Selection

The problem: Find the k-th smallest key in a set of keys.

1. Selection of the minimum.

\[
\text{function Min} \\
\theta(n) \left\{ \\
\text{min} := 1; \\
\text{for } i := 2 \text{ to } n \text{ do if } A[\text{min}] > A[i] \text{ then } \text{min} := i \text{ endif}
\text{endfor}
\]

2. Randomized Selection: Similar to Quicksort

\[
\text{function Random-Select}(p, r, i) \\
\text{if } p = r \text{ then return } p \\
\text{else } q := \text{Random-Partition}(p, r) \\
\text{k} := q - p + 1; \\
\text{if } i \leq k \text{ then Return Random-Select}(p, q, i) \\
\text{else Return Random-Select}(q+1, r, i-k) \text{ endif}
\]
Average Case Analysis:

Recall with prob. $\frac{2}{n}$ the array is split 1 and $(n-1)$, and for $2 \leq k \leq n-1$, with prob. $\frac{1}{n}$ it is split $k$ and $(n-k)$.

$$T(n) \leq \frac{1}{n} \left( T(\max\{1, n-1\}) + \sum_{k=1}^{n-1} T(\max\{k, n-k\}) + \Theta(n) \right)$$

$$\leq \frac{1}{n} \left( T(n-1) + 2 \sum_{\frac{n-1}{2}}^{n-1} T(k) \right) + \Theta(n)$$

$$= \frac{2}{n} \sum_{k=\left\lceil \frac{n}{2} \right\rceil}^{n-1} T(k) + O(n)$$

Assume inductively that

$$T(n) \leq Cn \text{ for a large enough } C.$$

$$T(n) \leq \frac{2C}{n} \left( \sum_{k=1}^{\left\lceil \frac{n}{2} \right\rceil-1} k - \sum_{k=1}^{\left\lceil \frac{n}{2} \right\rceil-1} k \right) + O(n)$$

$$\leq C(n-1) - \frac{C}{2} \left( \frac{n}{2} - 1 \right) + O(n)$$

$$\leq \frac{3C}{4} n + O(n) \leq Cn \text{ if } C \text{ is large enough.}$$

Think about which $C$ you should choose.
3. Deterministic Selection

Observe that the randomized one takes $\Omega(n^2)$ in the worst-case if splits are unbalanced.

It is possible to select in $O(n)$ time.

Algorithm

1. Divide the keys in groups of 5

   (e.g. for $k = \lceil n/5 \rceil$, $A[j], A[j+k], A[j+2k], A[j+3k], A[j+4k]$ is a group for $1 \leq j \leq k-1$.)

2. Find the median of each group.
   (use some simple sorting on each group)

3. Recurse on the $\lceil n/5 \rceil$ medians

4. Partition the array with the median-of-medians as pivot.

5. if $i \leq 9$ then find the key on the low part
   else find the $(i-9)$th key on higher part
4. Insertion Sort with Jumps

Procedure Insertion-Sort (p, k, n):

last := p+k;
while last ≤ n do
    pos := last;
        pos := pos-k
    endwhile
    last := last + k;
endwhile

We will use this sort in our Selection algorithm.
We implement the high-level algorithm for the Selection.

```
function Select(p, r, i)
    if r-p+1 \leq 50 then
        Insertion-Sort(p, r)
        return A[p+i-1]
    else
        k = ((r-p+1)+4) \div 5
        for j := 0 to k-1 do
            Insertion-Sort(p+j, k, r)
        endfor
        medmed := Select(p+2k, p+3k-1, k+2)
        q := Partition(p, r)
        if i \leq q-p+1 then
            return Select(p, q, i)
        else
            return Select(q+1, r, i-(q-p+1))
        endif
    endif
```
Analysis. For analysis we ignore the ceiling and floor functions.

- The number of medians less than or greater than the \( \overline{\text{med}} \) is \( \frac{n}{10} \).
- The number of elements less than or greater than these medians is \( \frac{2n}{10} \).

\[ \leq \text{med} < \overline{\text{med}} \]
\[ \overline{\text{med}} < \text{med} \leq \]

- We throw away (prune) at least \( \frac{3n}{10} \) elements.
Recurrence relation for time Complexity

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

Prove \( T(n) = O(n) \) by assuming

\[ T(n) \leq cn. \quad \text{Inductive hypothesis} \]

\[ T(n) \leq \frac{cn}{5} + \frac{cn}{10} + O(n) \]

\[ \leq c \cdot \frac{9n}{10} + O(n) \]

\[ \leq c \cdot \frac{9}{10} n + c'n \]

\[ \leq cn \left( \frac{9}{10} + \frac{c'}{c} \right) \]

Choose \( C \) such that \( \frac{c'}{c} + \frac{9}{10} \leq 1 \)

\[ \text{choice of } c \text{ is up to us.} \quad \text{or, } C \geq 10c'. \]