

Type: MCQ

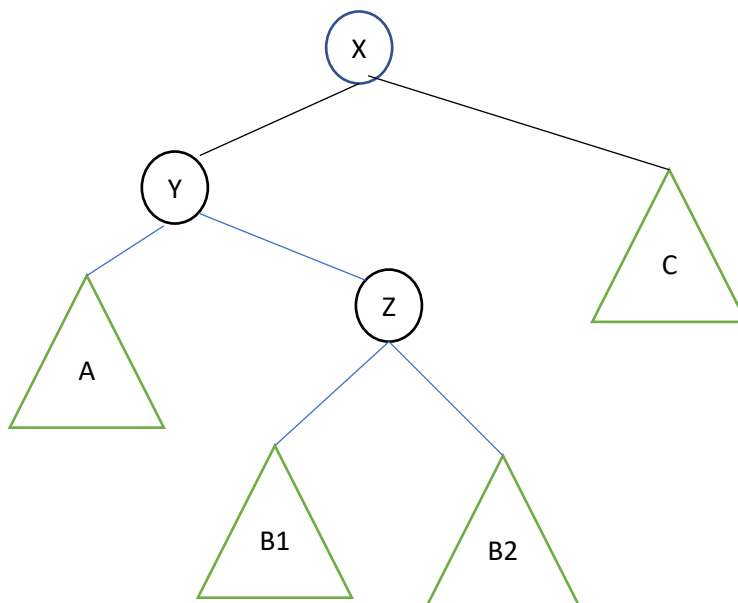
Q1. Consider the following array of elements 289,219,250,217,212,215,22,25,27,211,26,29, 300. What is the number of interchanges required to convert it into a max heap? (0.5)

1. 5
2. 2
3. 4
4. **3

Q2. Consider the array $A = \{4, 1, 3, 2, 16, 9, 10, 14, 8, 7\}$, After building max-heap from the array A, the depth of the heap and right child of root are ----- and ----- respectively. (0.5)

1. **3,10
2. 3,14
3. 4,14
4. 4,10

Q3. When the balance factor of X is 2 due to insertion of nodes at either B1 or B2, a rotation would result to (0.5)



1. B2 is the left subtree of Y
2. **B2 is the left subtree of right child of Z
3. B1 is the left subtree of right child of Z
4. B2 is the right subtree of left child of Z

Q4. How does a 2-3 tree balance when the root node has to be deleted? (0.5)

1. Swap of the elements in the subtrees of the root
2. Demotion (movement from higher levels to lower levels) of the nodes
3. **Promotion of nodes from lower levels
4. Deletion of one of the child nodes

Q5. Consider the following C program

```
int main() {
    int x, y, m, n;
    scanf ("%d %d", &x, &y); /* x > 0 and y > 0 */
    m = x; n = y;
    while (m != n)
    {
        if(m>n)
            m = m - n;
        else
            n = n - m;
    }
    printf("%d", n); }
```

What does the program compute? (0.5)

1. $x + y$ using repeated subtraction
2. $x \bmod y$ using repeated subtraction
3. **The greatest common divisor of x and y
4. The least common multiple of x and y

Q6. Consider the following pairs of functions $f(n)$, $g(n)$. Which pair the functions are such, that $f(n)$ is $O(g(n))$ and $g(n)$ is not $O(f(n))$? (0.5)

- | | |
|-------------------|------------------------|
| 1. $f(n)=n^3$ | $g(n) = n^2 \log(n^2)$ |
| 2. $f(n)=\log(n)$ | $g(n) = 10 \log n$ |
| 3. ** $f(n)=1$ | $g(n) = \log n$ |
| 4. $f(n)=n^2$ | $g(n) = 10n \log n$ |

Q7. Consider the following array: $A[] = \{12, 11, 13, 5, 6\}$. Apply the insertion sort algorithm to sort the array. Assuming that the cost associated with each swap is 25 rupees, what will be the total cost of the insertion sort when all the elements are sorted? (0.5)

5. 200
6. **175
7. 50

8. 150

Q8. Which of the following analogies best represents the process of sorting the numbers (15, 3, 28, 10, 6, 19, 25) in ascending order using bubble sort? (0.5)

1. Sorting a deck of shuffled playing cards by comparing adjacent cards and swapping them if they are not in the correct order.
2. ******Arranging a list of books on a shelf by repeatedly comparing two adjacent books and swapping them if they are not in alphabetical order.
3. Organizing a collection of various-sized boxes on a shelf by comparing their sizes and rearranging them to ensure they are in ascending order.
4. Placing a set of differently colored marbles in a jar by comparing their colors and rearranging them to ensure they are in rainbow order.

