[No extra sheet will be provided. Write your answer to the questions in this answer script.]
[Marks allocated to each question is given in the statement of corresponding question.]

1) Which of the following is **not** an asymptotic upper bound for $f(n) = 3n^2$? 1

a) n^2
b) n^3
c) $5n^2$
d) $20n$

$f(n) = O(3n^2)$

2) What is the runtime of the following code segment? 1

```
cnt = 0;
for (i=1; i<10; i++) {
      for (j=0; j<n; j++) {
            cnt++;
      }
}
```

a) $O(n^2)$
b) $O(\log^2 n)$
c) $O(n\log n)$
d) $O(n)$

3) Which of the following has the worst runtime? 1

a) n^3
b) n^{10}
c) 2^n
d) $n\log n$

$n^{\log_2 5}$

4) $O(m) + O(n) = ?$ 1

a) $O(m+n)$
b) $O(mn)$
c) $O(\max(m, n))$
d) None of the above

5) What is the runtime of $T(n) = 5T(n/2) + 4n^2$? 1

a) $O(n)$
b) $O(n^2)$
c) $O(n^2\log n)$
d) $O(n^{\log_2 5})$

$a = 5$
$b = 2$
$\Rightarrow 2^2 < 5$
$n^{\log_2 5}$

6) Sort the following runtimes from fastest to slowest: n^3, nlogn, logn, n, n^{100}, 2^n, $\sqrt{n}$. 3

3/ $\log n, \sqrt{n}, n, n\log n, n^3, n^{100}, 2^n$

7) Show that $(n^3 + 5n^2) = \theta(n^3)$. 6

$(n^3+5n^2) = \Theta(n^3) \Rightarrow (n^3+5n^2) = \Omega(n^3)$

$n^3+5n^2 \leq c_1 n^3$ $\quad$ $n^3+5n^2 \geq c_1 n^3$

let, $c_1 = 1$ $\quad$ $\Rightarrow$ let, $c = 1$.

$\cancel{n^3+5n^2}$

$n^3+5n^2 \leq n^3$ $\quad$ $n^3+5n^2 \geq 1 \cdot n^3$

$(n^3+5n^2) = O(n^3)$

let, $c = 2$ $\quad$ for $n_0 = 0$

6/ $(n^3+5n)^2 \leq 2n^3$

$5n^2 \leq n^3$ $\quad$ since, we can prove

$(n^3+5n^2) \leq 2n^3$ $\quad$ $(n^3+5n^2) = O(n^3)$

for $n_0 = 5$ $\quad$ $(n^3+5n^2) = \Omega(n^3)$

we can say, $(n^3+5n^2) = \Theta(n^3)$

8) Find the runtime of the following recurrence relation: 6

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

$T(1) = 1$

$T(n) = T(n-1) + n$ $\quad$ $T(n-1) = T(n-2) + (n-1)$

$T(n) = T(n-2) + (n-1) + n$ $\quad$ $T(n-2) = T(n-3) + (n-2)$

$T(n) = T(n-3) + (n-2) + (n-1) + n$

$\vdots$

$T(n) = T(n-k) + (n-2) + (n-1) + n$.

$\Rightarrow$ Assume, $n-k = 1$

4/ $+T(1) + (2) + (3) \cdots (n-2) + (n-1) + n$.

$\Rightarrow \frac{n(n+1)}{2}$ $\quad$ $1 + \frac{n(n+1)}{2} \Rightarrow O(n^2)$

$\Rightarrow 0 + \frac{n(n+1)}{2} = O(n^2)$

1) What is the **worst-case** runtime of **quick sort**? Show an **example array** which will run into the worst-case scenario. What can we do to **avoid** the worst-case scenario? [3]

Ans: worst case runtime of quick sort is $O(n^2)$ /

pivot ↓

1	2	5	10	11	13	17	20

— Ascending order

3/

To avoid the problem, we need to pick the pivot at ~~mo~~ middle of array or at random~~po~~ ~~point~~ index

X /

2) **Partition** the given array taking the **first** element as the **pivot**. Show the necessary steps by **simulation**.
[11, 7, 13, 3, 12, 16, 5]. [7]

Ans:

X/

pivot (11), 7, 13, 3, 12, 16, 5

11, 7, 13, 3, 12, 16, 5

11, 7, 5, 3, 12, 16, 13

3, 7, 5, 11, 12, 16, 13

pivot (11), 7, 13, 3, 12, 16, 5

11, 7, 13, 3, 12, 16, 5

swapped

11, 7, 5, 3, 12, 16, 13

11, 7, 5, 3, 12, 16, 13

swapped

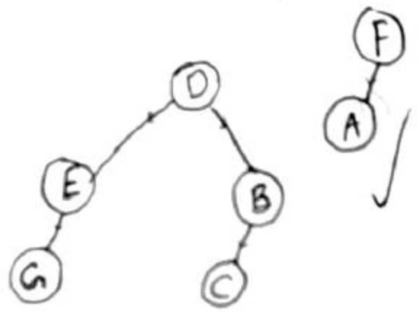
3, 7, 5, 11, 12, 16, 13

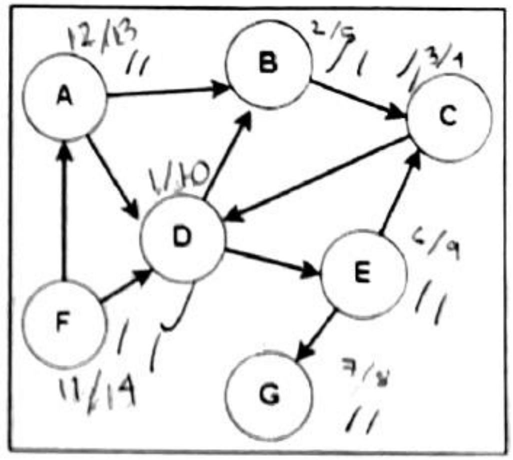
/

3) Taking node **D** as the **source**, use **DFS** to find the **discovery** and **finish times** of all the nodes in the following graph. Also draw the **DFS tree**. [7+3]

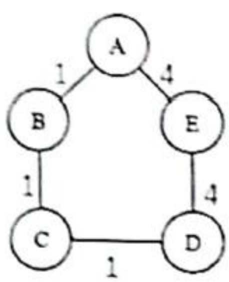
Node	A	B	C	D	E	F	G
Discovery time	12	2	3	1	6	11	7
Finish time	13	5	4	10	9	14	8

