Design And Analysis Of Algorithms

Date: 9-9-22

K20-1052
BSE-5B

Q.1 Insertion sort

6, 16, 12, 27, 9, 1, 18, 5, 31

6, 16, 12, 27, 9, 1, 18, 5, 31	K = 16
16, 6, 12, 27, 9, 1, 18, 5, 31	K = 12
16, 12, 6, 27, 9, 1, 18, 5, 31	K = 27
27, 16, 12, 6, 9, 1, 18, 5, 31	K = 9
27, 16, 12, 9, 6, 1, 18, 5, 31	K = 1
27, 16, 12, 9, 6, 1, 18, 5, 31	K = 18
27, 18, 16, 12, 9, 6, 1, 5, 31	K = 5
27, 18, 16, 12, 9, 6, 5, 1, 31	K = 31
31, 27, 18, 16, 12, 9, 6, 5, 1,	none

Time complexity = $O(n^2)$
for the worst case suppose all unsorted elements, where comparison and swapping increases in each iteration as 1, 2, 3 .. n
so it becomes a series of $n(n-1)/2 = O(n^2)$

Loop Invariant:
At the start of each ~~loop~~ iteration of for loop, the sub array A[1 .. j-1] consists of elements originally in A=[1 .. j-1] but in sorted order.

Initialization:
At the start of for loop j=2, the array has a first sorted element.

