

[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) $n \log n$

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

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

$$n^{\log_2 5}$$

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) $O(n^{\log_2 5})$

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

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

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

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

$$n^3 + 5n^2 \leq c n^3 \quad \text{and} \quad n^3 + 5n^2 \geq c_1 n^3$$

$$\text{Let } c_1 = 1 \quad \Rightarrow \text{Let } c = 1.$$

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

$$n^3 + 5n^2 \leq n^3 \quad \text{Let } c = 2$$

for $n_0 = 0 \checkmark$

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

$$5n^2 \leq n^3$$

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

for $n_0 = 5 \checkmark$

Since, we can prove

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

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

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

8) Find the runtime of the following recurrence relation:

6

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

$$T(1) = 1$$

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

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

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

$$T(n-2) = T(n-3) + (n-2) \checkmark$$

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

$$\vdots$$

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

Assume, $n-k = 1$

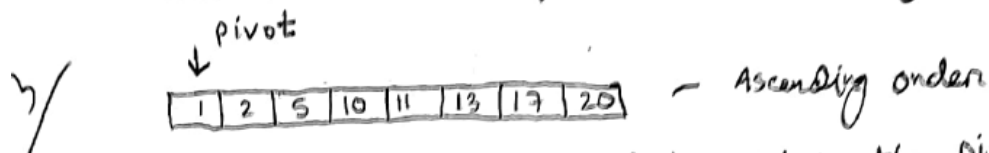
6) $T(1) + T(2) + T(3) + \dots + (n-2) + (n-1) + n$

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

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

- 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 $O(n^2)$ ✓

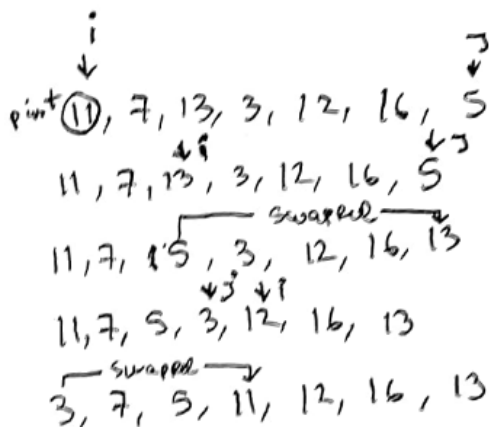
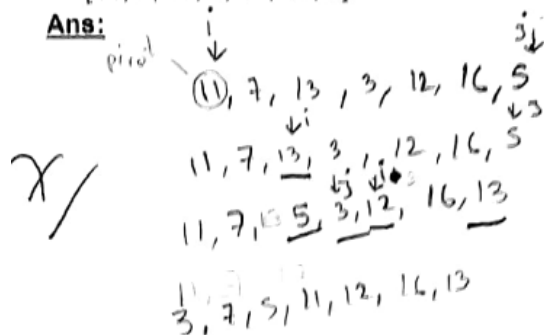


To avoid the problem, we need to pick the pivot at middle of array or at random point index

- 2) **Partition** the given array taking the **first element** as the **pivot**. Show the necessary steps by **simulation**. [7]

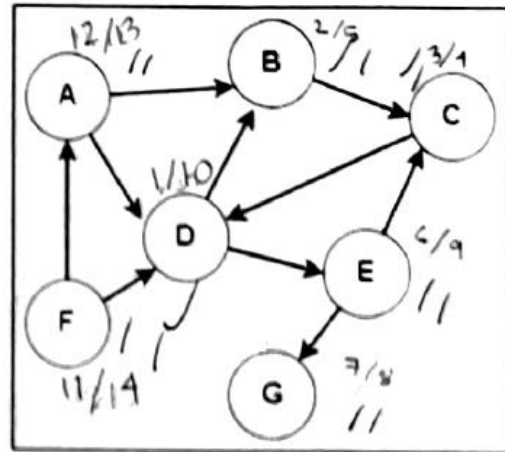
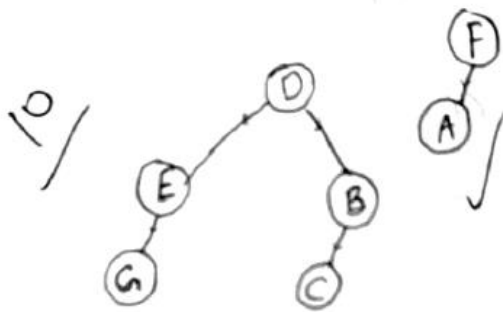
[11, 7, 13, 3, 12, 16, 5].

Ans:

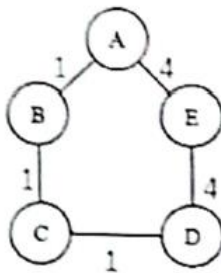


- 3) Taking node D as the **source**, use DFS to find the **discovery** and **finish times** of all the nodes in the following graph. Also draw the **DFS tree**. [7+3]

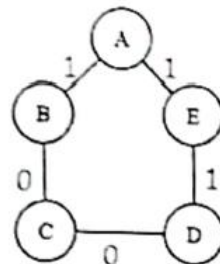
Node	A	B	C	D	E	F	G
Discovery time	12	2	3	1	6	11	7
Finish time	13	5	4	10	9	14	8



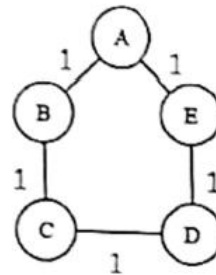
Question 1: [Points $0.25 \times 4 = 1$]



