A SIMPLER WAY IS TO CHOOSE A RANDOM ELEMENT IN A[p...r] AS PIVOT, SWAP THAT WITH A[r], THEN PARTITION AS USUAL

RandPartition(A, p, r)

1.) i ← Rand(p, r)
2.) A[i] ↔ A[r]
3.) return Partition(A, p, r)

RandQuicksort(A, p, r)

1.) if p < r
2.) q ← RandPartition(A, p, r)
3.) RandQuicksort(A, p, q-1)
4.) RandQuicksort(A, q+1, r)

THIS IS CONSIDERED BY SOME TO BE THE ALGORITHM OF CHOICE FOR SORTING LARGE INPUT.

Ex

MaxMin(A,p,r) FINDS THE MAXIMUM AND MINIMUM ELEMENTS IN THE SUBARRAY A[p..r].

Min(m_1, m_2)

1.) if $m_1 < m_2$
2.) return m_1
3.) return m_2

max(M_1, M_2) is similar. EACH DOES ONE COMPARISON.

MaxMin(A,p,r) (Pre: $p \leq r$)

1.) if $p = r$
2.) return (A[p], A[p])
3.) $q \leftarrow \lfloor \frac{p+r}{2} \rfloor$
4.) $(m_1, M_1) \leftarrow$ MaxMin(A, q+1, r)
5.) $(m_2, M_2) \leftarrow$ MaxMin(A, p, q)
6.) return (min(m_1, m_2), max(M_1, M_2))

LET T(n) DENOTE THE NUMBER OF COMPARISONS PERFORMED BY MaxMin(A,p,r) ON ARRAYS OF LENGTH n.

THEN

$$T(n) = \begin{cases} 0 & n = 1 \\ T(\lfloor \frac{n}{2} \rfloor) + T(\lceil \frac{n}{2} \rceil) + 2 & n \geq 2 \end{cases}$$

EXERCISE:
SHOW THAT THE EXACT SOLUTION IS $T(n) = 2n - 2$.

THIS IS NO BETTER THAN THE OBVIOUS ITERATIVE ALGORITHM.

EXERCISE:
DESIGN A DIVIDE AND CONQUER ALGORITHM WHICH FINDS MAXIMUM AND MINIMUM IN EXACTLY $\lceil \frac{3n}{2} \rceil - 2$ COMPARISONS. (HINT: SECTION 9.1 DESCRIBES AN ITERATIVE ALGORITHM TO DO THIS.)

EX. THE SELECTION PROBLEM

THE i^{TH} ORDER STATISTIC OF AN ARRAY $A[1 \cdots n]$ CONSISTING OF n DISTINCT ELEMENTS IS THE i^{TH} SMALLEST ELEMENT. EQUIVALENTLY THE i^{TH} ORDER STATISTIC IS THE UNIQUE ELEMENT IN A WHICH IS GREATER THAN EXACTLY $i-1$ OTHER ELEMENTS (WHERE $1 \leq i \leq n$).

e.g. $i = 1$ GIVES THE MINIMUM
$i = n$ GIVES THE MAXIMUM

THE i^{TH} ORDER STATISTIC IS GREATER THAN OR EQUAL TO EXACTLY i ELEMENTS OF A.

PROBLEM: GIVEN $A[1 \cdots n]$, WHERE ALL ELEMENTS ARE DISTINCT, DETERMINE THE i^{TH} ORDER STATISTIC.

ONE APPROACH WOULD BE TO SORT $A[1 \cdots n]$ THEN JUST RETURN $A[i]$. IN GENERAL THIS TAKES TIME $\Omega(n \lg n)$.

RandSelect IS A RANDOMIZED ALGORITHM WHICH FINDS THE i^{TH} ORDER STATISTIC IN LINEAR TIME, ON AVERAGE.

