① Randomized Partition(A, p, r): A is array, r & p are indexes.

```
i = Random(p, r)
exchange A[i] with A[r]
return Partition(A, p, r)
```

```
Partition(A, p, r):
x = A[r]
i = p - 1
for j = p to r - 1
    if A[j] <= x
        i = i + 1
        exchange A[i] with A[j]
exchange A[i+1] with A[r]
return i + 1
```

```
Randomized-Quicksort(A, p, r):
if p < r
    q = Randomized-Partition(A, p, r)
    Randomized-Quicksort(A, p, q-1)
    Randomized-Quicksort(A, q+1, r)
```

Each element has $\frac{1}{n}$ chance of being selected as pivot. $T(n) = T(i-1) + T(n-i) + O(n)$; partition time

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

$T_a(n) - T_a(n-1) = c + T(n-1) \rightarrow T_a(n) = c + 2T(n-1)$

$nT_a(n) = cn^2 + 2\sum_{i=0}^{n-1} T_a(i)$ $\quad (n-1)T_a(n-1) = c(n-1)^2 + 2\sum_{i=0}^{n-2} T_a(i)$

$nT_a(n) - (n-1)T_a(n-1) = cn^2 - (n-1)^2 + 2T_a(n-1) \rightarrow T'(n)$

$nT_a(n) = (n+1)T_a(n-1) + 2cn - c$ $\quad \frac{T_a(n)}{n+1} = \frac{T_a(n-1)}{n} + \frac{2cn - c}{n(n+1)}$

$T'(n) = T'(n-1) + \frac{c(2n-1)}{n(n+1)} \Rightarrow T'(n) = T'(n-1) + \frac{k}{n}$

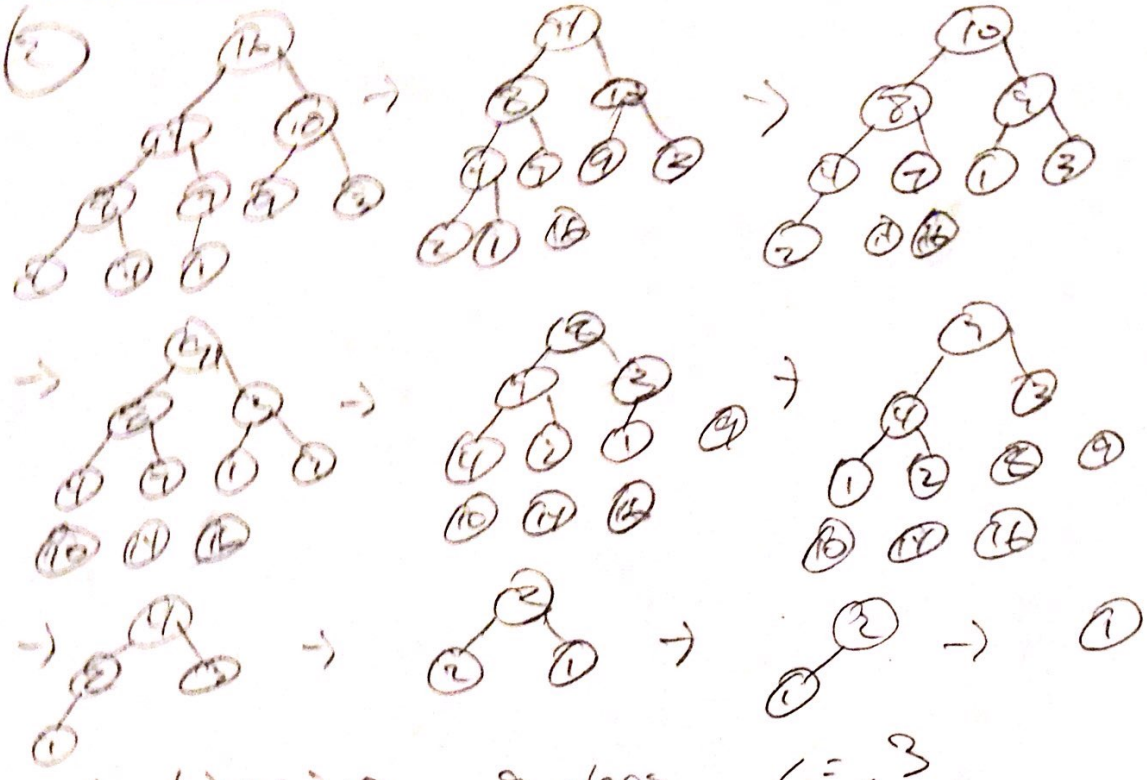
$= T'(n-2) + \frac{k}{n-1} + \frac{k}{n}$

$k\left(\frac{1}{n} + \frac{1}{n-1} + \dots + 1\right) = \ln(n)$ $\quad T'(n) = O(\ln(n))$

$T_a(n) = O((n+1)\ln(n))$. (choosing the $\frac{n}{4}$ element from the sorted list takes constant time.

Scanned by CamScanner

(2)



$n\log_2(n) \geq 30$ 9 steps $c = .3$

(3)
```
for(i:0; i<k; i++) // for each list
    add_to_minheap[i](k) // heap for each list
for(i:0; i<k; i++)
  for(j:0; j<heap[i].size; j++)
      while(heap[i] is not empty)
          add_to_single_list(heap[i][j])
```
The top for loop takes $k\log n$ because inserting to a heap is $O(\log n)$ and the loop runs k times.

(4) Complexity = (max length of numbers) · (work to bucket)
$O(n)$
The run time of radix sort depends on the intermediate sorting algorithm. If counting sort is used which is $O(n+k)$ where k is the maximum number of digits for a number in the list then each pass over n d-digit numbers takes time $O(n+k)$, the total time for radix sort is $O(d(n+k))$

(5) Every comparison based sorting algorithm is $\Omega(n\log(n))$. There are $\log n$ sequences to sort so the lower bound of this method is $\Omega(n\log^2(n))$

Scanned by CamScanner