CS3301 Data Structures

UNIT 1(LISTS):

2 Marks

1. Question: What is a linked list?
Answer: A linked list is a linear data structure where each element, called a node, contains data and a pointer to the next node in the list. The first node in the list is called the head and the last node is called the tail.

2. Question: What are the different types of linked lists?
Answer: There are three main types of linked lists: singly linked lists, doubly linked lists, and circular linked lists.

3. Question: A singly linked list is a linked list where each node only has a pointer to the next node.
A doubly linked list is a linked list where each node has pointers to the next node and the previous node.
A circular linked list is a linked list where the last node points back to the head node.

4. Question: How do you insert an element into a linked list?
Answer: To insert an element into a linked list, you need to create a new node and set the node's data to the desired value. Then, you need to update the next

pointer of the previous node to point to the new node. Finally, you need to set the new node's next pointer to NULL.
5. Question: How do you delete an element from a linked list?
Answer: To delete an element from a linked list, you need to first find the node that you want to delete. Then, you need to update the next pointer of the previous node to point to the node after the node that you want to delete. Finally, you need to free the memory that was allocated to the node that you deleted.

Detail Question:

1. What are the advantages and disadvantages of singly linked lists?
2. What are the advantages and disadvantages of doubly linked lists?
3. What are the advantages and disadvantages of circular linked lists?
4. How do you find the middle element of a linked list without iterating the list more than once?
5. How do you reverse a linked list in place?
6. How do you check if a linked list has a cycle?
7. How do you remove a cycle from a linked list?
8. How do you merge two sorted linked lists?
9. How do you sort a linked list?
10. How do you implement a polynomial ADT using a linked list?
11. How does radix sort work on a linked list?

UNIT 2(STACKS AND QUEUES):

2 Mark Question:

1. Question: What is a stack?
Answer: A stack is a data structure that follows the Last In First Out (LIFO) principle. This means that the last element that is pushed onto the stack is the first element that is popped off.

2. Question: What are the operations on a stack?
Answer: The basic operations on a stack are push, pop, peek, and is_empty.
Push: This operation adds an element to the top of the stack.
Pop: This operation removes the top element from the stack.
Peek: This operation returns the top element of the stack without removing it.
Is_empty: This operation checks if the stack is empty.

3. Question: What are the applications of stacks?
Answer: Stacks are used in a variety of applications, including:
Balancing symbols in arithmetic expressions
Evaluating arithmetic expressions
Infix to postfix conversion
Function calls
Reversing a string
Backtracking

4. Question: What is a queue?
Answer: A queue is a data structure that follows the First In First Out (FIFO) principle. This means that the first element that is enqueued is the first element that is dequeued.
Question: What are the operations on a queue?
Answer: The basic operations on a queue are enqueue, dequeue, and is_empty.

Enqueue: This operation adds an element to the rear of the queue.
Dequeue: This operation removes the front element from the queue.
Is_empty: This operation checks if the queue is empty.

5. Question: What are the applications of queues?
Answer: Queues are used in a variety of applications, including:
Scheduling jobs
Storing data in a first-in, first-out order
Simulations
Data compression
Breadth-first search

Details Question:

1. What is the difference between a stack and a queue?
2. How do you implement a stack using an array?
3. How do you implement a queue using an array?
4. How do you implement a stack using a linked list?
5. How do you implement a queue using a linked list?
6. How do you check if a stack is empty?
7. How do you check if a queue is empty?
8. How do you push an element onto a stack?
9. How do you pop an element from a stack?
10. How do you enqueue an element into a queue?
11. How do you dequeue an element from a queue?
12. How do you balance the symbols in the expression "(a+b) * (c-d)" using a stack?
14. How do you evaluate the arithmetic expression "(a+b) * (c-d)" using a stack?
15. How do you convert the infix expression "(a+b) * (c-d)" to postfix expression using a stack?
16. How do you implement function calls using a stack?

17. How do you implement a circular queue?
18. What are the advantages and disadvantages of a circular queue over a regular queue?
19. What is dequeue? How is it different from queue?

UNIT III (TREES):

2 Marks:

1. Question: What is a tree?
Answer: A tree is a data structure that consists of nodes connected by edges. The nodes represent data and the edges represent relationships between the data.

2. Question: What is a tree traversal?
Answer: A tree traversal is a process of visiting all the nodes in a tree, exactly once. There are many different ways to traverse a tree, each with its own advantages and disadvantages.

3. Question: What are the different types of tree traversals?
Answer: The most common tree traversals are:
Inorder traversal: Visits the left subtree, the root node, and then the right subtree.
Preorder traversal: Visits the root node, the left subtree, and then the right subtree.
Postorder traversal: Visits the left subtree, the right subtree, and then the root node.

4. Question: What is a binary tree?
Answer: A binary tree is a tree in which each node has at most two children. The children of a node are called the left child and the right child.

5. Question: What are the different types of binary trees?
Answer: The most common types of binary trees are:
Full binary tree: A binary tree in which every node has either zero or two children.