Q9. Assume that a mergesort algorithm in the worst case takes 30 seconds for an input of size 64. Which of the following most closely approximates the maximum input size of a problem that can be solved in 6 minutes? (0.5)

1. 256
2. ******512
3. 1024
4. 2048

Q10. If one uses a straight two-way merge sort algorithm to sort the following elements in ascending order: 20, 47, 15, 8, 9, 4, 40, 30, 12, 17 Then, the order of these elements after the second pass of the algorithm is (0.5)

1. ******8, 15, 20, 47, 4, 9, 30, 40, 12, 17
2. 8, 9, 15, 20, 47, 4, 12, 7, 30, 40
3. 15, 20, 47, 4, 8, 9, 12, 30, 40, 17
4. 4, 8, 9, 15, 20, 47, 12, 17, 30, 40

Type: DES

Q11. The airline's flight operations department receives a daily list of upcoming flight departure times for various destinations. The flight operations department receives the following unsorted list of flight departure times:

Flight 1: 04:00 AM to Paris

Flight 2: 10:00 AM to New York

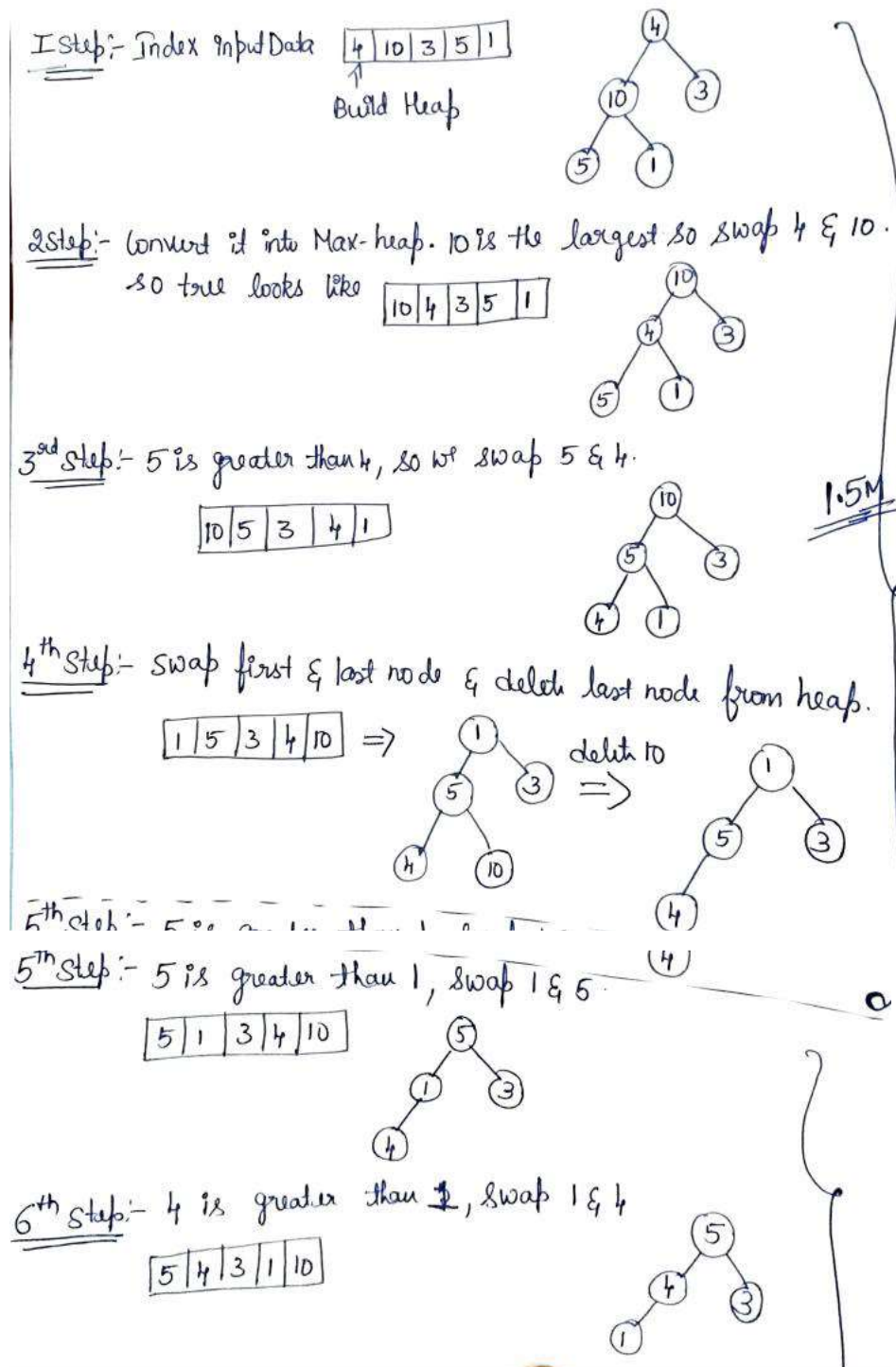
Flight 3: 03:00 AM to London

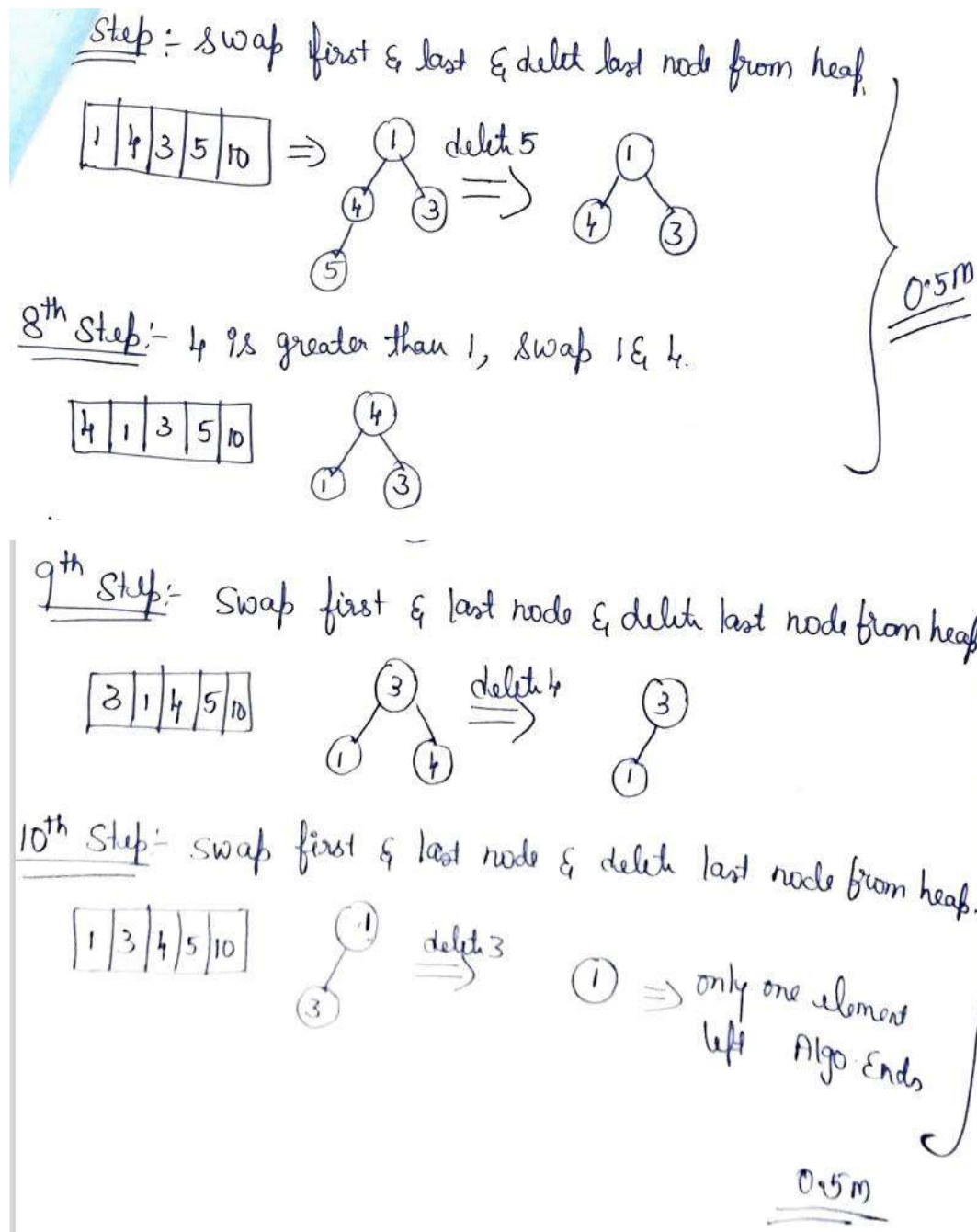
Flight 4: 05:00 AM to Tokyo

Flight 5: 1:00 AM to Dubai

