Analysis and Designs of Algorithms	
Sub Code: BCS401	**Branch: ISE Sem/Sec: IV / A,B,C**

IAT-1 QUESTION BANK

Module-1

1. Define algorithm. Explain asymptotic notations such as Big Oh, Big Omega and Big Theta with suitable examples.

Marks: 10 | Bloom's Level: L2 | CO: CO1

2. Explain the algorithm design and analysis process. Illustrate the various steps involved in algorithmic problem solving with a neat flow diagram.

Marks: 10 | Bloom's Level: L2 | CO: CO1

3. Explain the general plan for analyzing the efficiency of recursive algorithms. Develop a recursive algorithm to compute factorial of a positive integer and derive its efficiency.

Marks: 10 | Bloom's Level: L3 | CO: CO1

4. Explain the general plan for analyzing the efficiency of recursive algorithms with an example. Also analyze the recursive Tower of Hanoi problem and derive its time complexity.

Marks: 10 | Bloom's Level: L3 | CO: CO1

5. Design an algorithm to search for an element in an array using sequential search. Discuss the best case, worst case and average case efficiencies of the algorithm.

Marks: 10 | Bloom's Level: L3 | CO: CO1

6. Explain the general plan for analyzing the efficiency of non-recursive algorithms. Suggest a non-recursive algorithm to find the maximum element in a list of n numbers and derive its efficiency.

Marks: 8 | Bloom's Level: L2 | CO: CO1

7. Derive the worst-case efficiency of Bubble Sort algorithm.

Marks: 4 | Bloom's Level: L2 | CO: CO1

8. Discuss about Selection Sort algorithm with an example. Derive its time complexity.

Marks: 8 | Bloom's Level: L2 | CO: CO1

9. If $t1(n) \in O(g1(n))$ and $t2(n) \in O(g2(n))$, prove that $t1(n) + t2(n) \in O(\max\{g1(n), g2(n)\})$.

Marks: 4 | Bloom's Level: L2 | CO: CO1

Module – 2

1. Explain the concept of Divide and Conquer. Design Merge Sort algorithm and derive its time complexity.

Marks: 10 | Bloom's Level: L2 / L3 | CO: CO2

2. Design an Insertion Sort algorithm. Obtain its time complexity and apply the algorithm to sort a given list of elements.

Marks: 10 | Bloom's Level: L2 / L3 | CO: CO2

3. Design a Quick Sort algorithm. Obtain its best case, average case and worst case efficiency. Apply the algorithm to sort a given list of elements. 5, 3, 1, 9, 8, 2, 4, 7.

Marks: 10 | Bloom's Level: L2 / L3 | CO: CO2

4. Explain Strassen's Matrix Multiplication method with an example and derive its time complexity.

Marks: 10 | Bloom's Level: L2 / L3 | CO: CO2

5. Explain Strassen's Matrix Multiplication. Apply multiplication to multiply the following matrices:

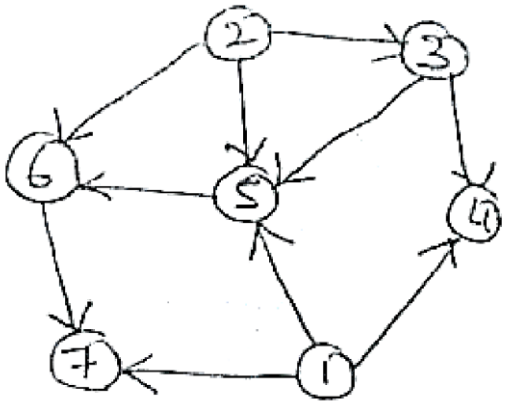
$$\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} \times \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}$$

Marks: 10 | Bloom's Level: L2 / L3 | CO: CO2

6. Define Topological Sorting. Explain the different approaches used for topological sorting such as Source Removal Method and DFS-based Method with examples.

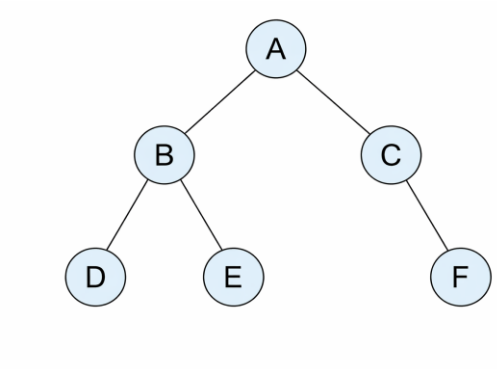
Marks: 10 | Bloom's Level: L2 / L3 | CO: CO2

7. Apply DFS-based Topological Sorting algorithm for the given directed graph.



Marks: 6 | Bloom's Level: L3 | CO: CO2

8. Write algorithms for Preorder, Inorder and Postorder traversals of a tree. Obtain the traversals for a given tree.



Marks: 6 | Bloom's Level: L2 | CO: CO2

9. Distinguish between Decrease and Conquer technique and Divide and Conquer technique with suitable examples.

Marks: 6 | Bloom's Level: L3 | CO: CO2

Module – 3

1. Define AVL Trees. Explain its four rotation types.

Marks: 10 | Bloom's Level: L2 | CO: CO3

2. Define Balanced Search Trees. Explain AVL Trees as a balanced binary search tree.

Marks: 8 | Bloom's Level: L2 | CO: CO3