|  |  |  |
| --- | --- | --- |
| Serial | Assignments  INDEX | Page no. |
| 1. | Implementation of graph using adjacency matrix.   1. Undirected Unweighted 2. Undirected Weighted 3. Directed Unweighted 4. Directed Weighted | 1 - 5 |
| 2. | Implementation of graph using incidence matrix.   1. Undirected Unweighted 2. Undirected Weighted 3. Directed Unweighted 4. Directed Weighted | 6 - 9 |
| 3. | Implementation of graph using adjacency list.   1. Undirected Unweighted 2. Undirected Weighted 3. Directed Unweighted 4. Directed Weighted | 10 - 14 |
| 4. | 1. Consider an undirected or directed graph, do DFS on it, and compute the DFS tree. 2. Compute the no of components of a given undirected graph using DFS. 3. Consider an undirected or directed graph, do DFS on it, & differentiate its edges based on your sequence visiting the vertices. | 15 - 20 |
| 5. | 1. Consider an undirected or directed graph, do BFS on it, and compute the BFS tree. 2. Do BFS for computing the shortest distance and the path between a pair of vertices of an undirected or directed graph, if one exists. | 21 - 24 |
| 6. | Devise a static scheme in computing a polynomial ‘C’ where ‘C’ is computed by   1. Adding two polynomials ‘A’ & ‘B’. 2. Subtracting polynomial ‘B’ from ‘A’. 3. Multiplying two polynomial ‘A’ & ‘B’. 4. Differentiating polynomial ‘A’. | 25 - 31 |
| 7. | Devise a scheme a sparse matrix to represent a sparse matrix M and transpose this representation of M in lexicographic order. | 32 - 35 |
| 8. | Devise a dynamic scheme in computing a polynomial ‘C’ where ‘C’ is computed by   1. Adding two polynomials ‘A’ & ‘B’. 2. Subtracting polynomial ‘B’ from ‘A’. 3. Multiplying two polynomial ‘A’ & ‘B’. 4. Differentiating polynomial ‘A’. | 36 - 41 |
| 9. | Consider a sequence of n elements and construct an AVL tree after insertion of each element into the tree. Also perform deletion of elements from the AVL tree and rebuild the AVL tree as needed. | 42 - 47 |
| 10. | Consider a sequence of n elements and construct a B-tree after insertion of each element into the tree. Also perform deletion of elements from B-tree and obtain the resulting B-tree after each deletion. | 48 - 55 |
| 11. | Implement Heap Sort | 56 - 58 |
| 12. | Implement Shell Sort | 59 - 61 |
| 13. | Implement Graph Sort | 62 - 65 |
| 14. | Implement Sieve Sort | 66 - 68 |
| 15. | Consider the problem of implementing n digit binary integers and compute amortized cost of this problem in terms of n. | 69 - 72 |
| 16. | Perform the operation of ‘Binomial\_Heap\_Extract\_Min’ for a Binomial Heap that comprises almost 6 binomial trees where minimum key value pressed in a node that belonging to the target tree as next to the largest tree. | 73 - 77 |