DESIGN AND ANALYSIS
OF ALGORITHMS.

DATE DAY M T W T F S S

MUHAMMAD BASIL ALI KHAN
20K-0477

Question #01

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

Step #01:
6, 16, 12, 27, 9, 1, 18, 5, 31
↑ Key

Step #02:
16, 6, 12, 27, 9, 1, 18, 5, 31
↑ Key.

Step #03:
16, 12, 6, 27, 9, 1, 18, 5, 31
↑ Key.

Step #04:
a: 16, 12, 27, 6, 9, 1, 18, 5, 31
↑ Key.

b): 16, 27, 12, 6, 9, 1, 18, 5, 31
↑ Key.

c: 27, 16, 12, 6, 9, 1, 18, 5, 31
↑ Key

Step #05:
27, 16, 12, 6, 9, 1, 18, 5, 31
↑ Key

Step #06:
27, 16, 12, 9, 6, 1, 18, 5, 31
↑ Key

Step #07:
a: 27, 16, 12, 9, 6, 1, 18, 5, 31
↑ Key

b

~~Step #08~~ b: 27, 16, 12, 9, 6, 18, 1, 5, 31
↑ Key.

DATE DAY M T W T F S S

~~Step #08~~

c : 27, 16, 12, 9, 18, 6, 1, 5, 31
↑ Key

d: 27, 16, 12, 18, 9, 6, 1, 5, 31
↑ Key

e: 27, 16, 18, 12, 9, 6, 1, 5, 31
↑ Key

f: 27, 18, 16, 12, 9, 6, 1, 5, 31
↑ Key

Step # 08 :
27, 18, 16, 12, 9, 6, 1, 5, 31
↑ Key

Step # 09 :
27, 18, 16, 12, 9, 6, 5, 1, 31
↑ Key

RESULTANT LIST OF INTEGERS :
31, 27, 18, 16, 12, 9, 6, 5, 1.

TIME COMPLEXITY :

Time complexity of insertion sort is $O(n^2)$ because there are two loops one runs to traverse the whole array and 2nd loop does the comparison if any ~~th~~ for about n-times depending on input so timep complexity is $O(n^2)$

LOOP INVARIANT :

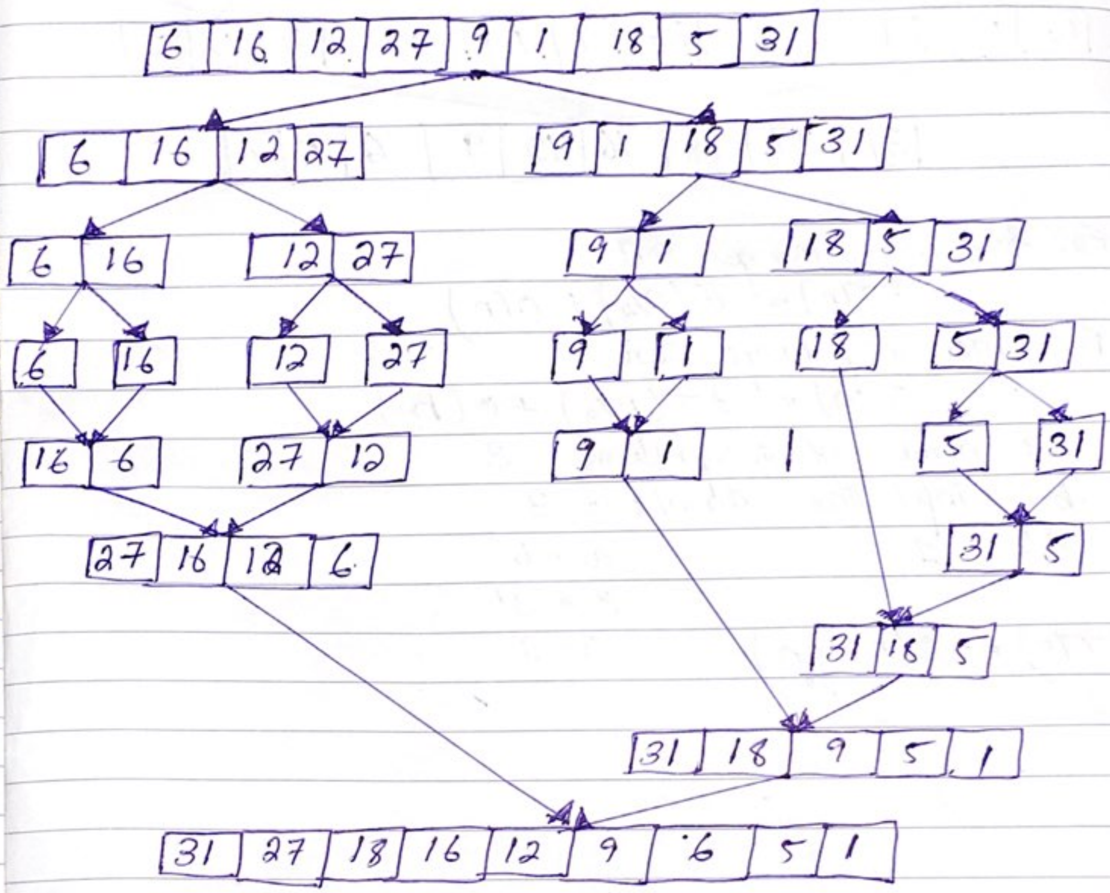
In insertion sort the subarray that is A [1, ... j-1] the array is alway ~~the~~ sorted. To prove we have took three properties

