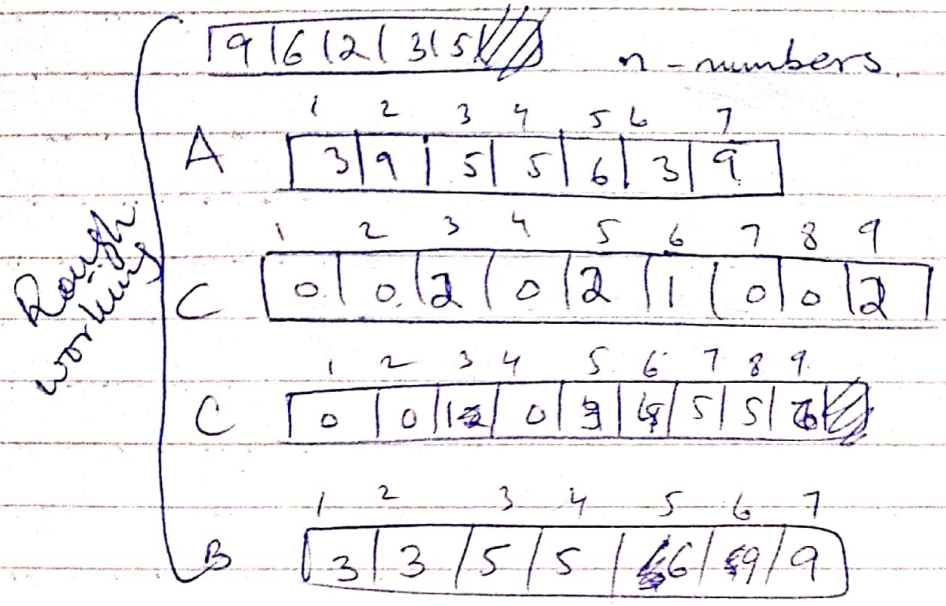
Momina Atif Dar
P18-0030

Rough working



Counting Sort is optimal than Quick Sort as in Counting Sort simple iterations are involved.

for i = 1 to k
C[i] = 0
] k times

~~for j = 1 to len(A)~~

for j = 1 to len(A)
do put count of numbers in C
] n-elements time

for f = 2 to k
do accumulate C
] k-1 times

for m = len(A) to 1 (reverse)
do put values in B
] n times

$= k + n + (k-1) + n$

$= k + n + k - 1 + n$

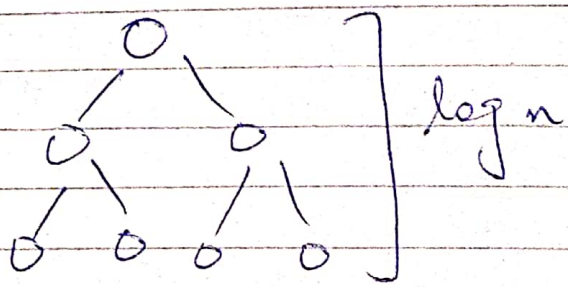
$= 2k + 2n - 1$

$= O(k+n)$

if $k = n$ then $O(n)$.

Whereas in Quick Sort we do partition and BST is formed at backend in best case.

Best case in Quick Sort:



Partition algo takes $O(n)$ running time

So total running time = $O(n \log n)$

In worst case: (edge case) where pivot is at index 0 or index n-1

$n + (n-1) + (n-2) \ldots + 1$

$= \frac{n(n-1)}{2}$

$= O(n^2)$

So Counting Sort is better as its running time is more efficient and less than Quick Sort.