Name: Asif Alam Enroll no.: 92201703058

Subject: DAA Class: SEC3

Qno. 1. What is an algorithm? Explain various properties of andgorithm

ans:- An algorithm is a set of instructions that are designed to complete a task or solve a problem. The steps in an algorithm must be followed in the correct order to produce consistent result.

- Properties of algorithm:

To evaluate an algorithm we have to satisfy the following criteria:

1) Input: The algorithm should be given zero or more input.
2) output: Atleast one quantity is produced. For each input the algorithm produced value from specific task.
3) Definiteness: Each instruction is clear and an unambiguous.
4) Finiteness: If we trace out the instructions of an algorithm, then for all cases, the algorithm terminates after a finite no. of steps.
5) Effectiveness: Every instruction must very basic so that it can be carried out, in principal, by a person using only pencil & paper.

Ex- Algorithm to find the largest no. among 3 numbers.

```
step 1: start
step 2  Read A, B, C
step 3: check if  a > b
        if a > c  then print a is largest
        else
        if b > c  then print b is largest
        else
        print c is largest
step 4: stop
```

Name: Asif Alam
Enroll: 9220170305B

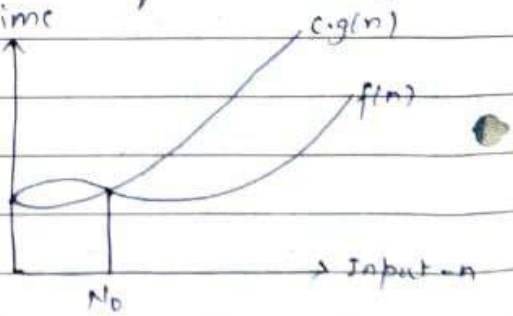
Qno2. Explain : Big Oh , Big Theta and Big Omega.

ans :- Asymptotic notations are used in asymptotic analysis of an algorithm refers to defining the mathematical boundaries of its run time performance.

⇒ It is used to find the best case, average case and worst case scenario of an algorithm.

⇒ The time required by an algorithm falls under three types.
- Best case - Minimum time required for program execution.
- Average case - Average time required for program execution.
- Worst case - Maximum time required for program execution.

- The notation $O(n)$ is the formal way to express the upper bound of an algorithm's running time. It measures the worst case time complexity or the longest amount of time an algorithm can possibly take to complete.



$f(n) \leq c\,g(n)$ for $n \geq n_0$

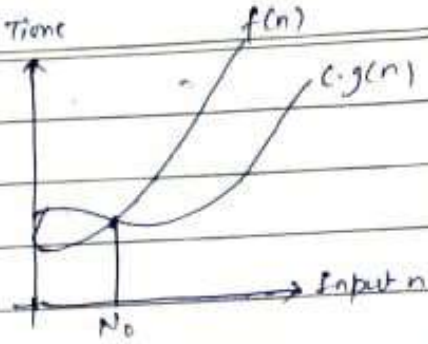
As n increase, $f(n)$ grows no faster than $g(n)$. In other words, $g(n)$ is an asymptotic ~~to~~ upper bound on $f(n)$.

- The notation $\Omega(n)$ is the formal way to express the lower bound of an algorithm's running time. It measures the best time complexity or the best amount of time an algorithm can possibly take to complete

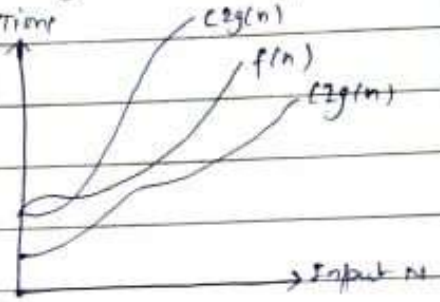
$f(n) \geq c \cdot g(n)$ for $n \geq n_0$

Name: Asif Alam
Enroll: 9220170305 8

g(n) should be as large as possible, gives the best running time.



- The notation $\theta(n)$ is the formal way to express both the lower bound and the upper bound of an algorithm's running time. Big-theta notation is used to define the average bound of an algorithm in terms of time complexity.