1- ~~Before~~ During intialization the initial will alway be true
2- Before nth iteration the subarray is sorted and after (n+1)th iteration the subarray sorted

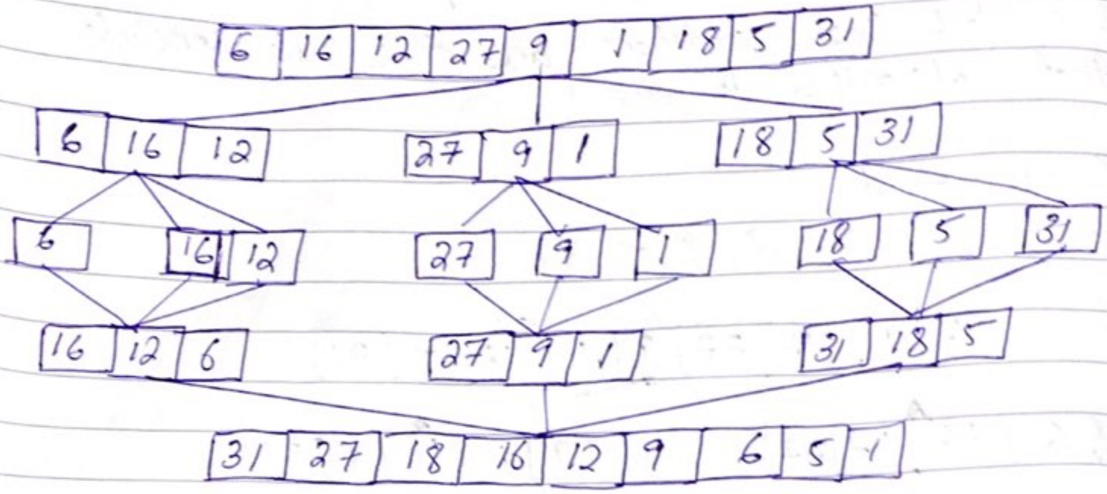
DATE ______ DAY M T W T F S S

3- And After the termination of loop the subarray is always sorted.
Since all these conditions are true we conclude that algorithm is correct.

Question #02:
6, 16, 12, 27, 9, 1, 18, 5, 31



THREE WAY MERGE SORT:



For two way merge sort:

$$T(n) = 2T(n/2) + O(n)$$

For three way merge sort:

$$T(n) = 3T(n/3) + O(n)$$

a: number of sub problems : 3

b: input size shrinks : 3

d: 1

$$a = b^d$$

$$3 = 3^1$$

$$3 = 3.$$

$$T(n) = O(n \log_3 n)$$

DATE DAY M T W T F S S

Question # 03

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

Pivot = 6

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

i j

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

j i

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

j i ⇒ j > i

Pivot = 18 Pivot = 5

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

i j i j

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

i j j i j > i

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

i j

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

j i ⇒ j > i

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

27 j i j i

Pivot = ~~18~~ Pivot = 12

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

i j i j

~~31, 27~~ 12, 16, 9, 6

27, 31, 18 i j

j i ⇒ j > i 12, 16, 9, 6

31, 27, 18 j i ⇒ j > i

j i 16, 12, 9, 6

Pivot = 31 j i

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

Resultant list of integers:

37, 27, 18, 16, 12, 9, 6, 5, 1

Page No.