10

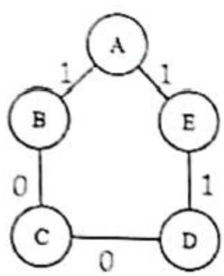




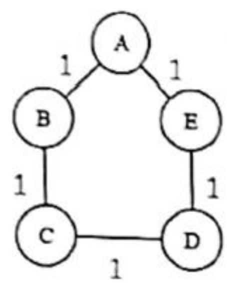
Question 1: [Points 0.25*4 = 1]



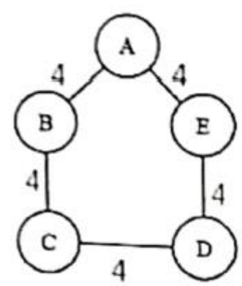
Graph 1



Graph 2



Graph 3



Graph 4

i) Implementing BFS on which graphs will provide the **correct** solution for any pair of source & destination?

Ans: Graph 3 and 4.

1. Check Whether $5n^3 \log_2{}^n + 7n^4 = \theta(n^4)$? (Here θ means Tight bound) (8)
2. Check Whether $7n^5 + 35n^2 + 10 = O(n^{10})$? (Here O means Upper bound) (12)
3. Find out the Worst Time Complexity of following Code Snippet: (10)

```
sum=0  O(1)
for ( i=1 ; i<=n ; i=i*5) {  O(log_5 n)
   for (j=1 ; j<=n ; j++) {  O(n)
        sum++;  O(1)
   }
}
```

30

Good

For Question no 1 and 2 properly mention the values of c and n_0. For Question no 3 properly mention the complexity of the each line and then compute the total time complexity of the whole code snippet.

1) $f(n) = 5n^3 \log_2 n + 7n^4 = \Theta(n^4)$

$2 = 112$, $3 = 7\cancel{8}81$, $4 = 12432$ | 192, 972, 2672

for tight bound:

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

$c_1 \cdot g(n) \leq f(n)$

$= c_1 \cdot n^4 \leq 5n^3 \log_2 n + 7n^4$

$\boxed{c_1 = 1,\ n_0 = 1}$

$1 \cdot 1^4 \leq 5(1)^3 \log_2(1) + 7(1)^4$

$1 \leq 7$

$\cancel{7 \cdot (2)^4 \leq 5(2)^3 \log_2(2) + 7(1)^4}$

$\cancel{16 \leq 5(2)^3 + 1}$

$\cancel{= 16 \leq 40}$

8

$f(n) \leq c_2 \cdot g(n)$

$7n^4 + 5n^3 \log_2 n \leq c_2 \cdot n^4$

$\cancel{c_2 = 5n^{4} \times \log_2 n^{4} + 7n^{4}}$

$\cancel{5n^4 \times 4 \log_2 n + 7n^4}$

$\cancel{= 20 n^4 \log_2 n + 7n^3}$

$\cancel{= 20n^4 \log_2 n + 7n^4}$

$\cancel{\text{if } n = 1}$, $\cancel{c_2}$ $c_2 = 7$.

$c_2 = 7n^4 + 5n^4$

$= 12$

$\boxed{\therefore c_2 = 12,\ n_0 = 1}$

$\therefore$ Yes, $5n^3 \log_2 n + 7n^4 = \Theta(n^4)$

if $c_1 = 1$, $c_2 = 12$, $n_0 = 1$

(ii) $7n^5 + 35n^2 + 10 = O(n^{10})$

12

$f(n) \leq c_1 \cdot g(n)$

~~$= 7n^5 + 35n^2 + 10 \leq c_1 \cdot n^{10}$~~

~~$\boxed{c_1 = 1,\ n_0 = 2}$~~

~~$7(2)^5 + 35(2)^2 + 10 \leq 1 \cdot 2^{10}$~~

~~$= 375 \leq 1024$~~

$\therefore$ Yes, $7n^5 + 35n^2 + 10 = O(n^{10})$

~~or,~~

$c_1 = 7n^5 + 35n^5 + 10n^5$

$= 7 + 35 + 10$

$= 52$

$\boxed{c_1 = 52,\ n_0 = 1}$

(iii) $\therefore O(1) + \{O(\log_5 n) \times O(n)\}$

10

$= O(1) + O(n \log_5 n)$

$= O(n \log_5 n)$

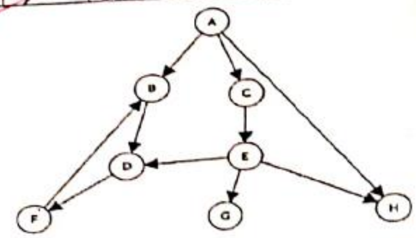
1. Suppose you have a array, A = [14, 11, 10, 8, 5, 2]. To Sort this array will you prefer Merge Sort or Quick Sort? Write 1 line defending your answer. (6)

The given array A is a reversely sorted array so, the time complexity for Quick sort will be $O(n^2)$. But incase of merge array, the time complexity will be $O(n \log n)$ which is better interms of time complexity. Thus, I will prefer merge sort.

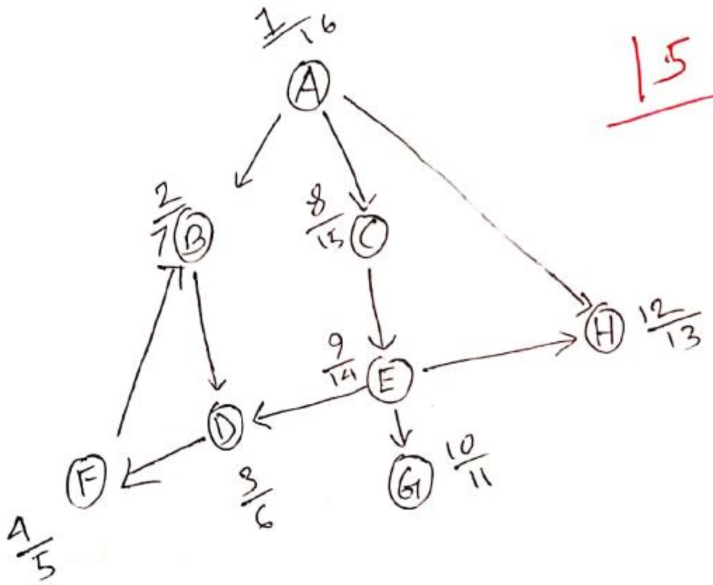
6

3. For the Graph shown below, Taking A as the Source vertex using DFS Algorithm calculate the Discovery time and Finish time. Initialize time=1. (15)