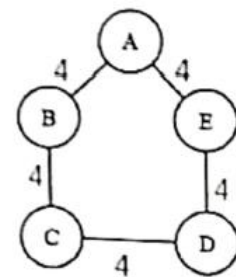
Graph 1



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)
    }
}

```

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.

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

for tight bound:

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

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

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

$$c_1 = 1, n_0 = 1$$

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

$$1 \leq 5(1)^3 + 7$$

$$1 \leq 12$$

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

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

$$c_2 = 7n^4 + 5n^3 \log_2 n$$

$$= 12$$

$$\therefore c_2 = 12, n_0 = 1$$

$$\therefore \text{Yes, } 5n^3 \log_2 n + 7n^4 = O(n^4)$$

$$\text{if } c_1 = 1, c_2 = 12, n_0 = 1$$

$$\begin{array}{r|l}
 2 = 152 & 102 \\
 3 = 781 & 972 \\
 4 = 2432 & 3672
 \end{array}$$

8

⑪ $7n^5 + 35n^2 + 10 = O(n^{10})$

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

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

$c_1 = 1, n_0 = 2$

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

~~$275 \leq 1024$~~

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

12

$c_1 = 7n^5 + 35n^5 + 10n^5$
 $= 7 + 35 + 10$
 $= 52$

$c_1 = 52, n_0 = 1$

⑫ $\therefore O(1) + \{O(\log_5 n) \times O(n)\}$

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

$= O(n \log_5 n)$

10

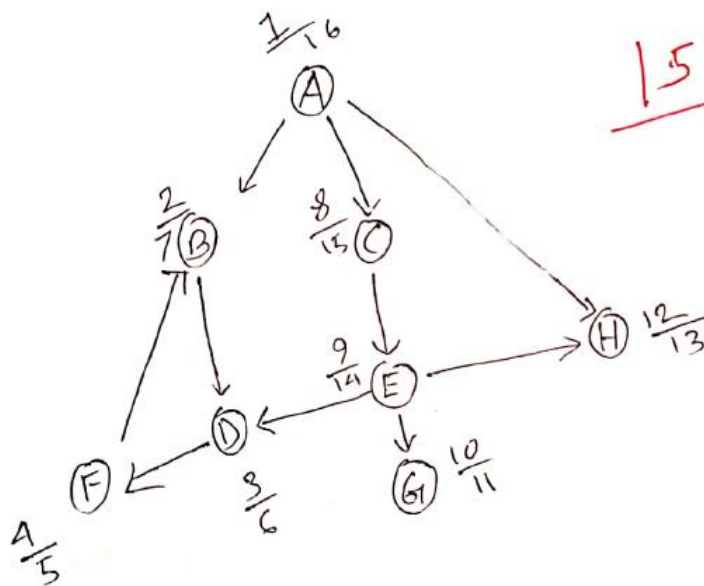
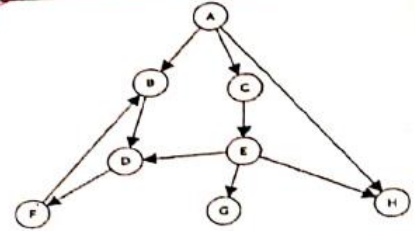
1. Suppose you have a array, $A = [14, 11, 10, 8, 5, 2]$. To Sort this array will you prefer Merge Sort or Quick Sort? Write 1 line defending your answer. (6)

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

b

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	<u>2</u>	8	3	9 14	4	10	12
Finish time	16	7	15	6	14 11	5	11	13



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 \exists and \forall .

$\Omega(f(n))$: for all there exists a positive c and n_0 for which $0 \leq f(n) \leq c g(n)$ for $\forall n \geq n_0$.

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 $t_j = 1$,

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

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

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

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

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