Complete binary tree: A binary tree in which every level, except possibly the last level, is completely filled.
Perfect binary tree: A binary tree in which all levels are completely filled and all leaves are at the same level.

6. Question: What is an expression tree?
Answer: An expression tree is a tree that represents an arithmetic expression. The nodes of the tree represent the operators and operands in the expression.

7. Question: What is a binary search tree?
Answer: A binary search tree is a binary tree that is used to store data in sorted order. The left subtree of a node contains all the nodes that are smaller than the node, and the right subtree contains all the nodes that are larger than the node.

8. Question: What is an AVL tree?
Answer: An AVL tree is a self-balancing binary search tree. This means that the tree is always kept in a balanced state, so that the worst-case time complexity of operations such as search and insertion is O(log n).

9. Question: What is a priority queue?
Answer: A priority queue is a data structure that stores elements in priority order. The element with the highest priority is always at the front of the queue.

10. Question: What is a binary heap?
Answer: A binary heap is a special type of priority queue that is implemented using a binary tree. The heap property is that the parent node must always have a smaller priority than its children.

Details Question:

1. What are the advantages and disadvantages of using trees?
2. How do you implement a tree using an array?
3. How do you implement a tree using a linked list?
4. How do you traverse a tree in-order?
5. How do you traverse a tree pre-order?
6. How do you traverse a tree post-order?
7. What is the difference between a binary tree and a general tree?
8. What are the different types of binary trees?
9. What is an expression tree?
10. How do you construct an expression tree from an arithmetic expression?
11. What is a binary search tree?
12. What are the advantages and disadvantages of using a binary search tree?
13. What is an AVL tree?
14. What are the advantages and disadvantages of using an AVL tree?
15. What is a priority queue?
16. What are the different types of priority queues?
17. What is a binary heap?
18. What are the advantages and disadvantages of using a binary heap?

UNIT IV (MULTIWAY SEARCH TREES AND GRAPHS)

2mark

1. Question: What is a B-tree?
Answer: A B-tree is a multiway search tree that is balanced. This means that the tree is always kept in a state where the height of the tree is O(log n), where n is the number of nodes in the tree.

2. Question: What is a B+ tree?
Answer: A B+ tree is a special type of B-tree that is optimized for search operations. In a B+ tree, all the leaf nodes are stored together at the bottom of the tree, and all the internal nodes store pointers to the leaf nodes.

3 .Question: What is a graph?
Answer: A graph is a data structure that consists of vertices (also called nodes) and edges (also called links). The vertices represent data and the edges represent relationships between the data.

4 .Question: What is a representation of graphs?
Answer: There are many different ways to represent graphs. The most common way is to use an adjacency list. An adjacency list is a list of lists, where each list stores the vertices that are adjacent to a given vertex.

5. Question: What are the different types of graphs?
Answer: The most common types of graphs are:
Undirected graph: A graph where the edges do not have a direction.
Directed graph: A graph where the edges have a direction.
Weighted graph: A graph where the edges have weights.
Complete graph: A graph where every vertex is connected to every other vertex.

Disconnected graph: A graph where there are two or more vertices that are not connected to each other.

Details Question:

1. What are the advantages and disadvantages of using B-trees?
2. What are the advantages and disadvantages of using B+ trees?
3. What are the different types of graph traversals?
4. What is the difference between a breadth-first traversal and a depth-first traversal?
5. What is bi-connectivity?
6. What is an Euler circuit?
7. What is topological sort?
8. What is Dijkstra's algorithm?
9. What is a minimum spanning tree?
10. What are Prim's algorithm and Kruskal's algorithm?

UNIT 5 (SEARCHING, SORTING AND HASHING TECHNIQUES)

2 Marks:

1. Question: What is searching?
Answer: Searching is the process of finding a specific item in a data structure.

2. Question: What is sorting?
Answer: Sorting is the process of arranging a data structure in a particular order.

3. Question: What is a hash function?
Answer: A hash function is a function that maps a key to a value.

4. Question: What is separate chaining?
Answer: Separate chaining is a collision resolution technique in hashing where each bucket points to a linked list of elements that have the same hash value.

5. Question: What is open addressing?
Answer: Open addressing is a collision resolution technique in hashing where elements are stored in buckets based on their hash value.

Details Question:

1. What are the different types of searching?

2. What are the different types of sorting?
3. What are the advantages and disadvantages of using hash functions?
4. What are the different collision resolution techniques in hashing?
5. What is the difference between separate chaining and open addressing?
6. What is rehashing?
7. What is extendible hashing?
8. How do you implement linear search?
9. How do you implement binary search?
10. How do you implement bubble sort?
11. How do you implement selection sort?
12. How do you implement insertion sort?
13. How do you implement shell sort?
14. How do you implement merge sort?
15. How do you implement hash functions?
16. How do you implement separate chaining?
17. How do you implement open addressing?
18. How do you implement rehashing?
19. How do you implement extendible hashing?