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

if there exist positive constant n_0, c_1 and c_2 such that to the right of n_0 the value of $f(n)$ always lies between $c_1 g(n)$ and $c_2 g(n)$.

Qno 3. Prove that: $20n^2 + 2n + 5 = \theta(n^2)$

sol^n- From big oh notation equation

$f(n) <= c * g(n)$ and $g(n) = n^2$

here, $f(n) = 20n^2 + 2n + 5$

so, $20n^2 + 2n + 5 <= c * g(n)$

$20n^2 + 2n + 5 <= c * n^2$

Let $c = 21$

so $20n^2 + 2n + 5 <= 21n^2$

$2n + 5 <= n^2$

n	$f(n) = 2n+5$	$g(n) = n^2$	$f(n) <= c * g(n)$
1	7	1	False
2	9	4	False
3	11	9	False
4	13	16	True

Name: Asif Alam
Enroll: 9220170305S

So, we can take n = 4 and c = 21

$2n+5 <= c * g(n)$

$2 \times 4 + 5 <= 21 * 16$

$13 < 336$

Hence proved $2n^2 + 2n + 5 = O(n^2)$

Qno4.) $f(n) = 2n^2 + n$, is $f(n)$ $\Omega(g(n^2))$?

soln. From big-omega notation equation -

$f(n) >= c \cdot g(n)$

here, $f(n) = 2n^2 + n$

$g(n) = n^2$

So, $2n^2 + n \geq c * n^2$

let's take c = 2

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

$n \geq 0$

For all the value greater than or equal to zero given equation hence proved.

Qno5 What is Amortized analysis? Explain aggregate method with suitable example.

soln. Amortized analysis means finding an average running time per operation over a worst case sequence of operation.

- It can used to show that -

The average cost of an operation is small, if one averages over a sequence of operations, even though a single operation within

Name: Asif Alam
Enroll: 92201703058

the sequence might be expensive.

=> There are 3 techniques of Amortized analysis
1) Aggregate analysis
2) Accounting method
3) Potential method

- Aggregate analysis :- show that a sequence of n operations take T(n) time. we can then say that the amortized cost per operation is T(n)/n.

Makes no distinction between operation types.

=> stack operations :

Push(S, x)	→	O(1)
Pop(S)	→	O(1)
Multipop(S, k)	→	O(min(S, k))

Let us consider a sequence of n Push, Pop, Multipop. The worst case cost for multipop in the sequence is O(n). Since the stack size is at most n.
Cost of sequence is $O(n^2)$

$$\text{Aggregate analysis} = \frac{\text{Total cost of } n \text{ operation}}{n}$$

Assign the amortized cost of each operation to be the average cost. In this example therefore, all three stack operations have an amortized cost of O(1).

Name: Asif Alam
Enroll: 9220170305S

Ans 6. Solve the following recurrence relation using master's theorem.