```
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
        }
    }
}
```

Step	i	j
1	1	0
2	2	1
3	4	3
4	8	7
5	16	15
6	32	31
7	64	63
8	128	127
9	256	255
10	512	511
11	1024	1023
12	2048	2047
13	4096	4095
14	8192	8191
15	16384	16383
16	32768	32767
17	65536	65535
18	131072	131071
19	262144	262143
20	524288	524287
21	1048576	1048575
22	2097152	2097151
23	4194304	4194303
24	8388608	8388607
25	16777216	16777215
26	33554432	33554431
27	67108864	67108863
28	134217728	134217727
29	268435456	268435455
30	536870912	536870911
31	1073741824	1073741823
32	2147483648	2147483647
33	4294967296	4294967295
34	8589934592	8589934591
35	17179869184	17179869183
36	34359738368	34359738367
37	68719476736	68719476735
38	137438953472	137438953471
39	274877906944	274877906943
40	549755813888	549755813887
41	1099511627776	1099511627775
42	2199023255552	2199023255551
43	4398046511104	4398046511103
44	8796093022208	8796093022207
45	17592186044416	17592186044415
46	35184372088832	35184372088831
47	70368744177664	70368744177663
48	140737488355328	140737488355327
49	281474976710656	281474976710655
50	562949953421312	562949953421311
51	1125899906842624	1125899906842623
52	2251799813685248	2251799813685247
53	4503599627370496	4503599627370495
54	9007199254740992	9007199254740991
55	18014398509481984	18014398509481983
56	36028797018963968	36028797018963967
57	72057594037927936	72057594037927935
58	144115188075855872	144115188075855871
59	288230376151711744	288230376151711743
60	576460752303423488	576460752303423487
61	1152921504606846976	1152921504606846975
62	2305843009213693952	2305843009213693951
63	4611686018427387904	4611686018427387903
64	9223372036854775808	9223372036854775807
65	18446744073709551616	18446744073709551615
66	36893488147419103232	36893488147419103231
67	73786976294838206464	73786976294838206463
68	147573952589676412928	147573952589676412927
69	295147905179352825856	295147905179352825855
70	590295810358705651712	590295810358705651711
71	1180591620717411303424	1180591620717411303423
72	2361183241434822606848	2361183241434822606847
73	4722366482869645213696	4722366482869645213695
74	9444732965739290427392	9444732965739290427391
75	18889465931478580854784	18889465931478580854783
76	37778931862957161709568	37778931862957161709567
77	75557863725914323419136	75557863725914323419135
78	151115727451828646838272	151115727451828646838271
79	302231454903657293676544	302231454903657293676543
80	604462909807314587353088	604462909807314587353087
81	1208925819614629174706176	1208925819614629174706175
82	2417851639229258349412352	2417851639229258349412351
83	4835703278458516698824704	4835703278458516698824703
84	9671406556917033397649408	9671406556917033397649407
85	19342813113834066795298816	19342813113834066795298815
86	38685626227668133590597632	38685626227668133590597631
87	77371252455336267181195264	77371252455336267181195263
88	154742504910672534362390528	154742504910672534362390527
89	309485009821345068724781056	309485009821345068724781055
90	618970019642690137449562112	618970019642690137449562111
91	1237940039285380274899124224	1237940039285380274899124223
92	2475880078570760549798248448	2475880078570760549798248447
93	4951760157141521099596496896	4951760157141521099596496895
94	9903520314283042199192993792	9903520314283042199192993791
95	19807040628566084398385987584	19807040628566084398385987583
96	39614081257132168796771975168	39614081257132168796771975167
97	79228162514264337593543950336	79228162514264337593543950335
98	158456325028528675187087900672	158456325028528675187087900671
99	316912650057057350374175801344	316912650057057350374175801343
100	633825300114114700748351602688	633825300114114700748351602687
101	1267650600228229401496703205376	1267650600228229401496703205375
102	2535301200456458802993406410752	2535301200456458802993406410751
103	5070602400912917605986812821504	5070602400912917605986812821503
104	10141204801825835211973625643008	10141204801825835211973625643007
105	20282409603651670423947251286016	20282409603651670423947251286015
106	40564819207303340847894502572032	40564819207303340847894502572031
107	81129638414606681695789005144064	81129638414606681695789005144063
108	162259276829213363391578010288128	162259276829213363391578010288127
109	324518553658426726783156020576256	324518553658426726783156020576255
110	649037107316853453566312041152512	649037107316853453566312041152511
111	1298074214633706907132624082305024	1298074214633706907132624082305023
112	2596148429267413814265248164610048	2596148429267413814265248164610047
113	5192296858534827628530496329220096	5192296858534827628530496329220095
114	10384593717069655257060992658440192	10384593717069655257060992658440191
115	20769187434139310514121985316880384	20769187434139310514121985316880383
116	41538374868278621028243970633760768	41538374868278621028243970633760767
117	83076749736557242056487941267521536	83076749736557242056487941267521535
118	166153499473114484112975882535043072	166153499473114484112975882535043071
119	332306998946228968225951765070086144	332306998946228968225951765070086143
120	664613997892457936451903530140172288	664613997892457936451903530140172287
121	1329227995784915872903807060280344576	1329227995784915872903807060280344575
122	2658455991569831745807614120560689152	2658455991569831745807614120560689151
123	5316911983139663491615228241121378304	5316911983139663491615228241121378303
124	10633823966279326983230456482242756608	10633823966279326983230456482242756607
125	21267647932558653966460912964485513216	21267647932558653966460912964485513215
126	42535295865117307932921825928971026432	42535295865117307932921825928971026431
127	85070591730234615865843651857942052864	85070591730234615865843651857942052863
128	170141183460469231731687303715884105728	170141183460469231731687303715884105727
129	340282366920938463463374607431768211456	340282366920938463463374607431768211455
130	680564733841876926926749214863536422912	680564733841876926926749214863536422911
131	1361129467683753853853498429727072845824	1361129467683753853853498429727072845823
132	2722258935367507707706996859454145691648	2722258935367507707706996859454145691647
133	5444517870735015415413993718908291383296	5444517870735015415413993718908291383295
134	10889035741470030830827987437816582766592	10889035741470030830827987437816582766591
135	21778071482940061661655974875633165533184	21778071482940061661655974875633165533183
136	43556142965880123323311949751266331066368	43556142965880123323311949751266331066367
137	87112285931760246646623899502532662132736	87112285931760246646623899502532662132735
138	174224571863520493293247799005065324265472	174224571863520493293247799005065324265471
139	348449143727040986586495598010130648530944	348449143727040986586495598010130648530943
140	696898287454081973172991196020261297061888	696898287454081973172991196020261297061887
141	1393796574908163946345982392040522594123776	1393796574908163946345982392040522594123775
142	2787593149816327892691964784081045188247552	2787593149816327892691964784081045188247551
143	5575186299632655785383929568162090376495104	5575186299632655785383929568162090376495103
144	11150372599265311570767859136324180752990208	11150372599265311570767859136324180752990207
145	22300745198530623141535718272648361505980416	22300745198530623141535718272648361505980415
146	44601490397061246283071436545296723011960832	44601490397061246283071436545296723011960831
147	89202980794122492566142873090593446023921664	89202980794122492566142873090593446023921663
148	178405961588244985132285746181186892047843328	178405961588244985132285746181186892047843327
149	356811923176489970264571492362373784095686656	356811923176489970264571492362373784095686655
150	713623846352979940529142984724747568191373312	713623846352979940529142984724747568191373311
151	1427247692705959881058285969449495136382746624	1427247692705959881058285969449495136382746623
152	2854495385411919762116571938898990272765493248	2854495385411919762116571938898990272765493247
153	5708990770823839524233143877797980545530986496	5708990770823839524233143877797980545530986495
154	11417981541647679048466287755595961091061972992	11417981541647679048466287755595961091061972991
155	22835963083295358096932575511191922182123945984	22835963083295358096932575511191922182123945983
156	45671926166590716193865151022383844364247891968	45671926166590716193865151022383844364247891967
157	91343852333181432387730302044767688728495783936	91343852333181432387730302044767688728495783935
158	182687704666362864775460604089535377456991567872	182687704666362864775460604089535377456991567871
159	365375409332725729550921208179070754913983135744	365375409332725729550921208179070754913983135743
160	730750818665451459101842416358141509827966271488	730750818665451459101842416358141509827966271487
161	1461501637330902918203684832716283019655932542976	1461501637330902918203684832716283019655932542975
162	2923003274661805836407369665432566039311865085952	2923003274661805836407369665432566039311865085951
163	5846006549323611672814739330865132078623730171904	5846006549323611672814739330865132078623730171903
164	11692013098647223345629478661730264157247460343808	11692013098647223345629478661730264157247460343807
165	23384026197294446691258957323460528314494920687616	233840261972944466912589573

Question 1: [5 points]

Explain the time complexity of the following code snippet?

