Selection

k'th Order Statistic

L = List of n numbers (possibly with duplicates.)

$x \in L$ is the k'th *order statistic* of S if x is the k'th smallest element of L.
(x is the k'th element of L when L is sorted in increasing order.)

What is the first order statistic of L?

What is the n'th order statistic of L?

k'th Order Statistic: Median

L = List of n numbers (possibly with duplicates.)

$x \in L$ is the k'th *order statistic* of S if x is the k'th smallest element of L.
(x is the k'th element of L when L is sorted in increasing order.)

$x \in L$ is the (lower) median of L if x is the $\lfloor (n+1)/2 \rfloor$'th smallest element of L.

Algorithm `SortAndSelect`

Input : Array A.
Integer k.

Result : k'th order statistic of A.

```
SortAndSelect(A[ ],k)
Sort A[ ];
return (A[k]);
```

Randomized Selection: Example

Find the 7'th order statistic of the following 13 elements:

$$4, 17, 15, 32, 7, 19, 25, 9, 20, 14, 36, 12, 2$$

Note: Since $\lfloor (13+1)/2 \rfloor = 7$, the 7'th order statistic is the median.

Selection: Randomized Algorithm

Input : Array A of at least j elements.
Integers i, j and k.

Output : k'th order statistic of A.

```
RandomizedSelect(A[ ],i,j,k)
p ← RandomElement(A,i,j);
s ← Partition(A, i, j, p);
/* A[s] = p and A[i'] ≤ p ≤ A[j'] for i ≤ i' ≤ s ≤ j' ≤ j          */
m ← s − i + 1;                 /* A[s] is the m'th element of A[i..j] */
if (m = k) then  x ← A[s] ;
else if (m > k) then  x ← RandomizedSelect(A, i, s − 1, k) ;
else /* m < k */
|  x ← RandomizedSelect(A, s + 1, j, k − m);
end
return (x);
```

`RandomizedSelect`: Analysis

$ET(n)$ = expected running time of `RandomizedSelect` on n values.

Assume array has no duplicates and all permutations are equally likely.

$$\begin{aligned} ET(n) &= \sum_{q=1}^{n} Pr(m = q) ET(m = q) \\ &= \sum_{q=1}^{n} \frac{1}{n} ET(m = q). \end{aligned}$$

$$ET(m = q) \quad \leq \quad \max(ET(m = q \text{ and } k < m), ET(m = q \text{ and } k > m)).$$

Therefore,

$$ET(n) = \frac{1}{n} \sum_{i=1}^{n} ET(m = q) \leq \frac{1}{n} \sum_{i=1}^{n} \max(ET(q-1), ET(n-q)).$$

`RandomizedSelect`: Upper Bounds

X_n = running time of **`RandomizedSelect`** on array of size n.
$ET(X_n) = E(X_n)$ = expected running time on array of size n.
Assume array has no duplicates.
$m = s - i + 1$. (After partition, p is m'th element of $A[i], \ldots, A[j]$.)

$$\begin{aligned} ET(n) = {} & Pr(m < n/4) ET(X_n | m < n/4) + \\ & Pr(n/4 \leq m \leq 3n/4) ET(X_n | n/4 \leq m \leq 3n/4) + \\ & Pr(m > 3n/4) ET(X_n | m > 3n/4). \end{aligned}$$

$$\begin{aligned} Pr(m < n/4) &= 1/4. \\ Pr(m > 3n/4) &= 1/4. \\ Pr(n/4 \leq m \leq 3n/4) &= 1/2. \\ ET(X_n | m < n/4) &\leq ET(X_n | m = 1) = cn + ET(n-1). \\ ET(X_n | m > 3n/4) &\leq ET(X_n | m = n) = cn + ET(n-1). \\ ET(X_n | n/4 \leq m \leq 3n/4) &\leq ET(X_n | m = n/4) \\ &\leq cn + \max(ET(n/4), ET(3n/4)) = cn + ET(3n/4). \end{aligned}$$