1) $T(n) = 4T(n/2) + n^2$

soln- given $a = 4, b = 2, K = 2$, and $f(n) = n^2$

$\therefore \log_b a = \log_2 4 = 2$

here $\log_b a = K = 2$

and $p > -1 \quad (\because p = 0)$

$\therefore T(n) = O(n^K \log^{p+1} n)$

$= O(n^2 \log n)$

2) $T(n) = 2T(n/2) + n\log n$

soln- given $a = 2, b = 2, K = 1$ and $f(n) = n\log n$

$\therefore \log_b a = \log_2 2 = 1$

here $\log_b a = K = 1$

and $p > -1 \quad (\because p = 1)$

$\therefore T(n) = O(n^K \log^{p+1} n)$

$= O(n \log^2 n)$

3) $T(n) = 2T(n/2) + n^2 \log n$

soln- given $a = 2, b = 2, K = 2$ and $f(n) = n^2 \log n$

$\therefore \log_b a = \log_2 2 = 1$

here $\log_b a < K$

and $p \geq 0 \quad (\because p = 1)$

$\therefore T(n) = O(n^K \log^p n)$

$= O(n^2 \log n)$

Name : Asif Alam
Enroll : 9220170305S

4.) $T(n) = 9T(n/3) + 1$

Sol^n- given $a = 9$, $b = 3$, $k = 0$ and $f(n) = 1$

$\therefore \log_b a = \log_3 9 = 2$

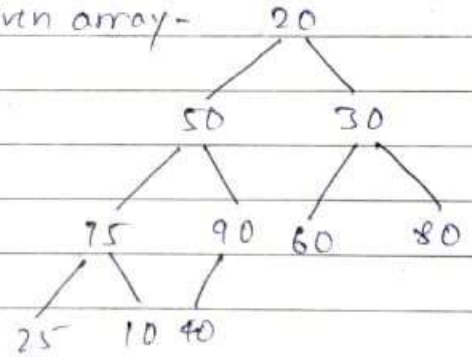
here $\log_b a > k$

$\therefore T(n) = O(n^{\log_b a})$

$= O(n^2)$

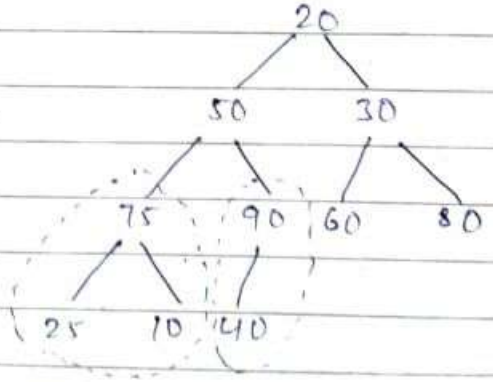
Qno 7. Sort data <20, 50, 30, 75, 90, 80, 25, 10, 40> using heap sort.

Sol^n- Binary tree of given array-



Heapify :

(I) Starting from lower level and check if it is max-heap or not if not then convert it to max heap.

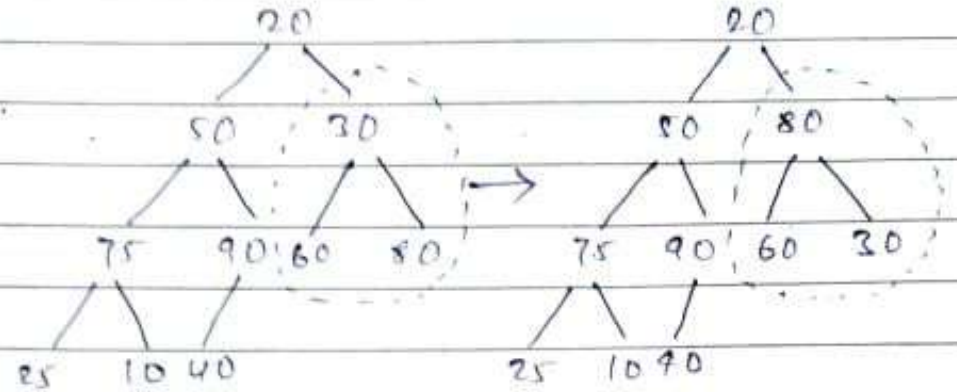


II subtree which has root 75 and 90 are already sorted.