Vertex	A	B	C	D	E	F	G	H
Discovery time	1	2	8	3	9	4	10	12
Finish time	16	7	15	6	14	5	11	13



15



visit: A B D F C E G H

Answer all 4 questions. [2 + 5 + 5 + 3]

1. Write the formal definition of ***Big omega (Ω)***. Your definition must contain ∃ and ∀.

~~Ω(g(n)):~~ $\Omega(\cancel{cg(n)})$ f(n) : ~~for all~~ there exists a positive c and ~~no~~ n_0 for which $0 \le f(n) \geq cg(n)$ for $\forall\ n > n_0$

2. Mathematically prove how the **worst-case** running time of insertion sort is *$O(n^2)$*.

Assume

$$T(n) = c_1 n + c_2(n-1) + c_4(n-1) + c_5 \sum_{j=2}^{n} t_j + c_6 \sum_{j=2}^{n}(t_j - 1) + c_7 \sum_{j=2}^{n}(t_j - 1) + c_8(n-1) .$$

assuming ~~$t_j = 1$~~,

$$T(n) = c_1 n + c_2(n-1) + c_4(n-1) + c_5\left(\frac{n(n+1)}{2} - 1\right) + c_6\left(\frac{n(n-1)}{2}\right) + c_7\left(\frac{n(n-1)}{2}\right) + c_8(n-1)$$

$$= n\underbrace{\left(c_1 + c_2 + c_4 - \frac{c_5}{2} - \frac{c_6}{2} - \frac{c_7}{2}\right)}_{b} + n^2\underbrace{\left(\frac{c_5}{2} + \frac{c_6}{2} + \frac{c_7}{2}\right)}_{a} \underbrace{- c_2 - c_4 - c_5 - 2c_6 - 2c_7 - c_8}_{c}$$

$$= an^2 + bn + c$$

$$an^2 + bn + c = O(n^2)$$

3. *Find the **worst-case** Time complexity of the following snippets:*

step	i	j
0	1	0
1	2	1
2	4	3
3	8	7
⋮	⋮	$2^k - 1$
k	n	$n-1 = 2^k$

```
for( int bound = 1; bound <= n; bound *= 2 ) {
        for( int i = 0; i < bound; i++ ) {
                for( int j = 0; j < n; j += 2 ) {
                        ... // constant number of operations
                }
                for( int j = 1; j < n; j *= 2 ) {
                        ... // constant number of operations
                }
        }
}
```

first loop → lg n

second loop → n-1

third part → n + lg n

3

$O((\lg n)(n-1)(n + \lg n))$

$O((n \lg n - \lg n)(n + \lg n))$

$O(n^2 \lg n - n \lg n + n(\lg n)^2 - (\lg(n))^2) = O(n^2 \lg n)$

4. Simulate the approach of ***Binary Search*** operations for the following array. ***You must show full steps and <u>mid</u> values for each step.***

Search 27 in a sorted array with 10 elements

0	1	2	3	4	5	6	7	8	9
1	5	8	10	13	16	27	32	45	60

mid

5

5	6	7	8	9
16	27	32	45	60

mid

5	6
16	27

mid

6
27

27 returned

mid

Question 1: [5 points]

5

Explain the time complexity of the following code snippet?

```
j = 0
for (i = 0; i < n; i++) :   → n
        while j > n:
                j = j + 1
```

For the outer loop to run, $n > 0$.

If $n > 0$, then ⇒ $j < n$.

then the inner while loop condition will NOT be satisfied

nd value of j will never increase.

∴ only outer loop will run n times.

∴ $O(n)$

Question 2: [5 points]

Explain the time complexity of the following code snippet?

```
sum = 0
for (i = 0; i < n; i++):        → n
      for (j = n; j > i; j--):  → n
            sum = i + j         → 1
```

$\therefore\ O(n) \times O(n) \times O(1)$

$= O(n^2)$

i	j	
0	n	n times
0	n-1	
⋮	⋮	
0	1	
1	n	n-1 time
1	n-1	
⋮	⋮	
1	2	
⋮		
n-1	n	1 time

Complexity for 2 nested loops is $O(n^2)$

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

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

$\cancel{Of} = \frac{n^2}{2} + \frac{n}{2}$

$= O(n^2)$

$\frac{1}{2}n^2 + \frac{1}{2}n \leq 0$

$n(n+1) \leq 0$

Question 3: [5 Points]

Prove that, $\frac{1}{2}n^2 - \frac{1}{2}n = \Theta(n^2)$

$c_1 n^2 \leq \frac{1}{2}n^2 - \frac{1}{2}n \leq c_2 n^2$

$\frac{1}{2}n^2 - \frac{1}{2}n \geq c_1 n^2$

Let $c_1 = \frac{1}{4}$

2

$\therefore \frac{1}{2}n^2 - \frac{1}{2}n \geq \frac{1}{4}n^2$

$\Rightarrow \frac{1}{4}n^2 \geq \frac{1}{2}n$

$\Rightarrow n \geq 2$

$\therefore \frac{1}{4}n^2 \leq \frac{1}{2}n^2 - \frac{1}{2}n$

For $n_0 = 2$

Proved

$\frac{1}{2}n^2 - \frac{1}{2}n \leq c_2 n^2$

Let $c_2 = 1$

$\therefore \frac{1}{2}n^2 - \frac{1}{2}n \leq n^2$

$\Rightarrow \frac{1}{2}n^2 \geq -\frac{1}{2}n$

$\Rightarrow n \geq -1$

$\Rightarrow n \geq 0$

$\therefore \frac{1}{2}n^2 - \frac{1}{2}n \leq n^2$

For $n_0 = 0$

Proved.

$\therefore \frac{1}{2}n^2 - \frac{1}{2}n = \Theta(n^2)$ Proved.

5

Question 4: [5 points]

Find the time complexity using Master theorem. If it's not possible to find time complexity using Master theorem:

$T(n) = T(n-1) + \log n$

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

$= O(n \log n)$

$T(n) = T(n-1) + \log n$

$a\,T(n-b) + f(n)$

When $a=1$, $b=1$, $f(n) = \log n$

If $a=1$

$O(n * f(n))$

$= O(n \log n)$

5/5

Question 1: [5 points]

What does divide and conquer technique mean? Which sorting algorithms are based on this technique?