`RandomizedSelect`: Upper Bounds

$m = s - i + 1$. (After partition, p is m'th element of $A[i], \ldots, A[j]$.)

$$\begin{aligned}
Pr(m < n/4) &= 1/4. \\
Pr(m > 3n/4) &= 1/4. \\
Pr(n/4 \leq m \leq 3n/4) &= 1/2. \\
ET(X_n | m < n/4) &\leq ET(X_n | m = 1) = cn + ET(n-1). \\
ET(X_n | m > 3n/4) &\leq ET(X_n | m = n) = cn + ET(n-1). \\
ET(X_n | n/4 \leq m \leq 3n/4) &\leq ET(X_n | m = n/4) \leq cn + ET(3n/4).
\end{aligned}$$

$$\begin{aligned}
ET(n) = {} & Pr(m < n/4) ET(X_n | m < n/4) + \\
& Pr(n/4 \leq m \leq 3n/4) ET(X_n | n/4 \leq m \leq 3n/4) + \\
& Pr(m > 3n/4) ET(X_n | m > 3n/4) \\
\leq {} & \frac{1}{4}\Big(cn + ET(n-1)\Big) + \frac{1}{2}\Big(cn + ET(3n/4)\Big) + \\
& \frac{1}{4}\Big(cn + ET(n-1)\Big).
\end{aligned}$$

`RandomizedSelect`: Upper Bounds

$$\begin{aligned}
ET(n) &\leq \frac{1}{4}\Big(cn + ET(n-1)\Big) + \frac{1}{2}\Big(cn + ET(3n/4)\Big) + \\
&\quad \frac{1}{4}\Big(cn + ET(n-1)\Big) \\
&= cn + \frac{1}{2}ET(n-1) + \frac{1}{2}ET(3n/4) \\
&\leq cn + \frac{1}{2}ET(n) + \frac{1}{2}ET(3n/4). \\
ET(n) - \frac{1}{2}ET(n) &\leq cn + \frac{1}{2}ET(3n/4). \\
\frac{1}{2}ET(n) &\leq cn + \frac{1}{2}ET(3n/4). \\
ET(n) &\leq 2cn + ET(3n/4) \\
&\leq c_2 n + ET(3n/4) \qquad \text{where } c_2 = 2c.
\end{aligned}$$

$$\therefore ET(n) \in O(n).$$

`RandomizedSelect`: Lower Bounds

`RandomizedSelect` always executes `Partition` at least once.
`Partition` takes cn time. Therefore, $ET(n) \in \Omega(n)$.

Selection: Deterministic Algorithm

Input : Array A of at least j elements.
Integers i, j and k.
Output : k'th order statistic of A.

`Select`(A[ ],i,j,k)
$p \leftarrow$ `ApproxMedian`(A,i,j);
$s \leftarrow$ `Partition`(A, i, j, p);
/* *$A[s] = p$ and $\mathsf{A}[i'] \leq p \leq A[j']$ for $i \leq i' \leq s \leq j' \leq j$* */
$m \leftarrow s - i + 1$; /* *$A[s]$ is the m'th element of $\mathsf{A}[i..j]$* */
if $(m = k)$ **then** $x \leftarrow A[s]$;
else if $(m > k)$ **then** $x \leftarrow$ `Select`(A, i, $s-1$, k) ;
else /* $m < k$ */
  $x \leftarrow$ `Select`(A, $s+1$, j, $k-m$);
end
return (x);

ApproximateMedian

Input : Array A of at least j elements.
Integers i, j and k.

Output : Approximate median of $\mathsf{A}[i], \ldots, \mathsf{A}[j]$.

```
ApproxMedian(A[ ],i,j)
n ← j − i + 1;
Partition A[i],...,A[j] into ⌈n/5⌉ groups of 5;
for i ← 1 to ⌈n/5⌉ do
|   B[i] ← median of i'th group;
end
p ← Select(B,1,⌈n/5⌉,⌊n/10⌋);
return (p);
```