Name: Asif Alam

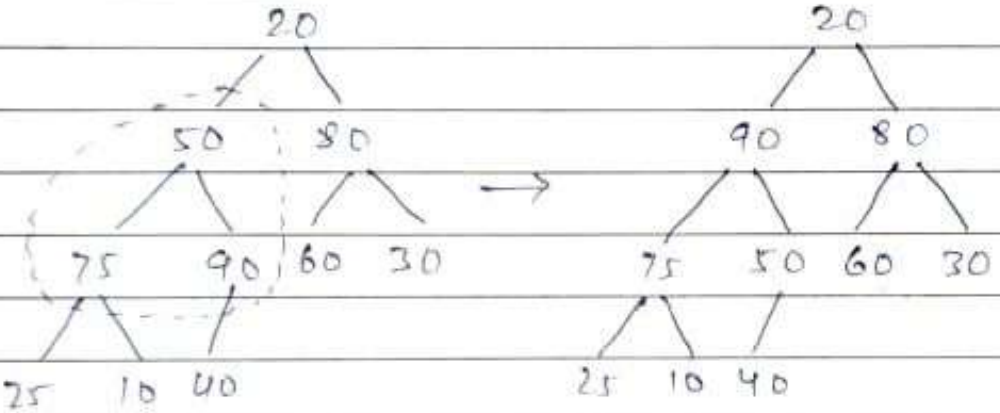
Enroll: 9220170305S

III



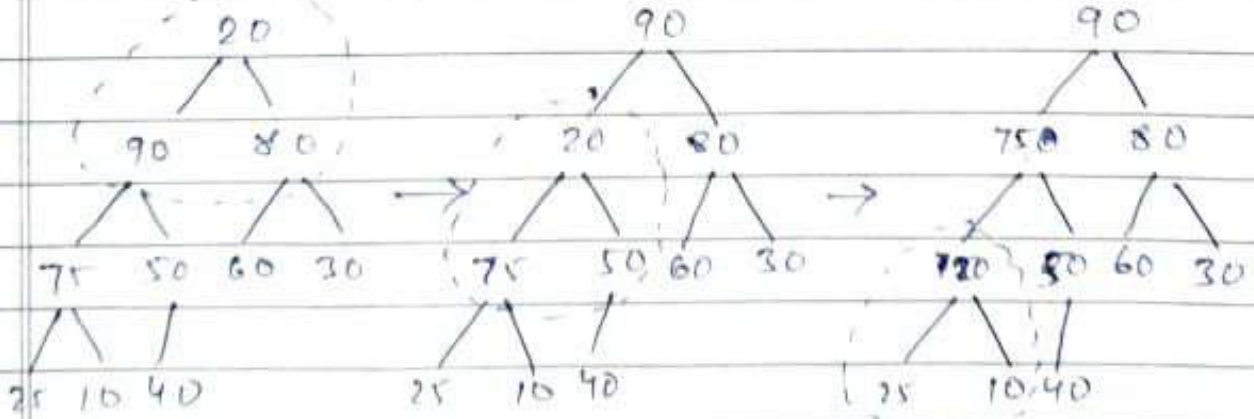
Swapped the subtree root 30 with his child 80 to maintain the max-heap.

IV



Swapped the subtree root 50 with his child 90 to maintain the max heap.

V

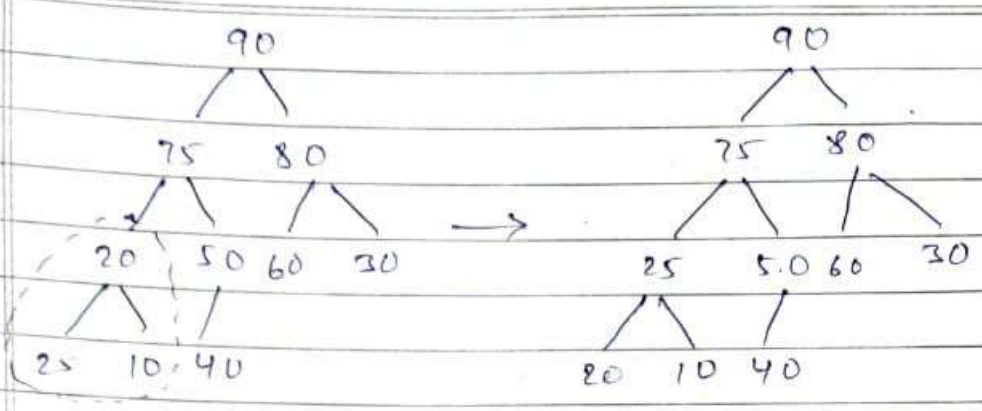


Swapped 20 with 90 and again 20 with 75.

