www.hbpatel.in

CT303-N: Data Structures & Algorithms

(Sample Questions)

Part 1: Introduction

(A) Short Questions:

1(A)-1: What is a data structure?
1(A)-2: What are the major concerns while maintaining a data structure?
1(A)-3: What is an ABSTRACT DATA TYPES?
1(A)-4: Explain the computational complexity with three notations.

(B) Medium and Long Questions:

1(B)-1: Differentiate between primitive data structure and non-primitive data structure.
1(B)-2: Differentiate between linear data structure and non-linear data structure.
1(B)-3: Differentiate between static data structure and dynamic data structure.
1(B)-4: List the various operations which can be performed on a data structure.

Part 2: Linear Data Structure

(A) Short Questions:

2(A)-1: Explain the scenario in which we use SPARSE matrix? OR Why do we need sparse matrix?
2(A)-2: Create a SPARSE matrix for the following matrix.

$$\begin{bmatrix} 0 & 0 & 5 & 2 \\ 0 & 0 & 0 & 6 \\ 9 & 0 & 0 & 0 \\ 7 & 0 & 0 & 0 \end{bmatrix}$$

2(A)-3: Give two real time examples where STACK can be used.
2(A)-4: Write an algorithm/code to for PUSH operation on STACK.
2(A)-5: Write an algorithm/code to for POP operation on STACK.
2(A)-6: When do we use the condition `if (top == size-1)` in STACK?
2(A)-7: When do we use the condition `if (top == -1)` in STACK?
2(A)-8: What is the disadvantage of normal queue? How that is overcome using circular queue?
2(A)-9: Give a case study (example) where PRIORITY QUEUE can be used.
2(A)-10: When do we use the condition `if(rear==size-1)` in QUEUE?
2(A)-11: When do we use the condition `if(front==-1 && rear==-1)` in QUEUE?
2(A)-12: When do we use the condition `if(front==rear)` in QUEUE?
2(A)-13: What is recursion?

(B) Medium and Long Questions:

2(B)-1: Write an algorithm/program to create a SPARSE matrix from a given matrix.
2(B)-2: List various operations which can be performed on a STACK and explain them in brief.
2(B)-3: Show the diagrammatic presentation on following STACK operations.
Let `S` is an instance of stack. Consider the following sequence of operations performed on `S` which initially contains element with 55 as top most elements. What will be the element at the top of the stack after the following sequence of operations?
`S.push(7), S.push(20), S.pop(), S.push(14), S.pop(), S.pop()`
2(B)-4: Convert the equation `(a+b^c^d)*(e+f/d)` from infix to postfix. (Show all the steps)
2(B)-5: Convert the expression `A B - C D E + * + F G + –` from postfix to infix. (Show all the steps)
2(B)-6: Demonstrate (with algorithm/code) the working of Tower of Hanoi using recursion.
2(B)-7: Differentiate STACK and QUEUE.
2(B)-8: List various operations which can be performed on a QUEUE and explain them in brief.
2(B)-9: Explain ENQUEUE operation on a QUEUE using algorithm/code.
2(B)-10: Explain DEQUEUE operation on a QUEUE using algorithm/code.
2(B)-11: Explain ENQUEUE operation on a CIRCULAR QUEUE using algorithm/code.
2(B)-12: Explain DEQUEUE operation on a CIRCULAR QUEUE using algorithm/code.
2(B)-13: List the types of DOUBLE ENDED QUEUE and explain them in brief.

2(B)-14: Demonstrate following sequence of operation on a DOUBLE ENDED QUEUE and show the status of queue (including the `front` and `rear` pointers) after each operation in a diagrammatic way.
`enqueueFront(11), enqueueRear(22), enqueueRear(33), dequeueFront(), enqueueFront(44), enqueueFront(55), dequeueFront(), dequeueRear(), enqueueFront(66), dequeueRear(), dequeueRear(), dequeueFront()`

2(B)-15: Demonstrate the insertion into a PRIORITY QUEUE with following 5 elements and their respective priorities.