ALBA

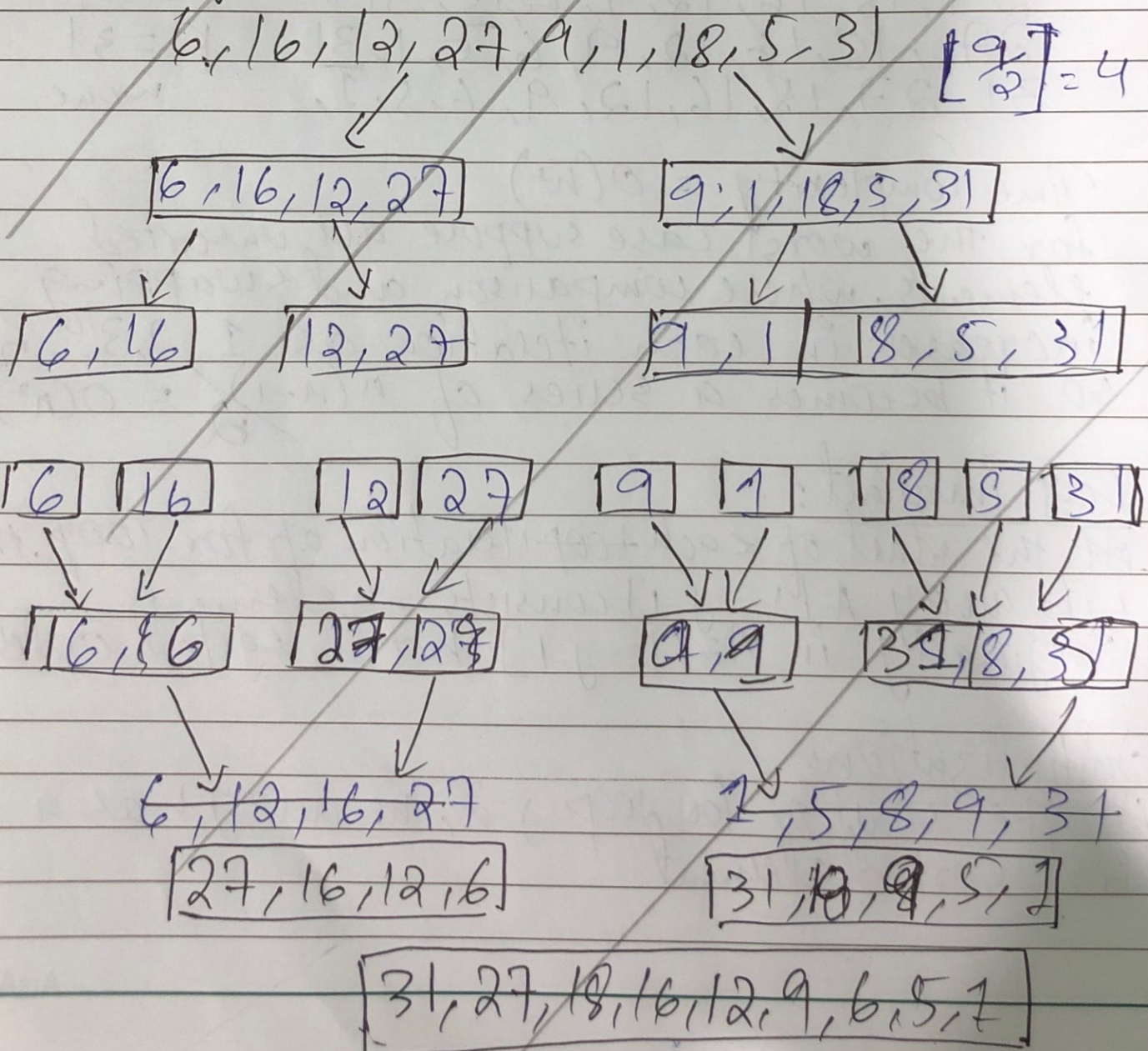
K20-1052

Date:

Maintenance:
At the start of each iteration, suppose $j = 4$ the element of the inner loop will shift forward towards ~~wait~~ sorted position. Therefore A[1---3] is sorted in this case.

Termination:
At the end of loop, we have array A[1--n] that is sorted.

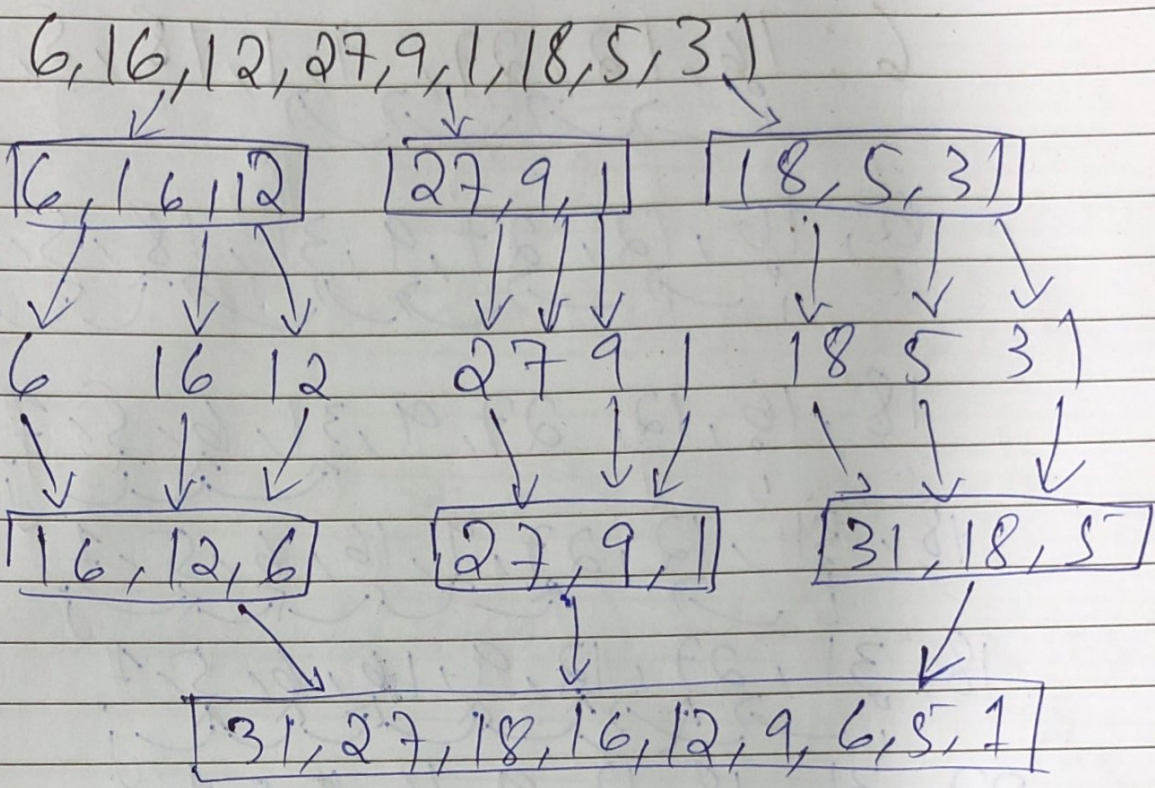
Q.2 Merge sort



K20-1052.