Name: Asif Alam

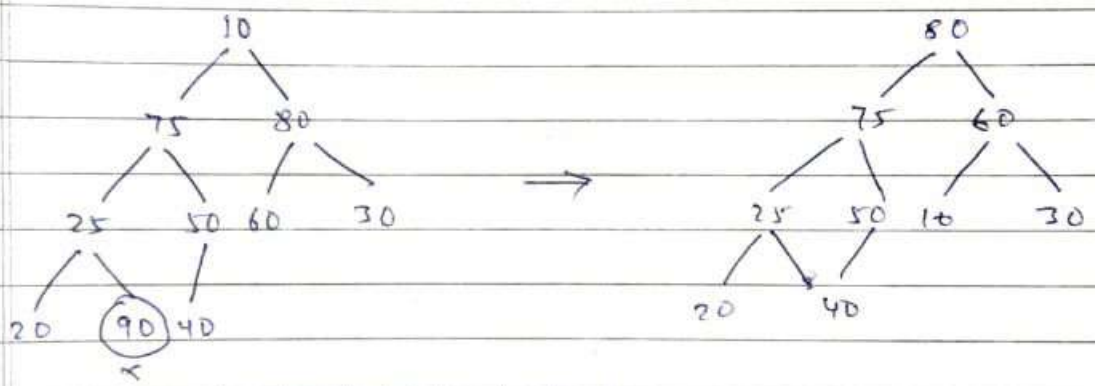
Enroll: 9220170305B

VI

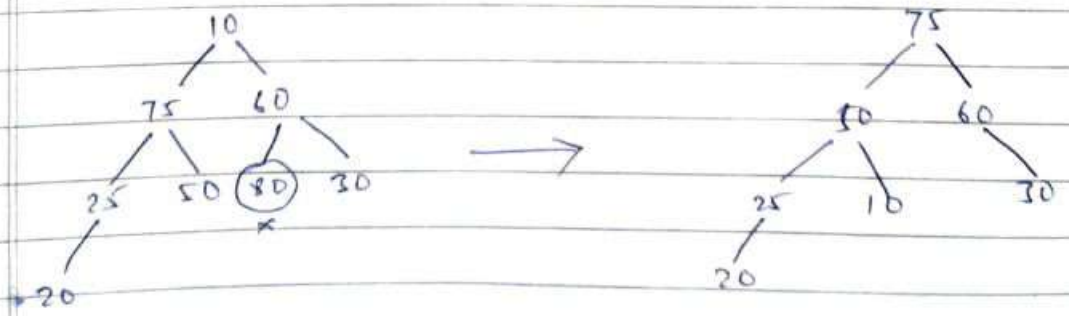


After swapping 20 with 25 we get the resultant tree.

- Perform the heap sort :