⇒ Divide and conquer tenchnique is a tecnique which is used in sorting algorithms. In this tecnique first an array is divided ~~an~~ over and over again. Finally, the divided parts are logically compared and merged~~d~~(conquered) together. This help sort arrays. Merge sort and Quick sort are based on this technique.

Question 2: [5 points]

Sort the list [7,2,5,1,3] using selection sort. Show every step.

~~Sorted~~ ~~Unsorted~~

~~[7, 5, 1, 3]~~

swap

[7, 2, 5, 1, 3]

sorted | unsorted

[1, | 2, 5, 7, 3]

sorted | swap

[1, 2 | 5, 7, 3]

unsorted

sorted | swap

[1, 2, 3, | 7, 5]

unsorted

[1, 2, 3, 5, 7]

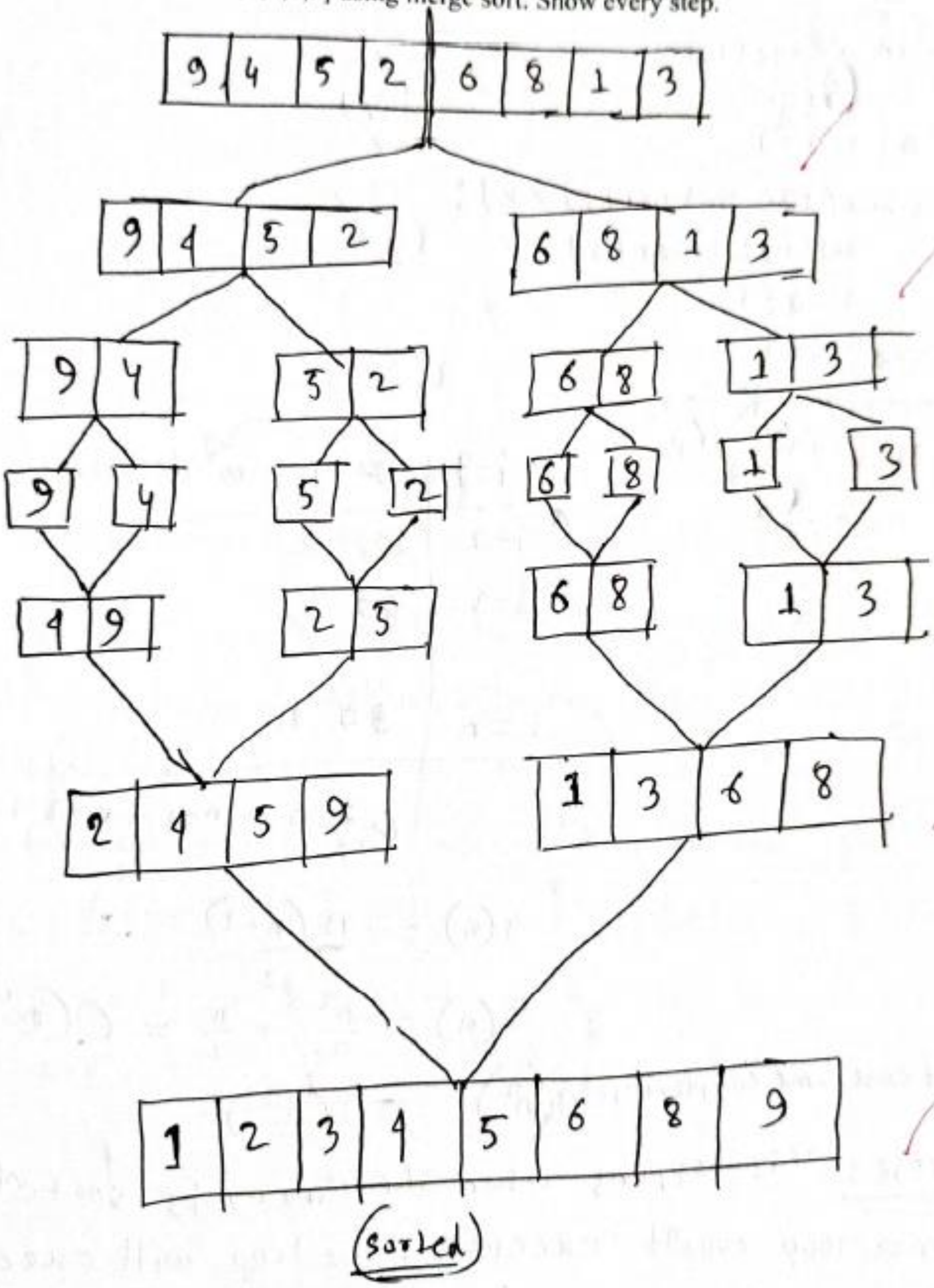
(Sorted)

5/5

/5

Question 3: [5 Points]

Sort the list [9,4,5,2,6,8,1,3] using merge sort. Show every step.



5

Question 4: [5 Points]

What is the best and worst case time complexity for insertion sort? Explain.

```
for i in range(1,N):
    k = arr[i]
    j = i-1
    while(j>0 and arr[j] > k):
        arr[j+1] = arr[j]
        j = j-1
  t=arr[j+1]
  ~~arr[i] = k~~
  arr[j+1] = k
```

worst case:

i	iterations
i=1	1
i=2	2
i=3	3
...	
i=n	n
	$= n + n-1 + n-2 + n-3 + \dots + 1$

$$T(n) = \frac{n(n+1)}{2}$$

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

$\therefore$ Worst case time complexity $= O(n^2)$ ~~$= O(n^2)$~~

Best case: this happens when the array is sorted.

$\therefore$ Inner loop won't execute. the loop will execute total $n-1$ times.