```
j = 0
for (i = 0; i < n; i++):  $\rightarrow n$ 
    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 and value of j will never increase.

\therefore Only outer loop will run n times.

$\therefore O(n)$

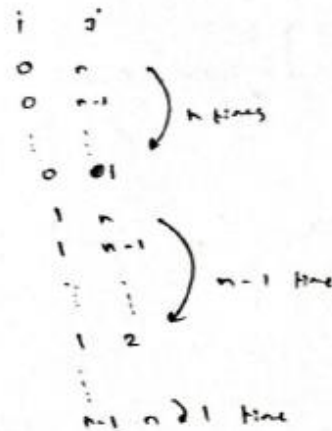
Question 2: [5 points]

Explain the time complexity of the following code snippet?

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

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

$$= O(n^2)$$



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

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

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

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

$$= O(n^2)$$

Question 3: [5 Points]

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

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

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

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

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

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

$$\Rightarrow n > 2$$

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

for $n_0 = 2$

Proved

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

Let $c_2 = 1$

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

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

$$\Rightarrow n > -1$$

$$\Rightarrow n \geq 0$$

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

for $n_0 = 0$

Proved.

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

Question 4: [5 points]

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

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

$$\begin{aligned} O(n) &= O(\log n) \\ &= O(n \log n) \end{aligned}$$

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

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

$$\text{when } a=1, b=1, f(n) = \log n$$

$$\text{If } a=1$$

$$O(n + f(n))$$

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

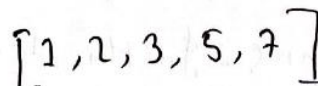
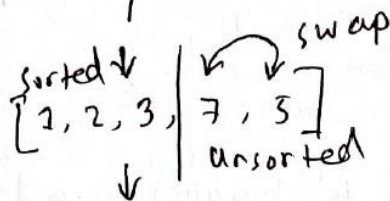
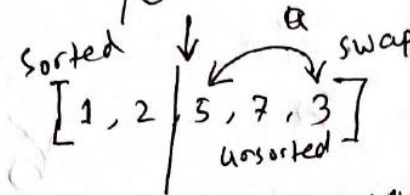
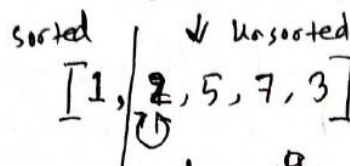
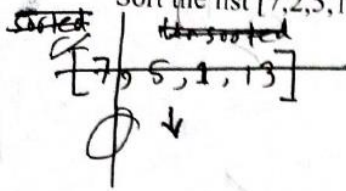
Question 1: [5 points]

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

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

Question 2: [5 points]