To enhance customer satisfaction, use the Balanced tree structure to get the sorted Departure Schedule. Further Discuss the time complexity of your solution. (4)

Solution:





Since we already know that the heap construction stage of the algorithm is in $O(n)$, we have to investigate just the time efficiency of the second stage. For the number of key comparisons, $C(n)$, needed for eliminating the root keys from the heaps of diminishing sizes from n to 2, we get the following inequality:

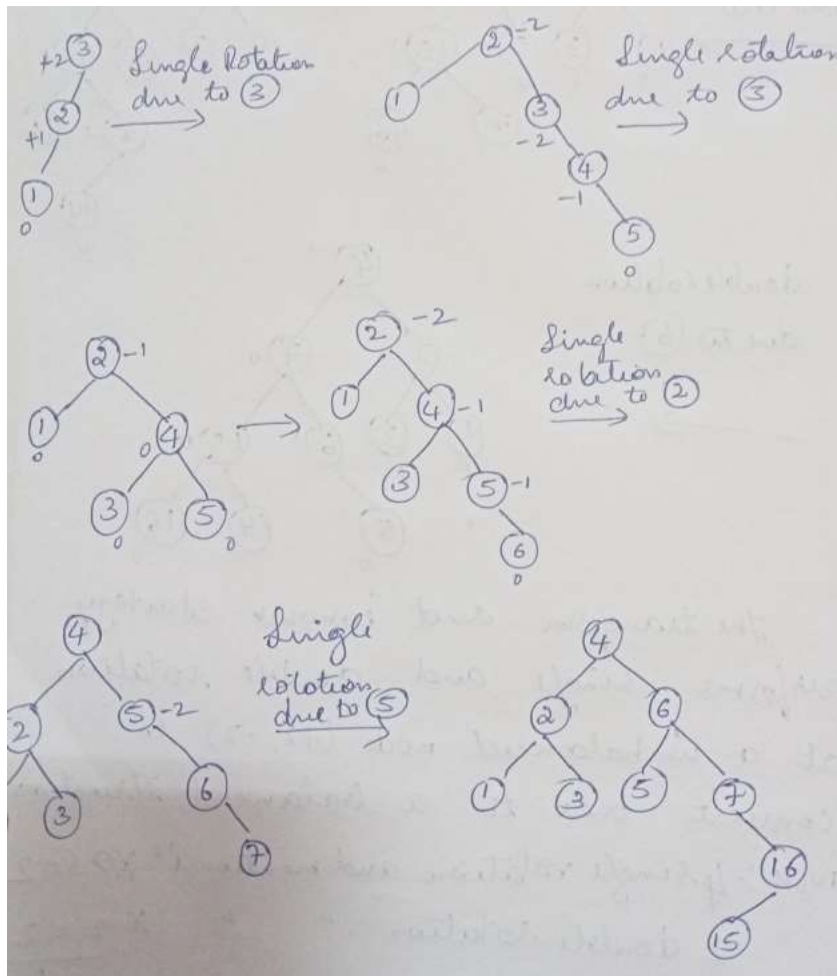
$$C(n) \leq 2\lfloor \log_2(n-1) \rfloor + 2\lfloor \log_2(n-2) \rfloor + \dots + 2\lfloor \log_2 1 \rfloor \leq 2 \sum_{i=1}^{n-1} \log_2 i$$