$\therefore$ Best case time complexity $= \Omega(n)$

1) Calculate the running time (f(n) or T(n)) of the code snippets in (a) and (b).
(Keep your elaboration as brief and short as possible) ———— 2+3 = 5

(a) 2	(b) 3

Pseudo code:
(n << input)
sum = 0
for (k = 1 ; k <= n ; k = k+1){
for (i = 0 ; i < n ; i *= 2){
sum += i*n
}
}
print(f"The code ran {sum} times")
	Pseudo code:
	(n << input)
	handshakes = 0
	X = [ ]
	for (k = n ; k >= 1 ; k = k-1){
	for (i = k-1 ; i >= 1; i = i-1){
	X.append(i)
	handshakes = handshakes + 1
	}
	print(f"Total number of handshakes are {handshakes}")

Answer:

(a)

~~step~~ i
0 0

for(K=1; K<=n; K=K+1) O(n)

for(i=0; i<n; i*=2){
this loop is infinite loop.
}

(b)

for (K=n; K>=1; K-1) O(n)

for(i=k-1; i>=1; i=i-1) O(n)

∴ the complexity = $O(n^2)$

2) Express each of the functions in column B as an asymptotic bound (upper, lower or tight) of the functions in column A. (for example : if $A = 3n^2$, $B = n^2$ you should write, $A = \Theta(B)$. It is a must to mention tight bound here so that the answer is more appropriate.) —— 4

3+1

A	B	Big - Oh / Big - Omega / Big - Theta ($O / \Omega / \Theta$)
$\sqrt[3]{n}$	$n^{\sin(n)}$	$A = O(B)$
$e^{\ln(n)}$	n^2	~~$A = \Theta(n^2)$~~ $A = O(B)$ ✓
n!	$n + 1^n$	$A = \Omega(B)$ ✓
5nlog(n)	$2\ln(e^n)$	$A = \Omega(B)$ ✓

3) Show the simulation of Binary Searching Algorithm for this list of integers : [34, 67, 42, 23, 14, 46, 37, 29, 52, 17, 49, 41] (0 indexed)

a) Search for 34. Show step number , low, mid, high for each step.

(Hint: Do all the necessary tasks needed) —— 5

indexes → 0 1 2 3 4 5 6 7 8 9 10 11

[14, 17, 23, 29, 34, 37, 41, 42, 46, 49, 52, 67]

5

step	low	High	mid	
0	0	11	5	✗
1	0	$(5-1)=4$	$(\frac{4+0}{2})=2$	✗
2	$(2+1=3)$	4	3	✗
3	$(3+1)=4$	4	4	✓✓

Ans;

(i) There is an undirected simple graph of 10 nodes. Nodes are labeled from 10 to 19. The edges are:

a) All the nodes that are labeled with a prime number have edges among them.
b) All the nodes that are labeled with a composite number have edges among them.

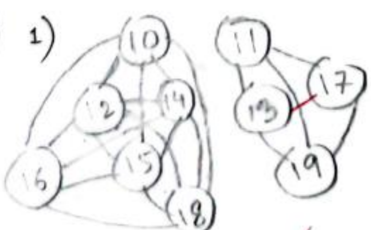
1) Draw the graph. — 3
2) How many new edges will be added to this graph if you are told to put an edge between every pair of prime and composite numbers. (meaning one edge will be adjacent to one prime and one composite number) — 1
3) Your friend, Jack ran a complete BFS traversal on the main graph, not the one from question (2) (means he reached all the nodes in that graph using BFS). How many trees will he get? What is the total number of edges in those trees? — 2
4) Jack wants to find the number of cycles in this graph using DFS traversal. Suggest him a solution for this task. — 3

(ii)

1) An unsorted array has the property that every element in the array is at most k distance from its position in the sorted array where k is a positive integer smaller than the size of the array. How will you modify the following sorting algorithms so that the sorting can be done in O(kn) time? (You don't need to write any code/algorithm) — 2
   a) Insertion Sort
   b) Selection Sort
2) After doing partition in the first step of Quick Sort algorithm the array looks like this : 19, 17, 15, 23, 27, 39, 32, 37
   a) What was the pivot before doing the partition? — 1
   b) If the given array had been : 24, 17, 15, 37, 39, 12, 32, 19. Show the simulation of the first partition of a Quick Sort algorithm in this array. — 3

Answer:

i) 1)



2.5

1

i) 2) 6×4 = 24 new edges

i) 3) He will get two trees
For first tree, 5 edges,
For second tree, 3 edges.

create a variable ~~named~~ 'cycles' = 0

i) 4) ~~Us~~ Use DFS traversal from any starting vertex. As it traverses through the edges and reaches its starting vertex, and the process; then cycles += 1. Repeat this process with other vertices ~~un~~till all the vertices are visited. Ultimately the value of 'cycles' provides the number of cycles.

2.5

ii)

1) a) For insertion sort, stop the loop when the index reaches kth position. Only swap values before kth position

2

ii) 1) b) For selection sort, find the minimum values that are before the kth position. And swap ~~sort~~ accordingly which only should occur ~~be~~ till the kth position.

ii) 2) 19, 17, 15, 23, 27, 39, 32, 37

a) 23 was the pivot before doing the partition. (Left partition from ~~of the~~ 23 have smaller values. Right partition from 23 have larger values).

1

b) pivot

[24], 17, 15, 37, 39, 12, 32, 19 ∞

i → → → i   j ← j

[24], 17, 15, 19, 39, 12, 32, 37

i → i   j ← ← j

pivot

[24], 17, 15, 19, 12, 39, 32, 37

i → i

[j] ← j

i crosses j, pivot ⇔ j

3

12, 17, 15, 19, [24], 39, 32, 37 → First partition done

pivot

Determine the worst case time complexity for the following code,

```
count = 0;                         1
for (x=1; x<=n; x++){n             n+1
        for (y=1; y<=x; y++){
                count++;
        }
}

for (z=0; z<n; z++){               n+1
        arr[z] = z;                n
}
```

Question 2: CO2 [4 Points]

Verify which one of the following relations is correct for **f(n) = 100 and g(n) = log 100**,

I. f(n) = O(g(n)) or
II. f(n) = Θ(g(n)) or
III. f(n) = Ω(g(n))

Question 3: CO2 [8 Points]

Solve the following recurrence relation using recursion tree method and find the worst case time complexity,

$T(n) = 2T(n/2) + O(n), T(1) = O(1)$

$f(n) = 100$ and $g(n) = \log 100$

$f(n) \leq c\, g(n)$ $\quad \log 100 = 2$

$\Rightarrow 100 \leq c \log 100$

$\Rightarrow 100 \leq c \times 2$ $\quad c_1 = 100$

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

4/4

We know,

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

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

$\Rightarrow 100 \geq c \log 100$

$\Rightarrow 100 \geq c \times 2$ $\quad c_2 = 1$

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

$\therefore c_2 \log 100 \leq 100 \leq c_1 \log 100$

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

7/8

1)

x	y	No. of times
1	1, 2×	1
2	1, 2, 3×	2
3	1, 2, 3, 4×	3
⋮		⋮
n		n

1st loop

terminate $y > x$

$1 + 2 + 3 + \dots + n$

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

$= n^2$

$O(n^2)$

Second loop.