Date:

Q. MergeSort



Time complexity:

$$T(N) = aT(n/b) + f(n)$$

$a = 3, b = 3 \quad f(n) = O(n)$

$$T(n) = 3T(n/3) + O(n) \quad \therefore a = b^d$$

$$O(n^d \log n)$$

$$O(n \log_3 n)$$

ALBA

K20-1052

Date:

Q.3 QuickSort.

6, 16, 12, 27, 9, 1, 18, 5, 31 — p → 6

6, 16, 12, 27, 9, 31, 18, 5, 1 → 6

18, 16, 12, 27, 9, 31, 6, 5, 1 → 18

18, 31, 12, 27, 9, 16, 6, 5, 1 → 18

18, 31, 27, 12, 9, 16, 6, 5, 1 → 18

27, 31, 18, 12, 9, 16, 6, 5, 1 → 27

31, 27, 18, 12, 9, 16, 6, 5, 1 → 12

31, 27, 18, 16, 12, 9, 6, 5, 1 → sorted.

Before first iteration $i = p+1$ and $j = n-1$
No value lie between p and i

As $arr[i] \geq p$, increments i and
$arr[j] \leq p$ decrements j

As j crosses i, swap arr[j] with p. At termination, pivot has elements greater on one side and lesser on other side.

K20-1052.

Date:

Q.4 Considering an input array having n number of cards.

Then use of divide and conquer strategy by dividing array into n/2. Splitting can be done recursively, base case is n=1

A relation such that $T(n) = 2T(n/2) + O(n)$

where $T(n) = O(n \log n)$ using master/iterative method.

For linear O(n) case.
we can use Boyer-Moore Algorithm where we can have count and max_index variable.
While traversing array. check if

Array[max_index] == Array[i].
count++

else count--
if count == 0, Max_index = i, count = 1

Since linear traversal of array and finding element so

$= O(n) + O(n) = O(n)$

ALBA

K20-1052.

Date:

Q.5 In order to minimize the sum, pair up the numbers in a way that largest number would be added to the smallest element of array. Thus we can use Merge sort to sort elements, then selecting an element from odd index (1, 3, 5, 7, n-1) and other from even index (2, 4, 6, 8 ... n) of sorted array.

Q.6 $n^3 - 2n + 1 = O(n^3)$

add constants $\Rightarrow$ $1 + 1 = 2$

$n^3 - 2n + 1 \leq c(n^3)$

$c = 2$

$n^3 - 2n + 1 \leq 2n^3$

when $n_0 = 1 \Rightarrow 0 \leq 2$, Hence $n_0 = 1$

$5n^2 \log_2 n + 2n^2 = O(n^2 \log_2 n)$.

$5n^2 \log_2 n + 2n^2 \leq c(n^2 \log_2 n)$

$5n^2 \log_2 n + 2n^2 \leq 7n^2 \log_2 n$ $\quad c = 5 + 2 = 7$

when $n_0 = 2$ we have

$28 \leq 28$ so $n_0 = 2$

K20-1052

Date:

Q.7 Mathematical way of representing time complexity is Asymptotic notation. also refers to study of function f as n reaches infinity.

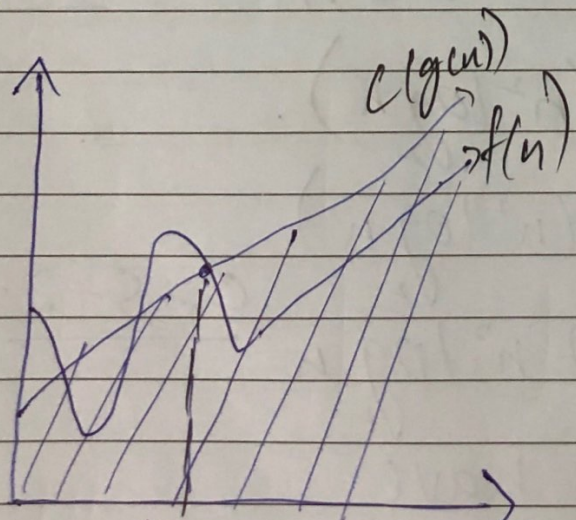
Big O (upper bound)