DATE DAY M T W T F S S

loop Invariant:

INITILIZATION: Before Loop starts condition of loop invariant is satisfied because the pivot and sub array $A[p \ldots i]$ and $A[i+1, \ldots j-1]$ are empty

MAINTENANCE: During iteration if $A[j] \leq$ pivot then $A[j]$ and $A[i+1]$ are swapped and then i and j are incremented if $A[i] >$ pivot then only j increment

TERMINATION: When loop ends the elements on left side of pivot are $\leq$ pivot and on right side of pivot are $>$ pivot.

Question #05

```
MinMaxSum (A) {
    len = A.Length               — 1
    Quicksort (A, 1, n)   sorts Array O(n log n)
    ~~[illegible]~~
    MAX_SUM = INT_MIN            — 1
    for j=1 to n/2               — n/2 + 1
        sum = A[j] + A[n-j+1]    — n/2
    if sum > MAX_SUM             — n/2
        MAX_SUM = sum            — n/2.
    return MAX_SUM               — 1
}.
```

Page No.

DATE DAY M T W T F S S

Question #06

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

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

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

$n_0 = 1 \quad c = 2$

Now if we check.

$2n^3 \le (2)(1)^3$

Proved.

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

$5n^2 \log_2 n + 2n^2 \le 5n^2 \log_2 n + 2n^2 \log_2 n$

$5n^2 \log_2 n + 2n^2 \le 7n^2 \log_2 n$

$c = 7$

$n_0 = 2$

Question #07

We consider scalibility, time complexity, and space complexity to find ~~time~~ efficiency of an algorithm. For complexities we need the input size. The O, Ω, Θ notation are used to relate growth of function

BIG O: The graph of $f(x)$ remain at right of $c * g(x)$ after some point x_0 so we say $f(x)$ is Big O of $g(x)$

BIG Ω: The graph of $f(x)$ form from below the graph of $c * g(x)$ we say $f(x)$ is Big Omega of $g(x)$ after point x

Page No.

DATE   DAY M T W T F S S

Big Ω : If graph $f(x)$ bond $g(x)$ from above and below then we say that $f(n)$ is in Big theta of $g(x)$ after some point.

All above Asymptotic bonds are limit of the functions

Question #08 :

$T(n) = 2T\left(\frac{n}{3}\right) + cn^2$

$a = 2, \ b = 3, \ d = 2$

$a \leq b^d$

$2 \leq 3^2$

so $2 < 9$.

$\therefore O(n^d) = O(n^2)$.

$T(n) = O(n^2)$

$T(n) = 4T\left(\frac{n}{3}\right) + c \cdot n$

$a = 4, \ b = 3, \ d = 1$

$a > b^d$

$4 > 3^1$

$4 > 3$

$\therefore O(n^{\log_b a}) = O(n^{\log_3 4})$.

$T(n) = O(n^{\log_3 4})$.

$T(n) = 8T\left(\frac{n}{2}\right) + c \cdot n^3$.

$a = 8, \ b = 2, \ d = 3$.

$a = b^d$

$8 = 2^3$

$8 = 8$

$\therefore O(n^d \log n) = O(n^3 \log n)$

$T(n) = O(n^3 \log n)$

DAY M T W T F S S

DATE

Question # 09

$$T(n) = 2T\left(\frac{n}{3}\right) + n^2$$

$$T(n) = \begin{cases} 1 & n = 1 \\ 2T(n/3) + n^2 & n > 1 \end{cases}$$

$$T(n) = 2T\left(\frac{n}{3}\right) + n^2$$

$$T(n) = 2\left[2T\left(\frac{n}{9}\right) + \frac{n^2}{9}\right] + n^2$$

$$T(n) = 4T\left(n/9\right) + \frac{2n^2}{9} + n^2 \quad — \ 1$$

$$T(n) = 4\left[2T\left(\frac{n}{27}\right) + \frac{n^2}{81}\right] + \frac{2n^2}{9} + n^2$$

$$T(n) = 8T\left(\frac{n}{27}\right) + \frac{4n^2}{81} + \frac{2n^2}{9} + n^2$$

$$T(n) = 2^k T\left(\frac{n}{3^k}\right) + \underbrace{n^2\left(\frac{4}{81} + \frac{2}{9} + 1\right)}_{\text{geometric series}}$$