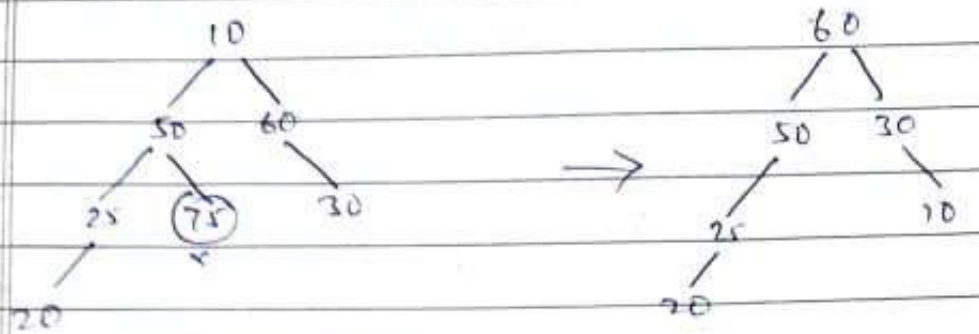
10	75	80	25	50	60	30	20	40	90



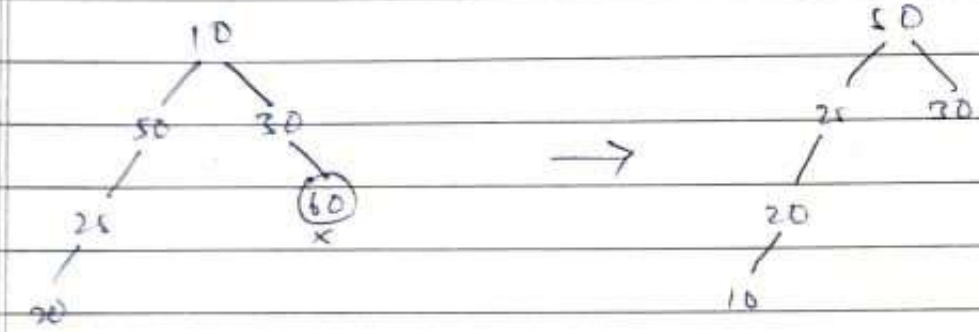
10	75	60	25	50	80	30	20	80	90

Name: Asif Alam

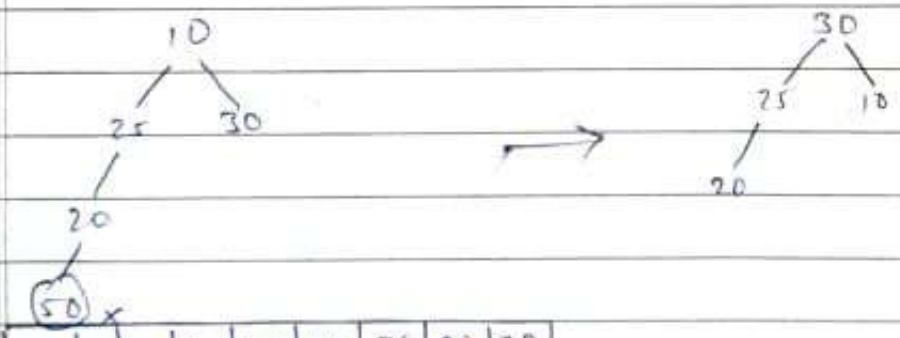
Enroll: 92201703053



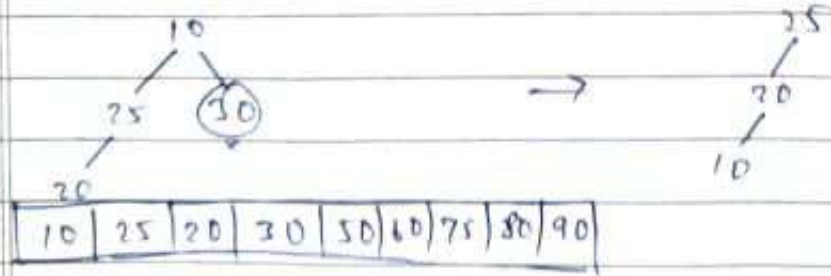
10	50	60	25	30	20	75	80	90



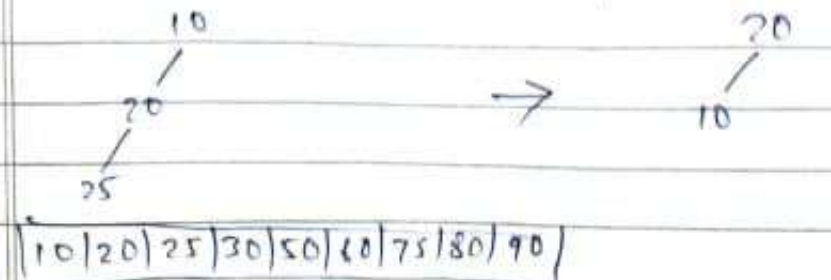
10	50	30	25	20	60	75	80	90



10	25	30	20	50	60	75	80	90



10	25	20	30	50	60	75	80	90



10	20	25	30	50	60	75	80	90

Name: Asif Alam
Enroll: 9220170 3058

Qno 8. Solve the following recurrence equation using tree method.

1. $T(n) = 2T(n/2) + n$ Here $T(1) = 1$

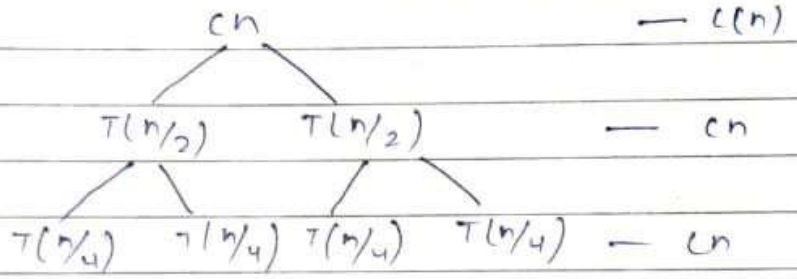
Soln. from given equation,

Root node = n

Size of sub problem = $T(n/2)$

No. of sub problem = 2

step 1: Find cost of each level



so, cost of each level is cn (same)