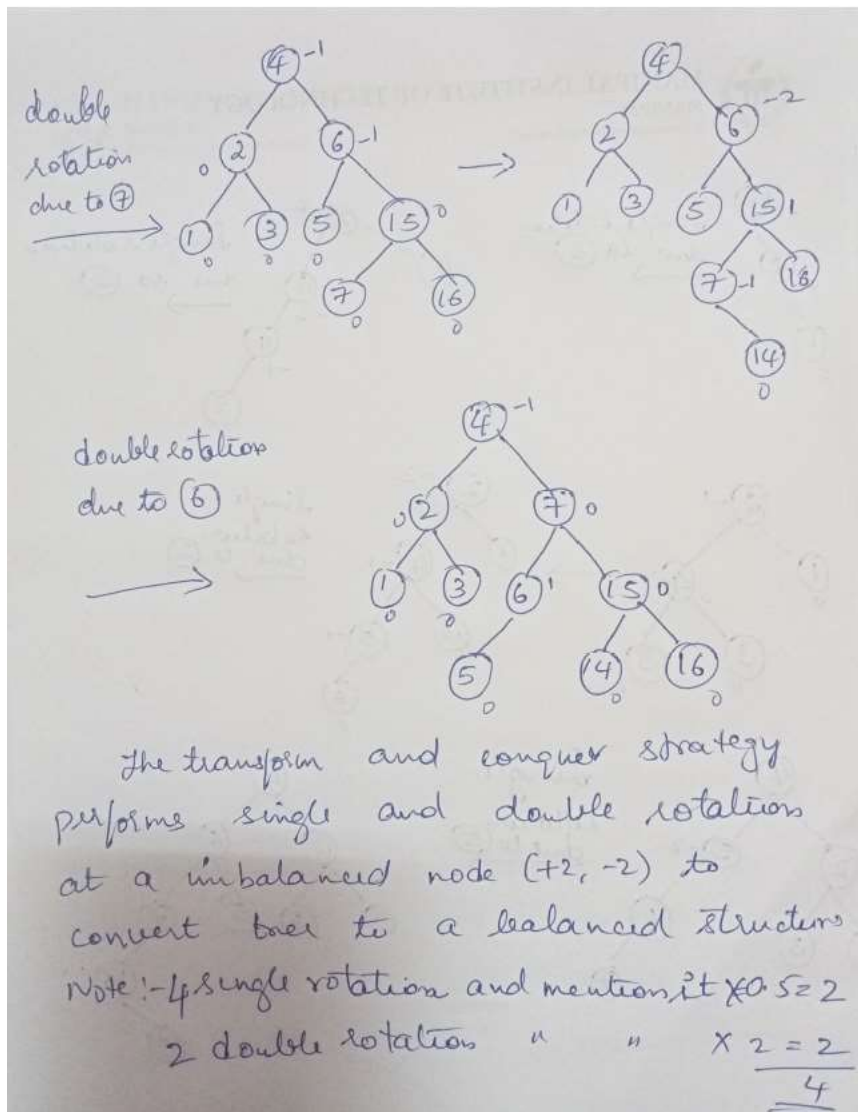
$$\leq 2 \sum_{i=1}^{n-1} \log_2(n-1) = 2(n-1) \log_2(n-1) \leq 2n \log_2 n.$$

This means that $C(n) \in O(n \log n)$ for the second stage of heapsort. For both stages, we get $O(n) + O(n \log n) = O(n \log n)$. A more detailed analysis shows that the time efficiency of heapsort is, in fact, in $(n \log n)$ in both the worst and average cases. **(1.5M)**

