (L,S) Correct response based on values.

If there are already \((n - 1)/2\) keys with status S or L, she must ignore rules and put all new keys into L (or S). When only one N key remains, it gets the value median.

All the comparisons above the line table are noncrucial.
HER ADVERSARIAL STRATEGY CONT.

How many noncrucial comparisons does this strategy force?

Each creates at most one L-key, at most one S key. So adversary can continue until she fills up one side or other: until there are $(n - 1)/2$ L-keys or S-keys, so she can force $(n - 1)/2$ noncrucial comparisons.

Since the algorithm can start out with $(n - 1)/2$ (N,N) comparisons, she can’t guarantee any more than $(n - 1)/2$ noncrucial ones.

SO... number of comparisons is at least $n - 1$ (crucial) + $(n - 1)/2$ (noncrucial), or

$$C(n) = 3n/2 - 3/2.$$
Actually this adversary is not the worst. The lower bound has crept up to $1.75 - \log n$, to about $1.8n$, and the best lower bound now is slightly above $2n$. There’s a gap between the best known lower bound and the best algorithm.