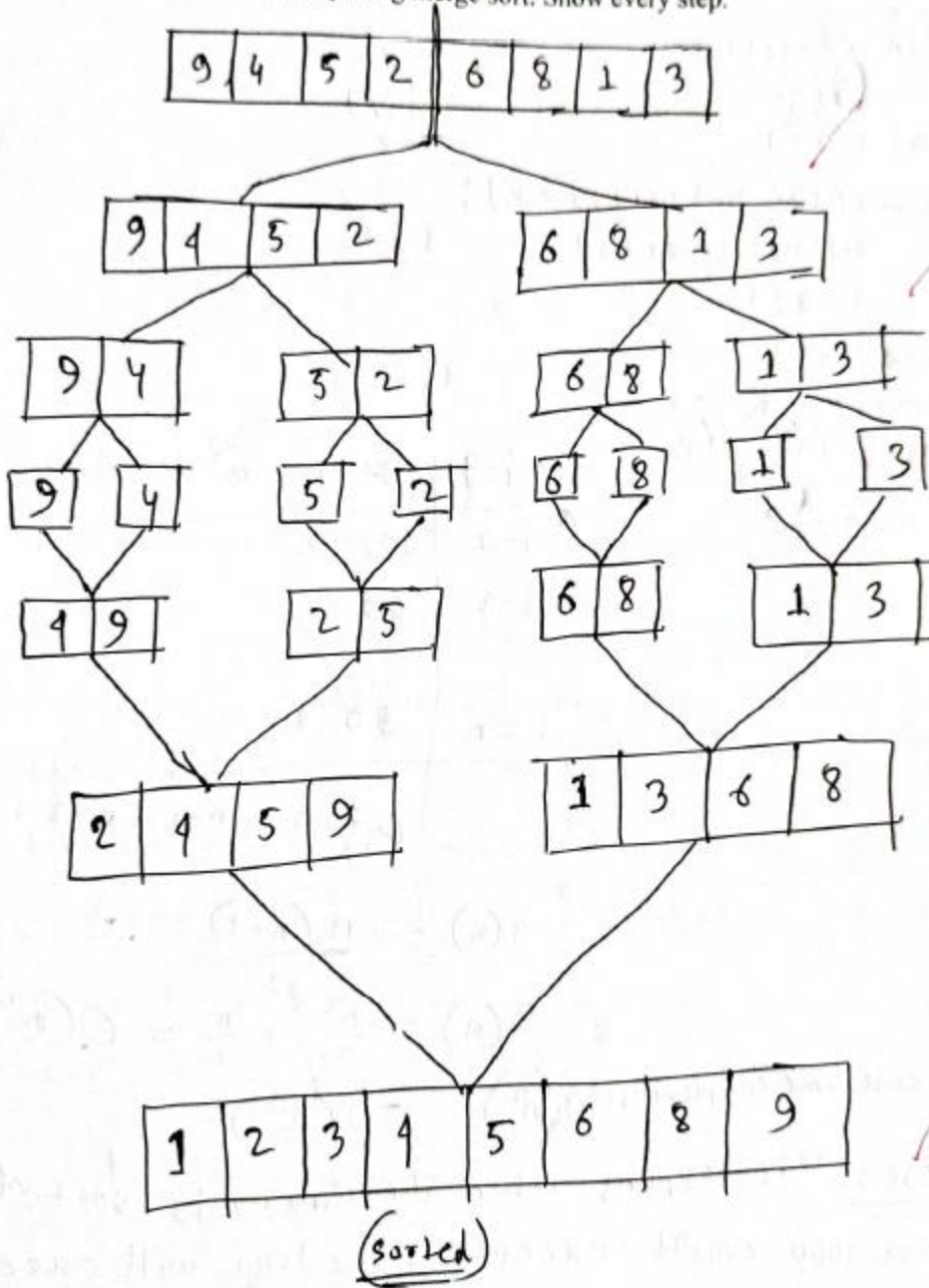
Sort the list [7, 2, 5, 1, 3] using selection sort. Show every step.



(Sorted)

Question 3: [5 Points]

