Data Structures an Applications

Linked Lists

1. What are the advantages and disadvantages of representing a group of items as an array versus a linear linked lists.
2. Write a c program to reverse the elements of a given linked lists.
3. Explain the various operations on a singly linked list. (insert, delete, display)
4. What is doubly linked list, circular linked list? Pseudocode or an algorithm for various operations on both. What are the advantages and disadvantages of both.
5. How can a polynomial in 2 variables (x and y) be represented by singly linked list? Each node should represent a term and should contain powers of x and y, as well as coefficient of the same. Write a c program to evaluate given polynomial for given values of x and y.
6. Draw and explain the representation of sparse matrix of order mxn using linked lists.
7. Mention the different uses of header node.
8. Implementation of stack and queues using singly linked list.
9. Implementation of stack and queues using doubly linked list.
10. Implementation of stack and queues using circular linked list.
11. Write an algorithm to perform the following operations.
    i. Combine 2 ordered lists into a singly linked list. (merge sort)
    ii. Form a list containing the union of the elements of two lists.
    iii. Form a list containing the intersection of the elements of two lists
    iv. Make a second copy of a list.
    v. Remove the duplicate element of the list
    vi. Write a c function **search(l,x)** that accepts a pointer ‘l’ to a list of integers and an integer x and returns a pointer to a node containing x, if it exists and the null pointer otherwise. Write another function srchinsrt(l,x), that adds ‘x’ to ‘l’ if it is not found and always returns a pointer to a node containing ‘xs’.
12. Concatenation of linked list. Explain with pseudocode for singly, doubly and circular.

	STACKS

1	Write a C recursive function to compute a+b where a and b are nonnegative integers using the following function :
	int succ(int x)
	{
	return (x++);
	}

2	Evaluate the following postfix expression: 6 2 3 + - 3 8 2 / + * 2 $ 3 + Show the contents of stack, op1,op2,symbol and value for each symbol read in the following format Symbol op1 op2 value stack-contents
3	Write a C subroutine to determine if an input string is of the form x C y where x is a string consisting of letters A and B and y is the reverse of x. Implement using stacks. [Examples: ABBCBBA and ABBACABBA are some of the strings of the given form. ABCAB and BAACABA are some of the strings not of the given form.]
4	For the following definition of f(n) write C recursive function and compute how many additions are done to evaluate f(6). f(n)=n if n=0 or n=1 f(n)=f(n-2)+f(n-1) if n>=2
5	Illustrate the working of infix expression to postfix expression algorithm to get postfix form of the infix expression ((A-(B+C))*D)$(E+F) by tabulating the result of each input character read in in the format given below: Read -Symbol postfix stack-contents
6	Show how to implement a stack of integers in C by using an array int s[stacksize] where s[0] is used to contain the index of the top element of the stack and s[1] through s[stacksize-1]contain the elements on the stack. Write C declaration and functions push(), pop() empty() for this implementation.
7	Convert the following given infix expressions to postfix and prefix expressions respectively. i. A + B * (C - (D / E) ^ G) ii. F - (K / (H * S) ^ J) + X Also write a C function **void conversion_infix_to_postfix(struct stack *s, char infix[ ]),** to convert infix expression to postfix expression.
8	Write a recursive call to move ‘n’ discs from peg A to peg C using B as auxiliary peg to solve the Tower of Hanoi problem.
9	Write a c function **int Balance_infix_expression(struct stack *s, char infix [ ])** to check whether the given input expression is balanced expression or not.
10	Write a c function **int is_palindrome(char str[ ])** to check whether given string is a palindrome or not using stacks.
11	Write C functions for the following operations of stack. i. Push() ii. Pop() iii. Peek() iv.Display()