Elements	A	B	C	D	E
Priority	2	4	1	5	3

2(B)-16: What are the benefits of using linked list over arrays? OR How dynamic memory allocations (DMA)resolved the issues of static memory allocations (SMA) OR Differentiate DMA Vs. SMA
2(B)-17: List various types of linked lists with diagram.
2(B)-18: Write an algorithm/code to create SINGLY LINKED LIST.
2(B)-19: What are the various ways to INSERT a node into a SINGLY LINKED LIST. Explain any one of them using algorithm/code.
2(B)-20: What are the various ways to DELETE a node into a SINGLY LINKED LIST. Explain any one of them using algorithm/code.
2(B)-21: Write an algorithm/code to create CIRCULAR LINKED LIST.
2(B)-22: Write an algorithm/code to create DOUBLY LINKED LIST.
2(B)-23: Write an algorithm/code to create DOUBLY CIRCULAR LINKED LIST.

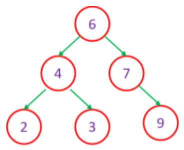
Part 3: Non-Linear Data Structure

(A) Short Questions:

3(A)-1: Explain the following terms with reference to TREE.
(i) node (ii) root (iii) child (iv) parent (v) path (vi) leaf (vii) depth (viii) height (ix) degree (x) level
3(A)-2: What are the properties of a BINARY TREE?
3(A)-3: Define the following terms with reference to GRAPH.
(i) Finite graph (ii) Infinite graph (iii) Trivial graph (iv) Simple graph (v) Multi graph (vi) Null graph (vii) Complete/full graph.
3(A)-4: What is a Spanning Tree?
3(A)-5: What is a Minimum Spanning Tree (MST)?

(B) Medium and Long Questions:

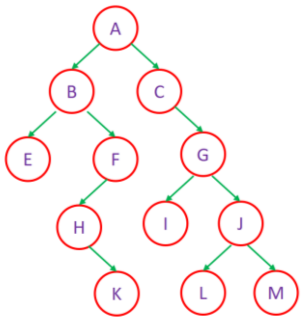
3(B)-1: Write an algorithm/code to create a BINARY TREE.
3(B)-2: What are the different types of TREE traversal mechanisms? For following BINARY TREE, illustrate the various tree traversal outcome.



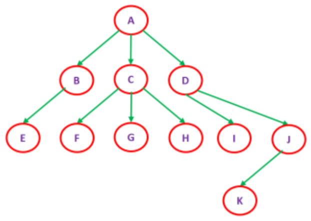
3(B)-3: Write an algorithm/code to demonstrate INORDER/PREORDER/POSTORDER tree traversal mechanism.
3(B)-4: Create a Threaded Binary Tree for the following Binary Tree.

CT303-N: Data Structures & Algorithms

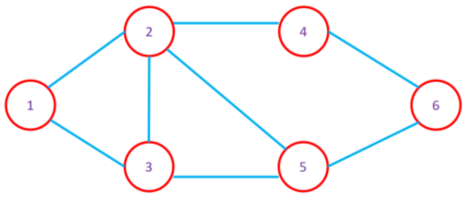
(Sample Questions)



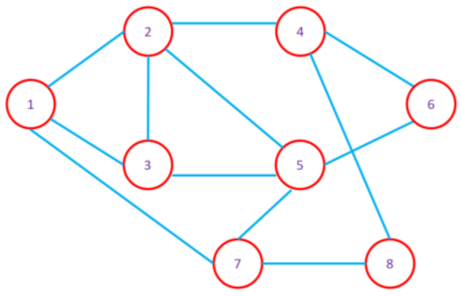
3(B)-5: Create a Binary Search Tree (BST) for following sequence of numbers: 12, 7, 9, 18, 14, 1, 45, 89, 100, 36 59
3(B)-6: Explain the various cases of node-deletion from a Binary Search Tree (BST).
3(B)-7: Constructing Binary Search Tree (BST) from a given Postorder Data (LRV): 20 40 30 70 90 80 60 50
3(B)-8: Constructing Binary Tree from the following General Tree