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



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] = t
```

worst case:

i	iterations
i=1	1
i=2	2
i=3	3
...	...
i=n	n
$1 + 2 + 3 + \dots + n = \frac{n(n+1)}{2}$	

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

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

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

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
<p>Pseudo code:</p> <pre> (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") </pre>	<p>Pseudo code:</p> <pre> (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}") </pre>

Answer:

(a)

~~sum~~
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=k-1) $O(n)$

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

\therefore 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

A	B	Big - Oh / Big - Omega / Big - Theta ($O / \Omega / \Theta$)
$\frac{1}{2}n$	$n^{\sin(n)}$	$A = O(B)$
$e^{\ln(n)}$	n^2	$A = O(B)$ $A = O(B)$ ✓
$n!$	$n + 1^n$	$A = \Omega(B)$ ✓
$5n \log(n)$	$2 \ln(e^n)$	$A = \Omega(B)$ ✓

- 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)

indexes \rightarrow $\overset{0}{\cancel{14}}, \overset{1}{\cancel{12}}, \overset{2}{23}, \overset{3}{29}, \overset{4}{34}, \overset{5}{37}, \overset{6}{41}, \overset{7}{42}, \overset{8}{46}, \overset{9}{\cancel{49}}, \overset{10}{52}, \overset{11}{62}$ 5

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

Ans:

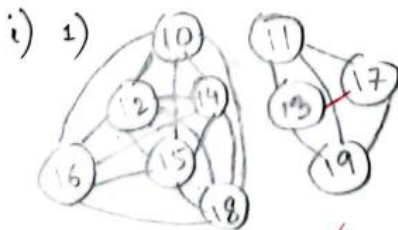
(i) There is an undirected simple graph of 10 nodes. Nodes are labeled from 10 to 19. The edges are:

- All the nodes that are labeled with a prime number have edges among them. — 3
 - All the nodes that are labeled with a composite number have edges among them. — 1
- Draw the graph. — 3
 - 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
 - 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
 - Jack wants to find the number of cycles in this graph using DFS traversal. Suggest him a solution for this task. — 3

(ii)

- 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
 - Insertion Sort
 - Selection Sort
- After doing partition in the first step of Quick Sort algorithm the array looks like this : 19, 17, 15, 23, 27, 39, 32, 37
 - What was the pivot before doing the partition? — 1
 - 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:



2.5

i) 2) $6 \times 4 = 24$ 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) Use DFS traversal from any starting vertex. As it traverses through the edges and reaches its starting vertex, and the process; then $\text{cycles} += 1$. Repeat this process with other vertices until all the vertices are visited. Ultimately the value of 'cycles' provides the number of cycles.

25

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

2

④

$$\text{ii) }$$

b)

၆၄

2

12.

First partition done

Determine the worst case time complexity for the following code,

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

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

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. $f(n) = \Theta(g(n))$ or
- III. $f(n) = \Omega(g(n))$

Question 3: CO2 [8 Points]

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

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

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

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

$$\log 100 = 2$$

$$\Rightarrow 100 \leq c \log 100$$

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

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

4/4

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

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

$$\Rightarrow 100 \geq c \log 100$$

$$\Rightarrow 100 \geq c \times 2 \quad c \leq 1$$

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

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

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

1) $\frac{7}{8}$

	x	y	No. of times
1	1	1	1
2	2	2x	2
3	3	3x	3
...
n	n	nx	n

1st loop

terminate
 $y > x$

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

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

$$= n^2$$

$$O(n^2)$$

Second loop

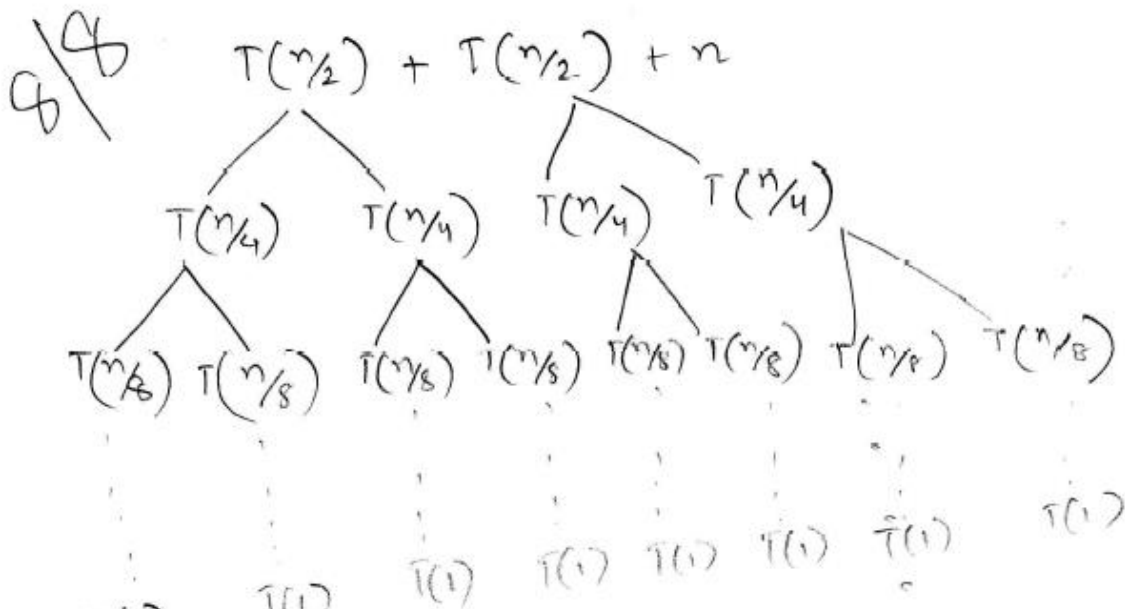
$$f(n) = 2n + 1$$

$$O(n)$$

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

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

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



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

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

$$\Rightarrow n = 2^k$$

$$\Rightarrow k = \log_2 n$$

$$\therefore O(n \log n)$$

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

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

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	

$j > \text{pivot}$
 $j = j - 1$
 $i < \text{pivot}$
 $i = i + 1$

8	12	15	7	10	5	4	13
i				p		j	

$p \rightarrow \text{pivot}$

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 $j < \text{pivot}$ and $i > \text{pivot}$,
 $\text{swap}(\text{arr}[i], \text{arr}[j])$

8	4	5	7	10	15	12	13
i				p		j	

8	4	5	7	10	15	12	13

\Downarrow
 all the elements
 less than pivot

\Downarrow
 all the elements
 greater than
 pivot

→ This is the partition part

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

\ When the pivot is the min or max element

II Worst case:

$$O(n^2)$$

Best case:

$$\Omega(n \log n)$$

3

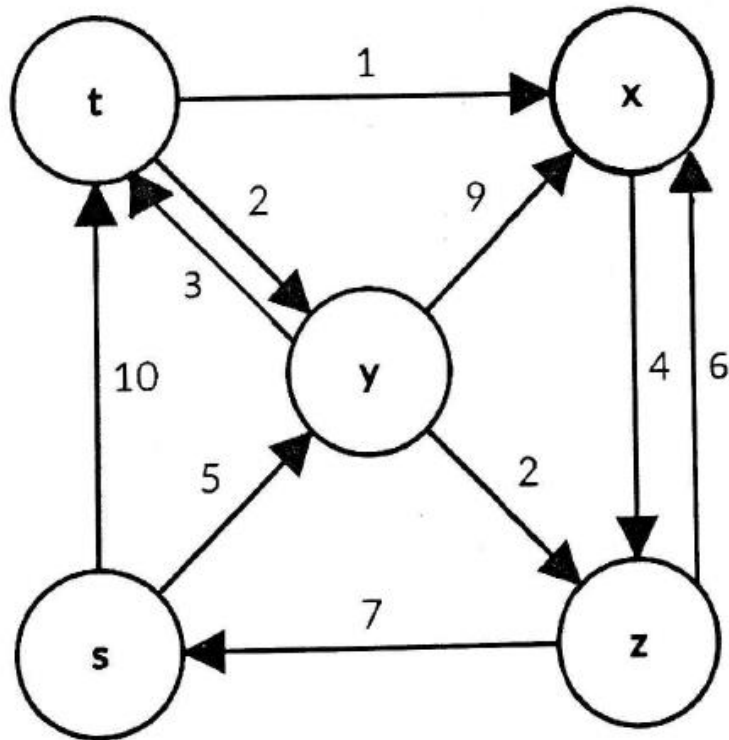
Average case:

$$\Theta(n \log n)$$

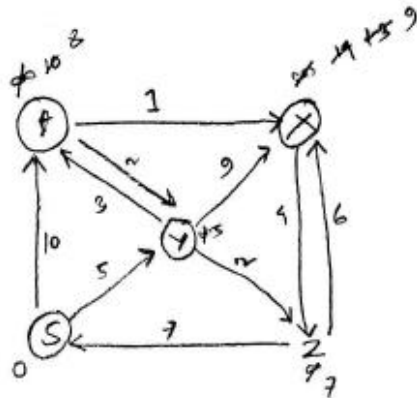
III Yes, it is an in-place algorithm so we don't
need ~~an~~ new array. in Quick sort.

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	S
X	25	Y
Y	5	S
Z	7	Y

Shortest Path:

S to Z:

$S \rightarrow Y \rightarrow Z = 7$

S to Y:

$S \rightarrow Y = 5$

S to t:

$S \rightarrow Y \rightarrow t = 8$

S to X:

$S \rightarrow Y \rightarrow t \rightarrow X = 9$

8/8

1. [Marks 2] Prove that, $n \log n + n/3 = O(n)$

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) }  $O(\infty)$   
        k = k + 1
```