step 2: Depth of tree = $\frac{n}{2^i} = 1$

$i = \log_2 n$

level = depth + 1

$= \log_2 n + 1$

step 3: Total cost = cost of each level * No. of level

$= cn \times (\log_2 n + 1)$

$= c(n \log_2 n) + cn$

$T(n) = O(n \log_2 n)$

Name: Asif Alam

Enroll: 9220170305S

ii.) $T(n) = 3T(n/4) + n^2$ Here $T(1) = 1$

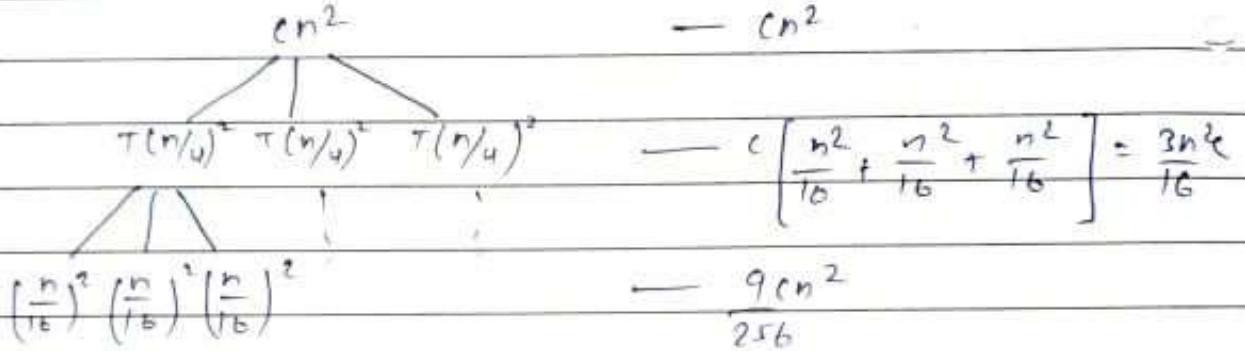
sol^n- from given equation -

Root node $= n^2$

Size of sub problem $= T(n/4)$

No. of sub problem $= 3$

step 1: Find the cost of each level



cost of each level $= \left(\frac{3}{16}\right)^i cn^2$

step 2: Find depth of tree $= \frac{n}{4^i} = 1$

$n = 4^i$

By taking log both side

$\log n = i \log 4$

$\therefore \boxed{i = \log_4 n}$

step 3: Total cost : $T(n) = cn^2 + \frac{3}{16}(n^2 + \ldots + \frac{3}{16}(d-1)(n^2)$

$= O(n^2)$

Name : Asif Alam

Enroll : 92201703058

Ques 9. Write an algorithm for insertion sort. Analyze insertion sort algorithm for best case and worst case.

Ans-

Algorithm insertion (a, n)

1. for i = 1 to (n-1) do
   1. temp = a[i]
   2. ptr = i-1
   3. while (temp < a[ptr] and ptr >= 0)
      a[ptr + 1] = a[ptr]
      ptr = ptr - 1
   4. a[ptr + 1] = temp
2. End

- Best case : $O(n)$

List is already sorted. In each iteration, first element of unsorted list compared with last element of sorted list, thus (n-1) comparison.

- Worst case : $O(n^2)$

List is sorted in reverse order. First element of unsorted list compared with one element of sorted list, second compared with 2 element, last element to be inserted compared with all the n-1 elements.

$$1 + 2 + 3 + \dots + (n-2) + (n-1)$$

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

$$= n^2$$

$$= O(n^2)$$

Name: Asif Alam

Enroll: 92201703058

Qno 10. write an algorithm to solve 5^9 using exponential method.

ans - $5^9 = 5 \times 5^8$

$= 5 \times (5^4)^2$

$= 5 \times ((5^2)^2)^2$

1 multiplication and 3 times square.

```
Exponential Algorithm : Ex( a , n) {
        if  n == 1  then  return a
        if  n % 2 == 0  then return  Ex(a^(n/2))^2
        else return  a * Ex(a^(n-1))
}
```

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

Qno 11 Solve the following fractional knapsack problem using greedy There are five items whose weights and values are given in following arrays and total capacity is 15.

weight w[] = {1, 2, 5, 6, 7}

value v[] = {1, 6, 18, 22, 28}

ans:- In greedy approach first we have to calculate v/w then sort the value and weight according to their v/w ratio.

w[] = {1, 2, 5, 6, 7}

v[] = {1, 6, 18, 22, 28}

v/w [] = {1, 3, 3.6, 3.66, 4}

As we can see v/w ratio is already sorted, then we have to take weigh according to capacity from back of array.