$\therefore \ x = 2/9 \quad x \neq 1 \qquad \therefore \ \frac{n}{3^k} = 1$

$$\frac{1 - (2/9)^{k+1}}{1 - 2/9} \qquad n = 3k \Rightarrow k = \log_3 n$$

$$T(n) = 2^{\log_3 n} T(1) + \frac{9n^2}{7} - \frac{9n^2 (2/9)^{k+1}}{7}$$

$T(n) = \frac{9n^2}{7}$ (high order Term).

$$T(n) = O(n^2)$$

Page No.

DATE DAY M T W T F S S

$T(n) = 4T\left(\frac{n}{3}\right) + n$

$$T(n) = \begin{cases} 1 & n = 1 \\ 4T(n/3) + n & n > 1 \end{cases}$$

$T(n) = n + 4T\left(\frac{n}{3}\right)$.

$T(n) = n + 4\left[\frac{n}{3} + 4T\left(\frac{n}{9}\right)\right]$

$T(n) = n + \frac{4n}{3} + 16T\left(\frac{n}{9}\right)$.

$T(n) = n + \frac{4n}{3} + 16\left[\frac{n}{9} + 4T\left(\frac{n}{27}\right)\right]$.

$T(n) = n + \frac{4n}{3} + \frac{16n}{9} + 64T\left(\frac{n}{27}\right)$.

$T(n) = n\left(1 + \frac{4}{3} + \frac{16}{9}\right) + 4^K T\left(\frac{n}{3^K}\right)$.

$\therefore \left(\frac{x^{K+1} - 1}{x - 1}\right) \quad x \neq 1 \qquad \therefore \frac{n}{3^K} = 1$

$\frac{(4/3)^{K+1} - 1}{4/3 - 1} \Rightarrow \frac{(4/3)^{K+1} - 1}{1/3}. \qquad n = 3^K \Rightarrow K = \log_3 n$

$T(n) = n\left(\frac{(4/3)^{K+1}}{3^{\log n + 1}} - \frac{1}{3}\right) + 4^{\log_3 n} T(1)$.

$= 3n\,(4/3)^{K} - \frac{3n}{3} + 4^{\log_3 n}(1)$.

$= 3n\left(\frac{4^{\log_3 n + 1}}{3^{\log_3 n + 1}}\right) - 3n + 4^{\log_3 n}$

$= 3n\left(\frac{n^{\log_3 4}}{n^{\log_3 3}}\right) - 3n + 4^{\log_3 n}$

$T(n) = n^{\log_3 4}$ (high order term)

$T(n) = O(n^{\log_3 4})$

DATE
DAY M T W T F S S

QUESTION #10 :

(a)

let $f(n) = n^2$ $\quad h(n) = n$

$g(n) = n^3$

$f(n) = O(n^3)$

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

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

$n^3 + n = n^2$

$n^3 + n \geq n^2$

$n^3 + n \in \Omega(f(n))$

$n^3 + n \in \Omega(n^2)$ True

(b)

$f(n) = n$

$g(n) = n^2$

$\max(n^2, n) = n^2$

so

$n^2 \in \cancel{\Theta(f(n))}\ \Theta(n^2 + n)$

Therefore

$\max(f(n), g(n)) = \Theta(f(n) + g(n))$ True

(c)

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

$c_1(g(n)) \leq f(n) \leq c_2\, g(n)$

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

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

$f(n) = n+1$ $\quad g(n) = n$

$(f(n))^2 = O(g(n)^2) \Rightarrow (n+1)^2 = O(n^2)$

$n^2 + 2n + 1 \in O n^2$

True

Page No.