Q12. Justify how the transform and conquer strategy helps to maintain the balance factor of either 0, -1, or 1 at every node during the insertion of the following elements: 3, 2, 1, 4, 5, 6, 7, 16, 15, 14 into a binary tree. (4)

Solution:





Q13. Prove or Disprove using formal definitions of asymptotic notations and limit theory

a. $10n^2 + 9 = O(n)$ is false

b. $n^3 + 10^6n^2 = \theta(n^3)$ is true. (3)

Solution:

Prove or Disprove using formal definitions of asymptotic notations and limit theory.

a. $10n^2 + 9 = O(n)$ is false.

n^2 is always greater than n for any value of c and $n \geq 1$ (0.5M)

Applying Limits :

$$\lim_{n \rightarrow \infty} \frac{f(n)}{g(n)} = \lim_{n \rightarrow \infty} \frac{(10n^2 + 9)}{n} = \lim_{n \rightarrow \infty} 10n + \frac{9}{n} = \infty + 0 = \infty$$

Hence $10n^2 + 9 = \omega(n)$

Which proves that n is a lower bound not upper bound.

(1.0M)

b. $n^3 + 10^6n^2 = \theta(n^3)$ is true

n^3 is always greater than $n^3 + 10^6n^2$ for $c = 10^{13}$ and $n \geq 1$ and

n^3 is always smaller than $n^3 + 10^6n^2$ for $c = 1$ and $n \geq 1$

Hence $n^3 + 10^6n^2 = \theta(n^3)$

(0.5M)

Applying Limits :

$$\lim_{n \rightarrow \infty} \frac{f(n)}{g(n)} = \lim_{n \rightarrow \infty} \frac{(n^3 + 10^6n^2)}{n^3} = \lim_{n \rightarrow \infty} 1 + \frac{10^6}{n} = 1 + 0 = 1$$

Hence $n^3 + 10^6n^2 = \theta(n^3)$

(1.0M)

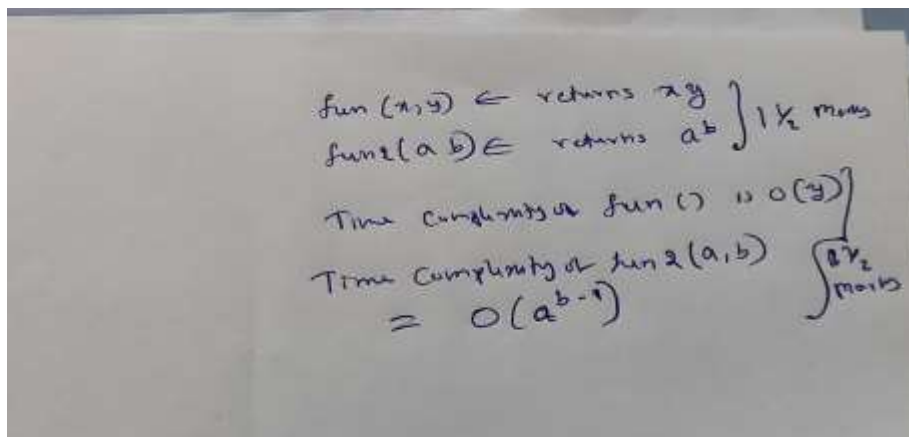
Q14. What does fun2() do in general?

```
int fun(int x, int y){
    if (y == 0) return 0;
    return (x + fun(x, y-1));
}
```

```
int fun2(int a, int b){
    if (b == 0) return 1;
    return fun(a, fun2(a, b-1));
}
```

Also find its time complexity. (3)

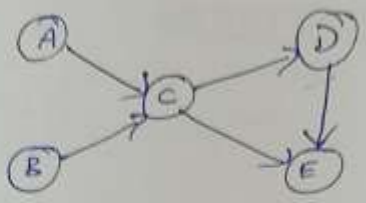
Solution:



Q15. Consider a project with five tasks, namely A, B, C, D, and E. Task C should start only after the completion of both Task A and Task B, while Task D should start only after the completion of Task C. Finally, Task E should start only after the completion of both Task C and Task D. create a digraph representation of the scenario and apply topological sorting order to solve the problem. (3)

Solution:


Q14



0.5 marks.