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

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

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

$$\Rightarrow n \leq 10^4 \cdot 10^{1/3}$$

~~2~~

$$\Rightarrow \frac{n}{10^{41/3}} \leq 1$$

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

$$\log_2 8 = 3$$

$$8 = 2^3$$

$\boxed{2}$ Time complexity ~~$O(\infty)$~~ $O(n \log n) \approx O(\infty)$

~~4~~

$$\begin{aligned}
 \boxed{3} \quad T(n) &= 4T\left(\frac{n}{3}\right) + n \\
 &= 4\left(4T\left(\frac{n}{3^2}\right) + \frac{n}{3}\right) + n \\
 &= 2^4 T\left(\frac{n}{3^2}\right) + \frac{4^2}{3}n + n \\
 &= 2^4 \left(2^2 T\left(\frac{n}{3^3}\right) + \frac{n}{3^2}\right) + \frac{2^2}{3}n + n \\
 &\Rightarrow 2^6 T\left(\frac{n}{3^3}\right) + \frac{2^4}{3^2}n + \frac{2^2}{3}n + n \\
 &\Rightarrow 2^6 \left(2^2 T\left(\frac{n}{3^4}\right) + \frac{n}{3^3}\right) + \frac{4^2}{3^2}n + \frac{4}{3}n + n \\
 &\Rightarrow 2^8 T\left(\frac{n}{3^4}\right) + \frac{4^3}{3^3}n + \frac{4^2}{3^2}n + \frac{4}{3}n + n \\
 &\Rightarrow 2^k T\left(\frac{n}{3^k}\right) + n \left(\frac{4^{k-1}}{3^{k-1}} + \frac{4^{k-2}}{3^{k-2}} + \dots + \frac{4}{3} + 1 \right)
 \end{aligned}$$

$$\begin{aligned}
 n &= 3^k \\
 \log_3 n &= k \\
 T\left(\frac{n}{n}\right) &= 0
 \end{aligned}$$

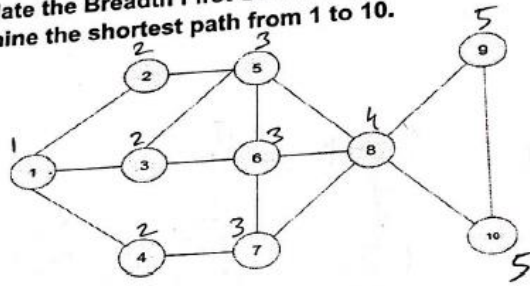
$$\Rightarrow 0 + n \left(\frac{4^{k-1}}{3^{k-1}} + \frac{4^{k-2}}{3^{k-2}} + \dots + \frac{4}{3} + 1 \right)$$

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

$$\begin{aligned}
 &\Rightarrow 3 \left[\left(\frac{4}{3} \right)^{\log_3 n} \left(\frac{3}{4} \right) - 1 \right] \\
 &\Rightarrow 3 \left[\frac{4^{\log_3 n}}{n} \times \frac{3}{4} - 1 \right] \\
 &\Rightarrow 3 \left[\frac{n^{\log_3 4}}{n} \times \frac{3}{4} - 1 \right] \\
 &\Rightarrow 3 \left[n^{\log_3 4 - 1} \times \frac{3}{4} - 1 \right] \\
 &\Rightarrow O(n^{\log_3 4}) \quad \text{Ans}
 \end{aligned}$$

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.

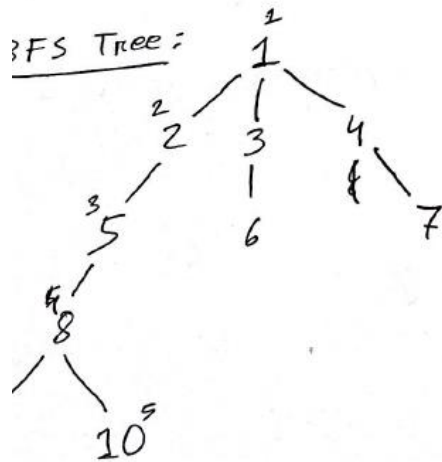


10

source : 1

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

BFS Tree :



10
9
8
7
6
5
4
3
2
1

∴ shortest path is : 1 → 2 → 5 → 8 → 10

4

AS

1. [Marks 2] Prove that, $\log n + n/3 = \Omega(\log n)$

2. [Marks 4] Find the complexity:

$k=0$

for ($i = 0$; $i \leq n$; $i = i + 1$)

for ($j = 0$; $j \leq n$; $j = j * 2$)

$k = k + 1$

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

Answer the question -02

for the first loop,

for ($i = 0$, $i \leq n$, $i = i + 1$) , it is a linear

loop and it will run n times. So the time complexity for the first loop is $O(n)$.