3(B)-9: Construct an AVL tree from the given data: 13, 16, 10, 8, 54, ,3, 12, 11, 9, 59, 20, 15, 21. Show all the steps clearly.
3(B)-10: Construct a B-Tree (with order m=4) for the given data: 5, 3, 21, 9, 1, 13, 2, 7, 10, 12, 4, 8.
3(B)-11: Write an algorithm/steps to create a 2-3 Tree.
3(B)-12: Create a 2-3 Tree for the given data: 45, 67, 35, 17, 9, 8 ,4, 50
3(B)-13: Construct a RED-BLACK TREE for the given data: 11, 19, 7, 14, 17, 30, 24, 41, 61, 3, 1, 69
3(B)-14: What are the applications of the GRAPH?
3(B)-15: Represent the following GRAPH using Adjacency Matrix method.
3(B)-16: Represent the following GRAPH using Adjacency List method.



3(B)-17: Write an algorithm/code for a Breadth First Search (BFS) for a GRAPH.
3(B)-18: Write an algorithm/code for a Depth First Search (DFS) for a GRAPH
3(B)-19: For the following GRAPH, show the outcome when we use a Breadth First Search (BFS). (Assume: Start=1)
3(B)-20: For the following GRAPH, show the outcome when we use a Depth First Search (DFS).



3(B)-21: Write the general properties of Spanning Tree.

CT303-N: Data Structures & Algorithms

(Sample Questions)

Part 4: Sorting and Searching

(B) Medium and Long Questions:

4(B)-1: Write an algorithm/code for Selection Sort algorithm.
4(B)-2: Write an algorithm/code for Bubble Sort algorithm.
4(B)-3: Write an algorithm/code for Merge Sort algorithm.
4(B)-4: Write an algorithm/code for Insertion Sort algorithm.
4(B)-5: Write an algorithm/code for Quick Sort algorithm.
4(B)-6: Write an algorithm/code for Heap Sort algorithm.
4(B)-7: Write an algorithm/code for Linear Search algorithm.
4(B)-8: Write an algorithm/code for Binary Search algorithm.
4(B)-9: With the data (42, 23, 74, 11, 65, 58), demonstrate the working of Selection Sort algorithm with specifying each step.
4(B)-10: With the data (42, 23, 74, 11, 65, 58), demonstrate the working of Bubble Sort algorithm with specifying each step.
4(B)-11: With the data (66, 33, 40, 22, 55, 88, 11, 80, 20, 50, 44, 77, 30), demonstrate the working of Merge Sort algorithm with specifying each step.
4(B)-12: With the data (42, 23, 74, 11, 65, 58), demonstrate the working of Insertion Sort algorithm with specifying each step.
4(B)-13: With the data (34, 90, 21, 43, 87, 2, 67, 53, 9, 23, 82), demonstrate the working of Quick Sort algorithm with specifying each step.
4(B)-14: With the data (77, 12, 8, 39, 27, 21, 44, 18, 6, 47, 11, 37, 60, 56), demonstrate the working of Heap Sort algorithm with specifying each step.
4(B)-15: With the data (6, 8, 11, 12, 18, 21, 27, 37, 39, 44, 47, 58, 60, 77), demonstrate the working of Binary Search algorithm to search 39, with specifying each step.

Part 5: Hashing

(A) Short Questions:

5(A)-1: What is hashing?
5(A)-2: Explain collision.
5(A)-3: What is a Symbol table? Draw a sample Symbol table.

(B) Medium and Long Questions:

5(B)-1: Explain the Chaining method under the Open hashing (Closed addressing) with example.
5(B)-2: Explain the Linear Probing method under the Closed hashing (Open addressing) with example.
5(B)-3: Explain the Quadratic Probing method under the Closed hashing (Open addressing) with example.
5(B)-4: Explain the Double hashing method under the Closed hashing (Open addressing) with example.
5(B)-5: List the various items which can be stored in a Symbol table.

Part 6: File Structure

(A) Short Questions:

6(A)-1: Why do we need file organization?
6(A)-2: Explain the following terms: (i) File (ii) Database (iii) Record (iv) Key fields

(B) Medium and Long Questions:

6(B)-1: List the various methods of file organization.
6(B)-2: Explain Sequential File Organization.
6(B)-3: Explain Direct Access File Organization.
6(B)-4: Explain Indexed Sequential File Organization.
6(B)-5: Explain Multi-Key File Organization.