$f(n) = O(g(n))$

$f(n) \leq c(g(n))$

if $0 \leq f(n) \leq cg(n)$ where $n \geq n_0$

we say $f(n)$ is big-O of $g(n)$



$f(n) = O(g(n))$

ex $n^2 + n = O(n^3)$

$c = 1$ at $n \geq 2$

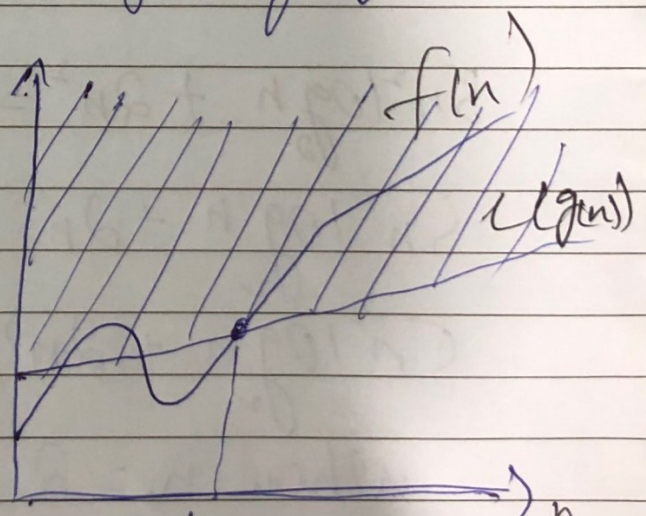
Big Ω (lower bound)

$f(n) = \Omega(g(n))$

$f(n) \geq c(g(n))$.

if $0 \leq c(g(n)) \leq f(n)$ where $n_0 \geq n_0$

we say $f(n)$ is Omega of $g(n)$



$f(n) = \Omega(g(n))$

ex $n^3 + 4n^2 = \Omega(n^3)$

$c = 1$ at $n \gg 1$

ALBA

K20-1052.

Date:

Big-Θ (tight bound)

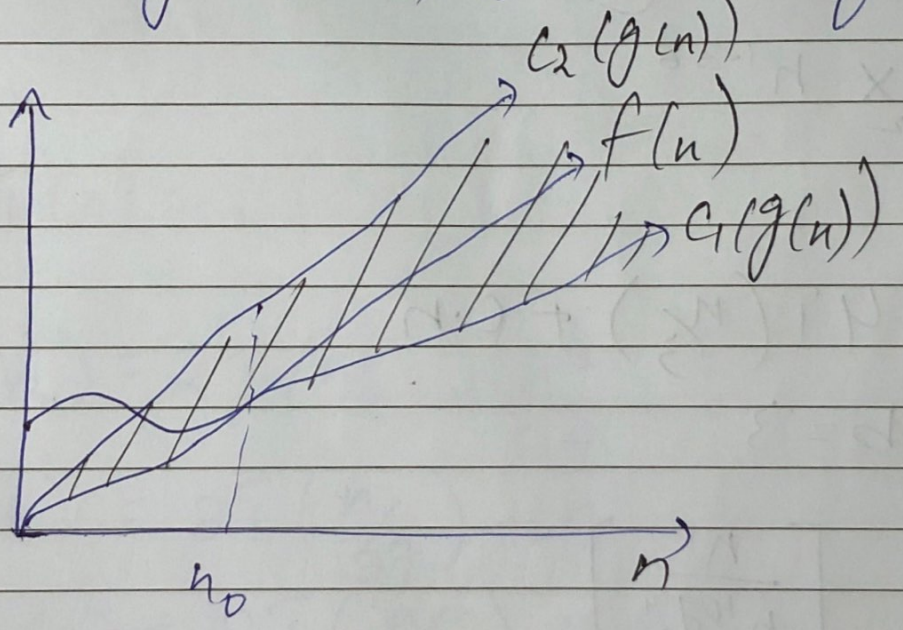
$f(n) = \Theta(g(n))$

$f(n) = c(g(n))$

if $0 \leq c_1(g(n)) \leq f(n) \leq c_2(g(n))$

where $n \geq n_0$

we say $f(n)$ is theta of $g(n)$



$f(n) = \Theta g(n)$

ex $n^2 + 5n = \Theta(n^2)$ $\quad n^2 + 5n \leq c(n^2)$

$c_1 = \cancel{1}$ $\quad c_2 = 6$

at $n \geq 1$

K20-1052

Date:

Q.8 Solve using Master Theorem

$T(n) = 2T(n/3) + c \cdot n^2$ → $f(n)$

$a = 2 \quad b = 3$

$n^{\log_3^2} \times \left[\frac{n^2}{n^{\log_3^2}}\right]$

$n^{0.63} \times [n^{1.36}]$ → $n^r : r > 0 \Rightarrow O(n)^r$

$n^{0.63} \times n^{1.36}$

$\Rightarrow n^2$

$T(n) = 4T(n/3) + c \cdot n$

$a = 4 \quad b = 3$

$n^{\log_3^4} \times \left[\frac{n^1}{n^{\log_3^4}}\right]$

$n^{1.26} \times \left[\frac{n^1}{n^{1.26}}\right]$ → $-ve\ n^r$ so $O(1)$

$n^{1.26}$

K20-1052.

Date:

$T(n) = 8T(n/2) + C \cdot n^3$

$a = 8 \quad b = 2$

$n^{\log_2 8} \times \left[\frac{n^3}{n^{\log_2 8}}\right]$

$n^3 \times \left[\frac{n^3}{n^3}\right] \rightarrow n^0$

$n^3 \times \log n$

Iteration Method

Q.9 $T(n) = 2T(n/3) + n^2 \quad , T(1) = 1$ → (1)

$T(n/3) = 2T(n/9) + \frac{n^2}{27}$

$T(n/9) = 2T(n/27) + \frac{n^2}{81}$

$T(n/27) = 2T(n/81) + \frac{n^2}{729}$

$3^k = n$

→ $2\left(2T(n/9) + \frac{n^2}{27}\right) + n^2$

→ $4T(n/9) + \frac{2n^2}{27} + n^2$ → (2)

→ $4\left(2T(n/27) + \frac{n^2}{81}\right) + \frac{2n^2}{27} + n^2$

$= 8T(n/27) + n^2 + \frac{2n^2}{27} + \frac{4n^2}{81}$ → (3)

$= 2^k T(n/3^k) + n^2$ → n^2 (T.P)

ALBA

K20-1052

Date:

$2^k T(\frac{n}{n}) + n^2$

$2^k (1) + n^2$

$2^{\log_3 n} + n^2$

$= O(n^2)$

$3^k = n$

$\log_3 3^k = \log_3 n$

$k = \log_3 n$

$T(n) = 4T(n/3) + n \quad (T(1) = 1)$

$T(n/3) = 4T(n/9) + n/3$

$T(n/9) = 4T(n/27) + n/9$

$T(n/27) = 4T(n/81) + n/27$

$= 4(4T(n/9) + n/3)$

$= 16T(n/9) + 4n/3 \Rightarrow (2)$

$= 16(4T(n/27) + n/9) + 4n/3$

$= 64T(n/27) + 4n/3 + 16n/9 \Rightarrow (3)$

$T(n/3^k) + n(4/3 + 16/9 + 64/27)$

$= T(n/n) + O(n) \times \text{constant}$

$= n^{\log_3 4} + O(n) \Rightarrow n^{1.26}$

$3^k = n$

$k = \log_3 n$

$4^{\log_3 n}$

$n^{\log_3 4}$

K20-1052

Date:

Q.10 if $f(n) = n$, $O(g(n))$ will be

1-

$f(n) \neq \leq c \cdot (g(n))$

$\Omega(h(n)) \Rightarrow \log n$

$g(n) + h(n) \leq f(n)$

$n + \log n \neq n$ so false

2-

suppose $f(n) = n$ $g(n) = 2n$.

$f(n) = O(g(n))$ $f(n) = \Omega(g(n))$

$f(n) \leq c.n$ $f(n) \geq c.n$

$n \leq c.n$ ✓ $2n \geq n$ ✓

so true for $(f(n))^3 = \Theta((g(n))^3$

3- $f(n) = n$ $O(g(n)) = \Omega \log n$.

$(f(n))^2 = (g(n))^2$

$(n^2) \neq (\log n)^2$ false.