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

Now for the second loop

for ($j=0$, $j \leq n$, $j = j \times 2$) here
it is a infinite loop because ~~the~~ the
initial value of j is 0. ~~So~~ So when
we multiply ~~the~~ the value of j will not
increase and it will never reach to n .

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

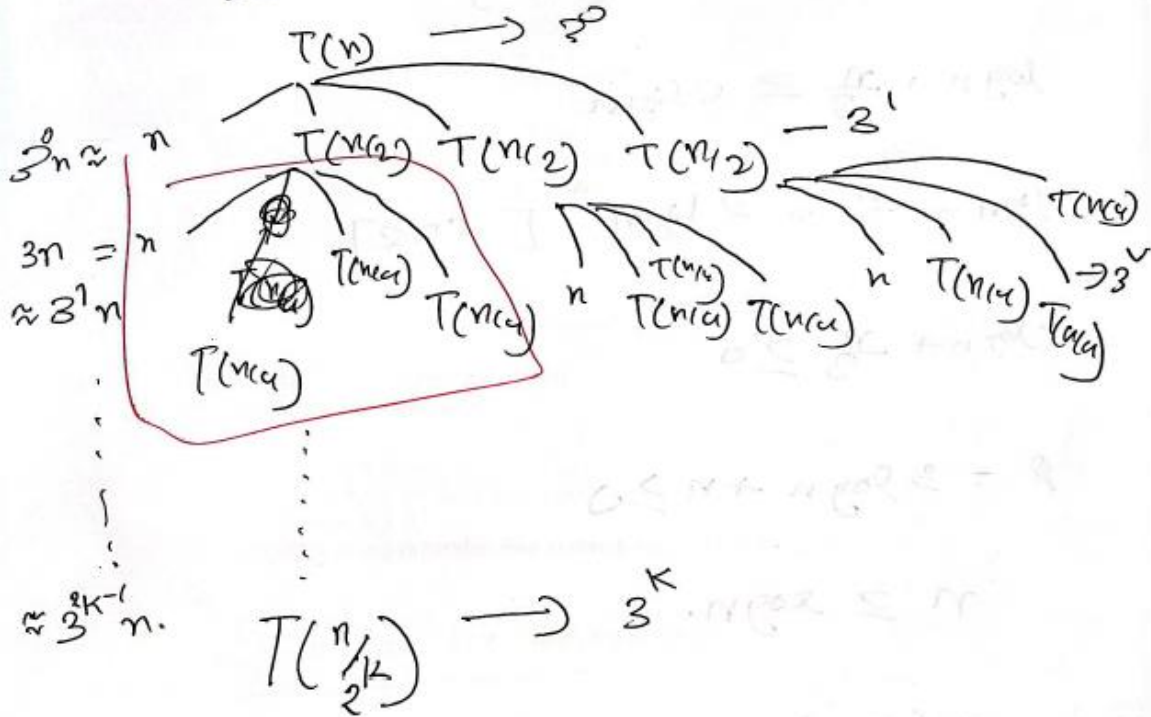
✓

✓

Ans 1

Answer to the question - 03

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



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

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

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

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

4

Answer to the question - 01

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

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

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

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

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

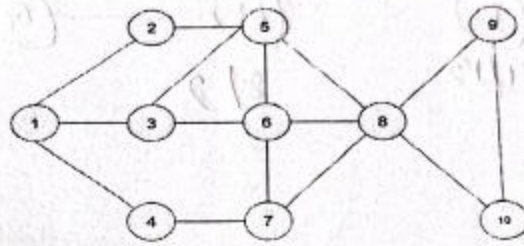
$$n \geq 3 \log n.$$

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

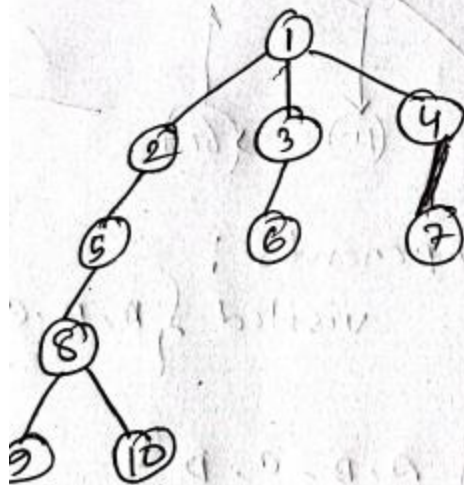
of $\log n + \frac{n}{3}$.

✓

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



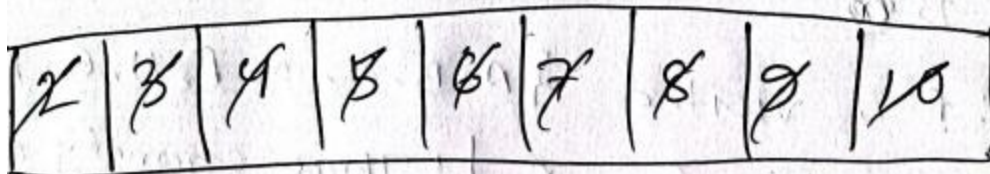
10



Shortest path from 1 to 10 is

1 - 2 - 5 - 8 - 10

4



Queue