RECALL THAT RandPartition(A, p, r) SPLITS THE SUBARRAY A[p..r] INTO TWO SUBARRAYS SATISFYING

$$A[p..(q-1)] \leq A[q] \leq A[(q+1)..r]$$

RandSelect(A, p, r, i) (Pre: $1 \leq i \leq r-p+1$)

```
1.) if p = r
2.)     return A[p]
3.) q ← RandPartition(A, p, r)
4.) k ← q - p + 1        // k is length of A[p..q]
5.) if k = i
6.)     return A[q]
7.) else if i < k
8.)     return RandSelect(A, p, q-1, i)
9.) else
10.)    return RandSelect(A, q+1, r, i-k)
```

RandSelect is similar in some respects to both Quicksort and BinarySearch. Like Quicksort it randomly splits the subarray A[p..r] in order to exploit a good average case running time. Like BinarySearch it recurs on only one subarray. Unlike BinarySearch we seek not an index, but an array element.

LET $t(n)$ DENOTE THE AVERAGE NUMBER OF (ARRAY) COMPARISONS BY RandSelect$(A, 1, n, i)$

ASSUME THAT EACH PERMUTATION OF $A[1 \ldots n]$ IS EQUALLY LIKELY, HENCE THE RETURN VALUE OF RandPartition IS EQUALLY LIKELY TO BE ANY OF THE NUMBERS $1 \le q \le n$.

A PRIORI $t(n)$ DEPENDS ON i. WE'LL SEE THAT IN FACT IT DOESN'T. RECALL THAT RandPartition DOES $(n-1)$ COMPARISONS. THUS

$$t(n) = \frac{\sum_{q=1}^{n} \left((n-1) + P(i<q)\, t(q-1) + P(i>q)\, t(n-q) \right)}{n}$$

WHERE

$$P(i<q) = \frac{n-i}{n} \quad \text{AND} \quad P(i>q) = \frac{i-1}{n}$$

ARE THE PROBABILITIES THAT q IS IN THE RANGE $i < q \le n$ AND $1 \le q < i$ RESPECTIVELY. THUS

$$t(n) = (n-1) + \frac{1}{n} \sum_{q=1}^{n} \left(\left(\frac{n-i}{n}\right) t(q-1) + \left(\frac{i-1}{n}\right) t(n-q) \right)$$

$$= (n-1) + \frac{1}{n^2} \left[(n-i) \sum_{q=1}^{n-1} t(q) + (i-1) \sum_{q=1}^{n-1} t(q) \right]$$

$$= (n-1) + \frac{1}{n^2}(n - i + i - 1) \sum_{q=1}^{n-1} t(q)$$

$$\therefore \boxed{t(n) = (n-1) + \left(\frac{n-1}{n^2}\right) \sum_{q=1}^{n-1} t(q)}$$

THEREFORE $t(n)$ DOES NOT DEPEND ON i, AS CLAIMED EARLIER.

OBSERVE THAT THIS RECURRENCE IS VERY SIMILAR TO THE ONE FOR THE AVERAGE RUN TIME OF Quicksort.

EXERCISE

SHOW THAT $t(n) = \Theta(n)$.

OBVIOUSLY $t(n) \geq n-1 = \Omega(n)$. PROVE THAT $t(n) = O(n)$ BY INDUCTION ON n. i.e. SHOW

$$\forall n \geq 1 \; : \; t(n) \leq 2n$$

INDUCTION HYPOTHESIS : $\forall \, q \leq n-1 \; : \; t(q) \leq 2q$

$$\therefore t(n) \leq (n-1) + \left(\frac{n-1}{n^2}\right) \sum_{q=1}^{n-1} 2q$$

$$= (n-1) + \left(\frac{n-1}{n^2}\right) \cdot 2 \, \frac{n(n-1)}{2} \leq 2n$$

↑ PROVE.

EXERCISE

FIND THE EXACT SOLUTIONS TO THIS RECURRENCE.

ANSWER:

$$t(n) = (n-1)\left\{1 + \sum_{r=1}^{n-1} \frac{\mu_r \cdot (r-1)}{\mu_n \cdot n(n^2+n-1)}\right\}$$

WHERE

$$\mu_n = \prod_{k=2}^{n} \left(\frac{k}{k^2+k+1}\right)$$