



Sub Code/ Subject: CS1201/ Data Structures

Year / Sem: II/III

UNIT I - PROBLEM SOLVING

PART – A (2 Marks)

1. Define Program
2. Define Algorithm
3. Define Problem Definition Phase
4. What are the problem solving strategies?
5. Define Top Down Design.
6. What is the basic idea behind Divide & Conquer Strategy?
7. Define Program Verification.
8. Define Input & Output Assertion.
9. Define Symbolic Execution
10. Define Verification Condition
11. Define the qualities of good algorithm.
12. Define Computational Complexity.
13. What is O – notation?
14. What is Recursion? Explain with an example.
15. List out the performance measures of an algorithm.

PART – B

1. Explain in detail the steps involved in Top down Design. (16)
2. Write the verification condition of a program segments with
 - a. Straight line statements (4)
 - b. Branches (6)
 - c. Loops (6)
3. Write a short notes on efficiency of an algorithm. (16)
4. Write a short notes on analysis of an algorithm. (16)
5. (a) Develop an algorithm to compute the sums for the first n terms
 - a. $S=1+(1/2)+(1/3)+....$ (8)
 - (b) Discuss in detail about the implementation of the algorithm. (8)
6. (a) Write an algorithm to reverse the digits of a decimal number. (8)
 - (b) Write an algorithm to compute the Fibonacci series for 'n' terms (8)

UNIT II - LISTS - STACKS AND QUEUES

PART – A (2 Marks)

1. Define ADT
2. List out the operations of the list ADT.
3. Define Dequeue.
4. How do you push & pop elements in a linked stack.
5. Convert the infix expression $a+b*c+(d*c+f)*g$ to its equivalent postfix expression.
6. List the characteristics of stack.
7. List out the applications of stack.
8. Define linked list & its types.
9. Define circular queue.
10. Define cursor space.

PART – B

1. Write a program in C to return the position of an element X in a List L. (16)
2. (a) State & explain the algorithm to perform Radix Sort. (8)
(b) Write a Program in C to create an empty stack and to push an element into it. (8)
3. Explain how queues can be implemented using Arrays (16)
4. (a) Write a 'c' program to multiply two polynomials. (8)
(b) Write a 'c' program to add two polynomials. (8)
5. (a) Write an algorithm to convert infix to postfix expression and explain it with example (8)
(b) Write an algorithm to evaluate a postfix expression and explain it with example (8)
6. (a) Write an algorithm to check given expression contains balanced parenthesis or not.(8)
(b) Write an algorithm for insertion and deletion operation in a circular queue (8)

UNIT III - TREES

PART – A (2 Marks)

1. Explain the representation of priority queue
2. Compare the various hashing Techniques.
3. List out the steps involved in deleting a node from a binary search tree.
4. What is binary heap?
5. Define Binary search tree.
6. List out the various techniques of hashing
7. Define hash function.
8. Show that maximum number of nodes in a binary tree of height H is $2^{H+1} - 1$.
9. Define hashing.
10. Define AVL tree.

PART – B

1. (a) Construct an expression tree for the expression $A+(B-C)*D+(E*F)$ (8)
(b) Write a function to delete the minimum element from a binary heap (8)
2. Write a program in C to create an empty binary search tree & search for an element X in it. (16)
3. Explain in detail about Open Addressing (16)
4. Explain in detail insertion into AVL Trees (16)
5. Write a recursive algorithm for binary tree traversal with an example. (16)
6. Write an algorithm for initializing the hash table and insertion in a separate chaining (16)

UNIT IV - SORTING

PART – A (2 Marks)

1. Mention the time complexities of merge sort & shell sort.
2. What is the worst case complexity of Quick sort?
3. State the algorithmic technique used in Merge sort.
4. What is the average number of inversions in an array of N distinct numbers?
5. What is the average number of comparisons used to heap sort a randomized permutation of N distinct items?
6. Develop pseudo code that will illustrate the process logic in insertion sort.
7. What is the best case time complexity of the quick sort algorithm?
8. Define external sorting.
9. State the algorithmic technique used in Increment Diminishing Sort.
10. Define sorting.

PART – B

1. State & explain the algorithm to perform Heap sort. Also analyze the time complexity of the algorithm (16)
2. Write a C program to perform Merge sort and analyze time complexity of the algorithm. (16)
3. State & explain the algorithm to perform Quick sort. Also analyze the time complexity of the algorithm. (16)
4. State & explain the algorithm to perform Shell sort. Also analyze the time complexity of the algorithm. (16)
5. State & Explain External sorting Algorithm – Two Way Merge, Multiway Merge. (16)

UNIT V - GRAPHS

PART – A (2 Marks)

1. Explain the topological sort.
2. Define NP hard & NP complete problem.
3. Prove that the number of odd degree vertices in a connected graph should be even.
4. What is an adjacency list? When it is used?
5. What is an activity node of a graph?
6. Define Breadth First Search.
7. Define Depth First Search.
8. Define Minimum Spanning Tree.
9. Define Shortest Path of a graph.
10. Define Biconnectivity.

PART – B

1. Formulate an algorithm to find the shortest path using Dijkstra's algorithm and explain with example. (16)
2. Explain the minimum spanning tree algorithms with an example. (16)
3. (a) Write short notes on Biconnectivity. (8)
(b) Write an algorithm for Topological Sort of a graph. (8)
4. Write and explain weighted and unweighted shortest path algorithm (16)
5. Explain the various applications of Depth First Search. (16)