Deletion Method. or if you are using DFS method you have to show each step.

A and B are 0 indegree remove any one

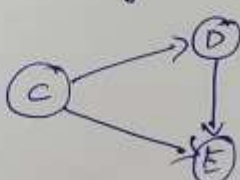


~~B is having indegree zero~~
B is having indegree zero.

visited

1	0	0	0	0
---	---	---	---	---


Each step carries.



C is having indegree zero

visited

1	1	0	0	0
---	---	---	---	---



D is having indegree zero

visited

1	1	1	0	0
---	---	---	---	---

visited

1	1	1	1	0
---	---	---	---	---

Finally visited all the vertex and topological order of visit is

Finally visited

1	1	1	1	1
---	---	---	---	---

A B C D E
B A C D E } 0.5 marks.

Q16. Imagine your friend is trying to find a specific "key" among many keys with unique "patterns" on them. Develop an algorithm inspired by you to systematically search for the right key by comparing its pattern and analyze the time complexity for the given problem. (3)

//Input: An array T [0...n-1] of n characters representing a text and an array P [0...m-1] of m characters representing a pattern

//Output: The index of the first character in the text that starts a matching substring or -1 if the search is unsuccessful ---- 0.5M (Need to mention the input and output)

for i ← 0 to n-m do

 j ← 0

 While j < m and P[j] = T[i+j] do

 j ← j+1

 if j = m return i

return -1 ----- (2M for algorithm)

worst-case time complexity, **O(nm)** ----- (0.5M for analysis)

Q17. The time required for dividing the main problem into sub-problems (T_d) and the time required for merging the solutions of the subproblems (T_m) are important in deciding the time complexity of an algorithm that follows the divide-and-conquer approach. Apply your understanding from the divide-and-conquer approach and establish the relation between T_d and T_m (i.e., $T_d > T_m$ or $T_m > T_d$ or $T_d = T_m$) for the (i) Merge Sort (ii) Binary Search (iii) Matrix Multiplication. (3)

Q18. Suppose a quick sort algorithm is applied for sorting n elements, and at level, the pivot element divides the array of size n into two subarrays of size (n/4) and (3n/4). Analyze the given scenario and write the recurrence relation. Also, solve the framed recurrence relation using the substitution method and identify which case (best/average/worst) this scenario belongs. (2)

The time required for dividing the main problem into sub-problems (T_d) and the time required for merging the solutions of the subproblems (T_m) are important in deciding the time complexity of an algorithm that follows the divide-and-conquer approach. Apply your understanding from the divide-and-conquer approach and establish the relation between T_d and T_m (i.e., $T_d > T_m$ or $T_m > T_d$ or $T_d = T_m$) for the (i) Merge Sort (ii) Binary Search (iii) Matrix Multiplication

Recurrence relation

$$T(n) = aT\left(\frac{n}{b}\right) + f(n)$$

(i) $T(n) = 2T\left(\frac{n}{2}\right) + O(n)$

(ii) $T(n) = T\left(\frac{n}{2}\right) + O(1)$

(iii) $T(n) = 8T\left(\frac{n}{2}\right) + O(n^2)$
or
 $7T$

(i) For Merge Sort $f(n) = O(n)$

This $O(n)$ cost is the merging cost
division cost is constant

\therefore

$$T_m > T_d$$

(ii) For Binary Search $T(n) = O(1)$

We divide the original array into two parts in constant time (time reqd to identify the middle of array). We then solve the half of the part and other half is discarded. So Merge Cost is not there (or you can take it as constant).