$f(n) = 2n + 1$

$O(n)$

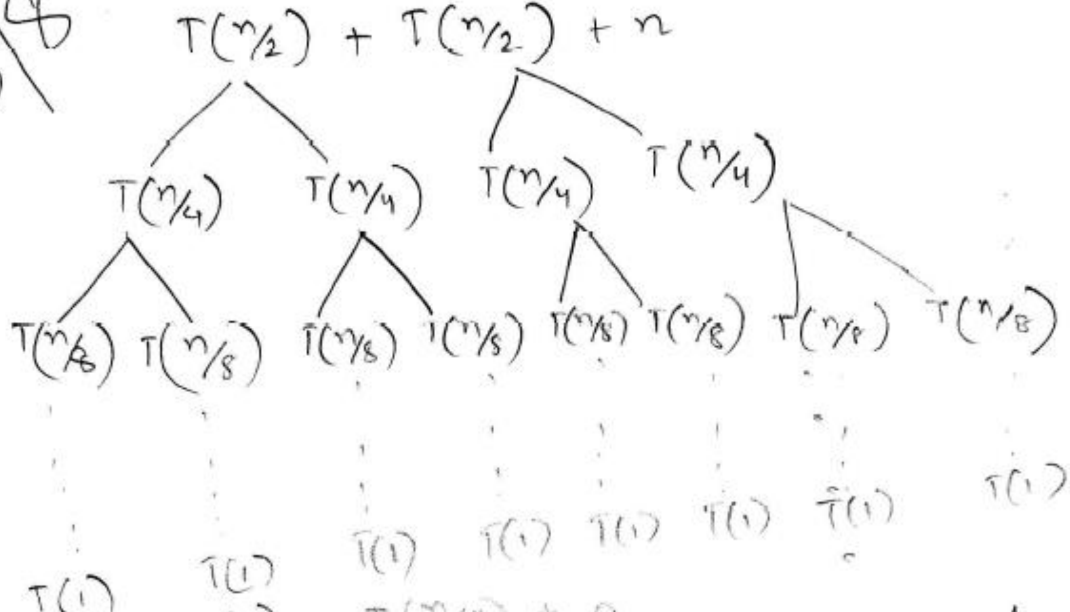
$O(n^2) + O(n)$

$\therefore O(n^2)$ as it is higher

3) $T(n) = 2T(n/2) + O(n)$ $T(1) = O(1)$

$= 2T(n/2) + n$ $\Rightarrow T(1) = 1$

8/8



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

$\frac{n}{2^k} = 1$ (Assume we have reached the base case)

$\Rightarrow n = 2^k$

$\Rightarrow k = \log_2 n$.

$\therefore O(n \log n)$

Master Theorem $a = 2$ $b = 2$

$\log_b^a$ $\log_2^2 = 1$ $k = 1$

both equal so case-2

$\therefore O(n \log n)$

Question 1: CO1, CO2 [14 Points]

Simulate the partitioning of Quick Sort algorithm on the following array. Mention which element you have selected as pivot. Show workings of each step in detail.

8	12	15	7	10	5	4	13

Question 2: CO1, CO2 [1+3+2 Points]

I. Mention when the worst case happens for Quick Sort algorithm.

II. Mention the worst case, average case and best case time complexity of Quick Sort algorithm with proper notation.

III. Explain whether Quick Sort is an in-place algorithm.

8	12	15	7	10	5	4	13
i				p			j

$j > pivot$
$j = j-1$

$i < pivot$
$i = i+1$

8	12	15	7	10	5	4	13
	i			p		j	

p → pivot

8	4	15	7	10	5	12	13
	i			p		j	

* Took a random pivot from the array

8	4	15	7	10	5	12	13
		i		p	j		

* if j<pivot and i>pivot,
swap(arr[i], arr[j])

8	4	5	7	10	15	12	13
			i	p, j			

8	4	5	7	10	15	12	13

→ This is the partition part

all the elements less than pivot

all the elements greater than pivot

2)I) $O(n^2)$

When the pivot is the min or max element

II Worst case:

$O(n^2)$

Best case:

$\Omega(n \log n)$

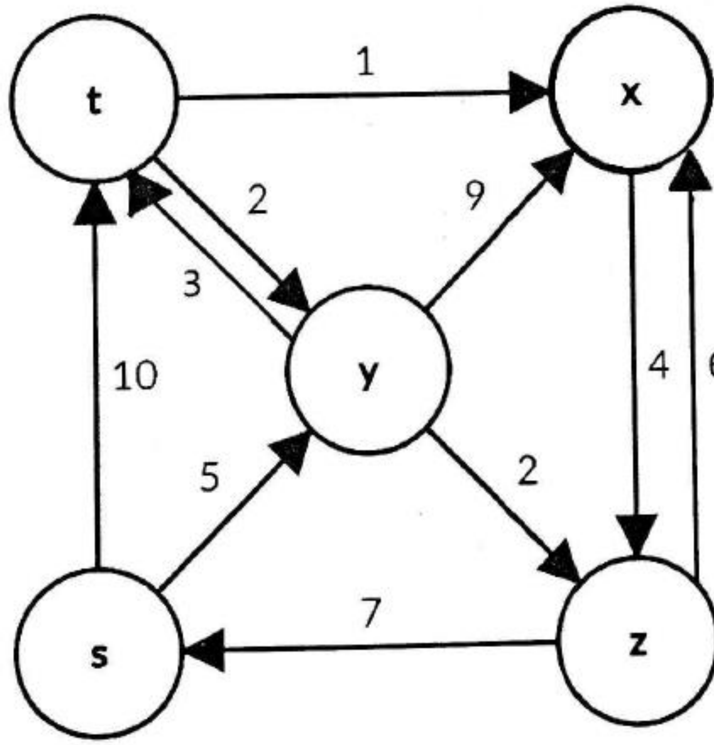
Average case:

$\theta(n \log n)$

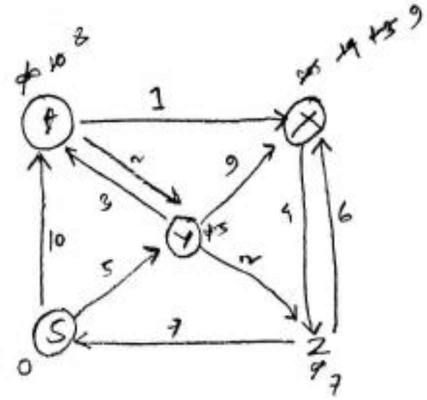
III Yes, it is an in-place algorithm so we don't need ~~an~~ new array. in Quick sort.