Name: Asif Alam

Enroll: 92201703058

Capacity = 15

V/w [] = { 1, 3, 3.6, 3.66, 4 }

x_1, x_2, x_3, x_4, x_5

$x_5 = 1$	Item	Capacity	Value
	x_5	$15 - 7 = 8$	28
	x_4	$8 - 6 = 2$	50
	x_3	$2 - \frac{x}{10} = 0$	$50 + \frac{2}{5} 18$

$= \frac{2}{1} - \frac{13}{20}$

$= 1.35$

Total value = $50 + \frac{2}{5} 18$ (3.6)

$= 50 + 7.2$

$= 57.2$

Fesiable sol^n = $\{ x_1, x_2, x_2, x_3, x_4, x_5 \}$

$= \{ 0, 0, 0, \frac{2}{5}, 1, 1 \}$

$= 0 + 0 + 0 + \frac{2}{5} \times 5 + 6 + 7$

$= 15$

optimal sol^n = total value = $\sum v_i x_i$

$= 28 + 22 + \frac{2}{5} 18$

$= 57.2$

Qno12. write greedy algorithm for activity selection problem. Give its time complexity. For following intervals, selects the activities according to your algorithm. 1|(1-3), 2|(0-2), 3|(3-6), 4|(2-5), 5|(5-8), 6|(3-10), 7|(7-9).

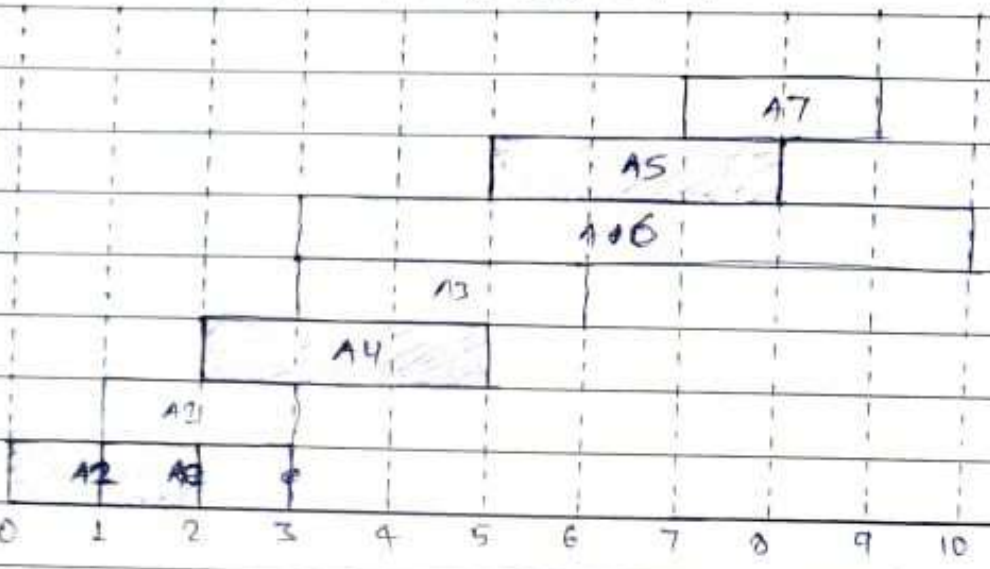
Name: Asif Alam

Enroll: 92201703038

Sol^{n} -

Activity	1	2	3	4	5	6	7
S_i	1	0	3	2	5	3	7
F_i	3	2	6	5	8	10	9

A2, A1, A4, A3, A6, A5, A7



F.S : { A2, A4, A5 }

= A2|(0-2) , A4|(2-5) , A5|(5-8)