$$T_d = T_m$$

(iii) For Matrix multiplication

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

This $O(n^2)$ cost comes when we merge the solution of the subproblems. Division cost is constant as we

simply consider $n/2 \times h/2$

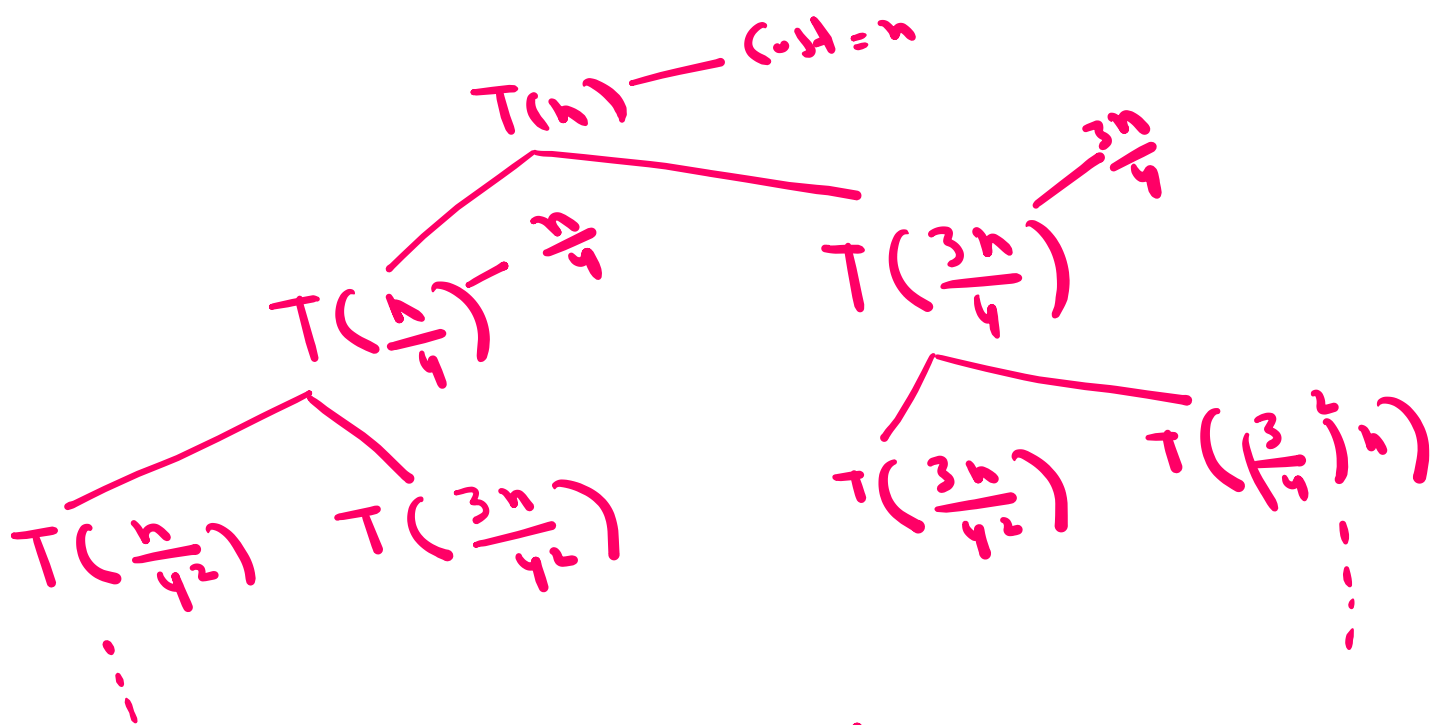
sub matrices while dividing
the problem into subproblems

$$T_m \geq T_d$$

Suppose a quick sort algorithm is applied for sorting n elements, and at level, the pivot element divides the array of size n into two subarrays of size $(n/4)$ and $(3n/4)$. Analyze the given scenario and write the recurrence relation. Also, solve the framed recurrence relation using the substitution method and identify which case (best/average/worst) this scenario belongs

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

(1 Mark)



No. of steps = Height of tree
 $= \log n$

Cost of each step = n

Step-1

$$\boxed{n} +$$

$$\boxed{\frac{n}{4} + \frac{3n}{4}}$$

Step-2

+ ...

log n times

$n + n + n + \dots$ log n times

$$\boxed{= n \log n}$$

[0.5 Marks]

This case belongs to Average Case
Scenario of Quick Sort [0.5 Marks]