Question 1: CO3 [12+8 = 20 Points]

Simulate a suitable Algorithm on the following graph to determine the shortest path from vertex **s** to all other vertexes. Show your workings in detail by keeping track of the predecessor vertex and shortest distance.



Ans - to - the - Q - NO - 1



Vis: S, Y, Z, t, x

12/12

Node	distance	Parent
s	0	
t	~~∞ 10~~ 8	~~X~~ Y
x	~~∞ 14 13~~ 9	~~X Z~~ t
y	~~∞~~ 5	s
Z	~~∞~~ 7	Y

Shortest Path:

s to z: $s \to y \to z = 7$

s to Y: $s \to Y = 5$

s to t: $s \to Y \to t = 8$

s to x: $s \to Y \to t \to x = 9$

8/8

1. **[Marks 2] Prove that,** nlogn+n/3 = O(n)
2. **[Marks 4] Find the complexity:**

```
k=0
for (i = 0 ; i <= n; i=i+1)          } O(n)
    for (j = 0; j <= n; j = j / 2)   } O(∞)
        k = k + 1
```

3. **[Marks 4] Solve using substitution method,** T(n) = 4T(n/3) + n

$\boxed{1}$ $n\log(n) + \frac{\cancel{n}}{3} \leq \cancel{4n} - \frac{n}{3}$

$\Rightarrow \log(n) + \frac{\cancel{1}}{\cancel{3}} \leq \cancel{4}\ \frac{11}{3}$

$\Rightarrow n \leq \cancel{10^4}\ 10^{11/3}$

$\Rightarrow \frac{n}{10^{11/3}} \leq 1$

It should be $O(n\log n)$

$\log_2 8 = 3$

$8 = 2^3$

2

$\boxed{2}$ Time complexity $\cancel{O(\infty)}$ $O(n\infty) \simeq O(\infty)$

4

Scanned with CamScanner

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

$$= 4\left(4T\left(\frac{n}{3^2}\right) + \frac{n}{3}\right) + n$$

$$= 2^4 T\left(\frac{n}{3^2}\right) + \frac{2^2}{3} n + n$$

$$= 2^4\left(2^2 T\left(\frac{n}{3^3}\right) + \frac{n}{3^2}\right) + \frac{2^2}{3} n + n$$

$$\Rightarrow 2^6 T\left(\frac{n}{3^3}\right) + \frac{2^4}{3^2}(n) + \frac{2^2}{3} n + n$$

$$\Rightarrow 2^6\left(2^2 T\left(\frac{n}{3^4}\right) + \frac{n}{3^3}\right) + \frac{4^2}{3^2}(n) + \frac{4}{3} n + n$$

4

$$\Rightarrow 2^8 T\left(\frac{n}{3^4}\right) + \frac{4^3}{3^3} n + \frac{4^2}{3^2}(n) + \frac{4}{3} n + n$$

$$\Rightarrow 2^{2k} T\left(\frac{n}{3^k}\right) + n\left(\frac{4^{k-1}}{3^{k-1}} + \frac{4^{k-2}}{3^{k-2}} + \dots + \frac{4}{3} + 1\right)$$

$n = 3^k$

$\boxed{\log_3 n = k}$

$T\left(\frac{n}{n}\right) = 0$

$$\Rightarrow 0 + n\left(\frac{4^{k-1}}{3^{k-1}} + \frac{4^{k-2}}{3^{k-2}} + \dots + \frac{4}{3} + 1\right)$$

Geometric progression $\Rightarrow \frac{a(r^n - 1)}{r - 1} = \frac{1\left(\left(\frac{4}{3}\right)^{k-1} - 1\right)}{1/3}$

$$\Rightarrow 3\left[\left(\frac{4}{3}\right)^{\log_3 n}\left(\frac{3}{4}\right) - 1\right]$$

$$\Rightarrow 3\left[\frac{4^{\log_3 n}}{n} \times \frac{3}{4} - 1\right.$$

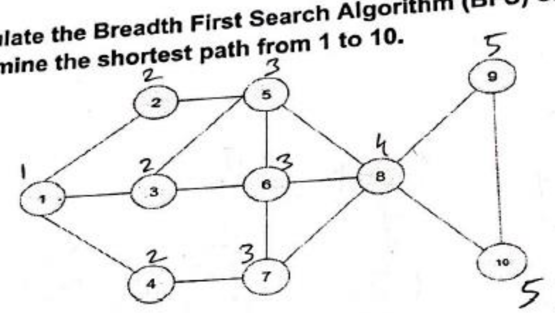
$$\Rightarrow 3\left[\frac{n^{\log_3 4}}{n} \times \frac{3}{4} - 1\right]$$

$$\Rightarrow 3\left[n^{\log_3 4 - 1} \times \frac{3}{4} - 1\right]$$

$$\Rightarrow O\left(n^{\log_3 4}\right) \text{ (Ans)}$$

Section: 14

1. **[Marks 4] Simulate the Breadth First Search Algorithm (BFS) on the following graph to determine the shortest path from 1 to 10.**

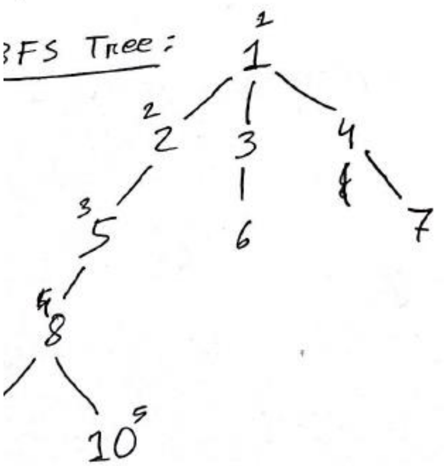


10

Source : 1

Output : 1 → 2 → 3 → 4 → 5 → 6 → 7 → 8 → 9 → 10

BFS Tree :



~~10~~
~~9~~
~~8~~
~~7~~
~~6~~
~~5~~
~~4~~
~~3~~
~~2~~
~~1~~

∴ shortage path is : 1 → 2 → 5 → 8 → 10 .

4

1. [Marks 2] Prove that, logn+n/3 = Ω(logn)
2. [Marks 4] Find the complexity:

```
k=0
for (i = 0 ; i <= n; i=i+1)
        for (j = 0; j <= n; j = j * 2)
                k = k + 1
```

3. [Marks 4] Solve using recursion tree method, $T(n) = 3T(n/2) + n$

Answer the question -02

for the first loop,

for (i=0, i<=n, i=i+1) , it a linear loop and it will run n time. So the time complexity for the first loop is O(n).

$\therefore$ for(i=0 ; i<=n , i=i+1) $\rightarrow$ O(n).

Now for the second loop

for (j=0 , j<=n , j = j×2) , here

it is a infinite loop because the initial value of j is 0. So when we multiply the value of j will not increase and it will never reach to n.

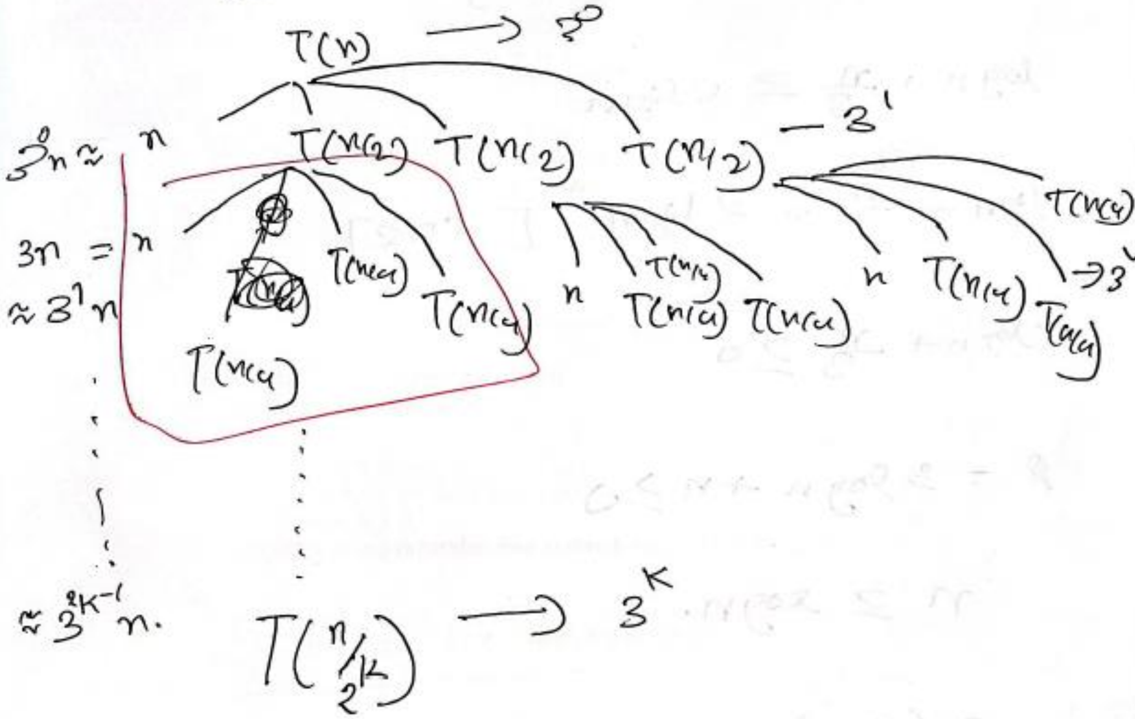
∴ So the total time complexity for the programme is $\boxed{O(\infty)}$. Because the second loop is a infinite loop.

4

Page-1

Answer to the question -03

$$T(n) = 3T(n/2) + n$$



Now, $\frac{n}{2^k} = 1$, $k = \log_2 n$.

4

$\therefore\ n + K \cdot (3^K)$

$= n + \log_2 n \cdot \left(3^{\log_2 n}\right)$

$= n + \log_2 n \cdot \boxed{3^{\log_2 n}}$

Answer to the Question - 01

$\log n + n/3 = \Omega(\log n)$

$\log n + \frac{n}{3} \geq c \log n$

$\log n + \frac{n}{3} = 2 \log n \quad [c = 2]$

$-\log n + \frac{n}{3} \geq 0$

$-3 \log n + n \geq 0$

$n \geq \log n.$

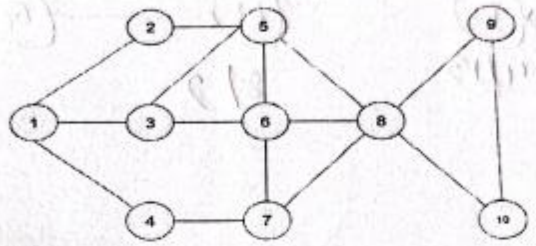
$\therefore \Omega(\log n)$ is the lower bound

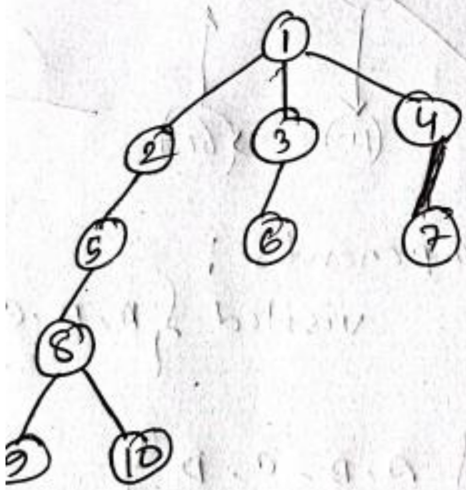
of $\log n + n/3$.

2

1. **[Marks 4] Simulate the Breadth First Search Algorithm (BFS) on the following graph to determine the shortest path from 1 to 10.**

10





Shortest path from 1 to 10 is

$\boxed{1 - 2 - 5 - 8 - 10}$

4

~~2~~	~~3~~	~~4~~	~~5~~	~~6~~	~~7~~	~~8~~	~~9~~	~~10~~

Queue