12	Define Stack. List the different applications of stack.
13	The Greatest Common Divisor of two integers x and y is defined as follows, gcd(x,y) = y if (y < = x && x % y = 0) gcd(x,y) = gcd(y,x) if (x < y) gcd(x,y) = gcd (y, x % y) otherwise Write a recursive C function to compute gcd (x, y). Also find for how many times the recursive function is called for the values of gcd (20,75).
	QUEUES
1	Differentiate between queue and priority queue.
2	In a bank, the customers entering for the cash transactions are made to sit in a row of chairs. Every customer entering for the transaction gets a token and he/she sits on a chair beside the customer at the tail end. The customer who is sitting at the initial chair in that row will be called for the transaction and is the first to leave the chair. Write pseudocodes to simulate this customer entry / exit model for the bank.
3	Write C routines to implement queue operations using output restricted deque for which the operations removeleft() , insertright() and insertleft() are valid.
4	Explain the different ways of implementing the structure of a priority queue.
5	Write C functions to implement enqueue(insert) and dequeue(remove) of circular queues.
6	Write C function to reverse all the elements in a queue.
7	Differentiate between i. Stacks and Queues ii. Linear queue and Circular queue
8	Write necessary C modules to implement an Ascending Priority Queue accepting integers as an input and ensure priority at the time of insertion.
9	Discuss the limitations of linear queues. Name and describe which data structure has overcome the limitations of linear queues.
10	Write the c functions for the following operations of circular queues. i. CQinsert() ii. CQdelete()
11	Explain with the pseudocode to implement queue operations using input Restricted Deque (double ended queue) works. insertqfront(), remvleft(), remvright(), displayq()
12	Compare and contrast between Stacks and Queues of linear data structure.
13	Discuss the limitations of linear queues. Write an algorithm, how the insertion of elements and deletion are done in circular queues.
14	Write a C routines for the implementation of a Descending Priority Queue with following given operations: • pqinsert ( ) pqmaxdelete ( ) pqempty ( )
15	Discuss the advantages of circular queues over linear queues. Explain the different

	scenarios of priority queues.
16	Show how to implement a queue of integers using an array q[10], where q[0] is used to indicate the front of the queue, q[1] is used to indicate its rear and q[2] through q[9] are used to contain queue elements. Also, show how to initialize such an array to represent empty queue and write routines dequeue() and enqueue() operations for such an implementation.
17	Explain with the pseudocode to implement queue operations using Output Restricted Deque (double ended queue) works. Insertqfront ( ), insertqqrear ( ), remvleft ( ) are valid display ( )
	TREES-I
1	Define the following and give example for each: i. Strictly binary tree ii. Level of the tree
2	Write C function to print maximum and minimum elements of a given BST.
3	Discuss various representations of a binary tree.
4	Construct an expression tree for the given postfix expression: A B C * + D E * F + G / - Show the contents of a stack and also draw the corresponding tree .for each step of construction.
5	Define the following with examples. i. Complete Binary tree ii. Strictly binary tree iii. AVL tree iv. Height of a tree v. Almost complete binary tree
6	Illustrate and discuss all the cases on deleting an element from the Binary Search Tree.
7	Define Binary tree. Write the properties of binary search tree.
8	Write a C function **BST_search(struct node *, int)** to search an element in binary search tree.
9	Construct a binary expression tree for the given postfix expression. Write the steps involved in constructing the binary expression tree using stack representation. a b + c d * e / - f g / h * + Also write the preorder traversal of the constructed expression tree and evaluate with a=3,b=9,c=8,d=2,e=7,f=4,g=6,h=1
10	Illustrate and discuss all the cases on deleting an element from the Binary Search Tree for the constructed BST given in the question 8c.
11	Construct a Binary Search Tree for the following elements. 65,45,80,30,49,75,100,20,38,49,69,78,85,110 Also write the Inorder traversal and Preorder traversal for the constructed binary tree. Write the recursive call function **Inorder(struct node *)**for the inorder traversal and **Preorder(struct node *)** for the preorder traversal of a tree.

TREES-II	
1	Define B-tree. Write the properties of B-trees .Explain the construction of 2-3 tree(B tree of order 3) for the following data: 50, 60, 70, 40, 30, 20, 10, 80, 90,100 Delete the elements in the sequence 70, 100, 80 from the tree constructed. Also draw the tree after each insertion and deletion.
2	Starting from an empty height balanced tree(AVL), insert the following data one by one in the sequence as given below: 14, 17, 11, 7, 53, 4, 13, 12, 8, 60, 19, 16, 20 Also draw the tree after each insertion.
3	Explain the construction of 2-3-4 tree(B tree of order 4) for the following data: 20, 50, 40, 70, 80, 15, 90,100 Delete the elements in the sequence 80, 40, 15 from the tree constructed. Also draw the tree after each insertion and deletion.
4	Define AVL. Discuss the different rotations of AVL trees. Explain the different cases of deleting an element from a constructed AVL tree. Also construct an AVL tree for the following data given. 63, 9, 19, 27, 18, 108, 99, 81
5	Construct BST for the given set of data and check whether it is height balanced tree (AVL) or not. If not convert and reconstruct the balanced AVL tree. 20, 11, 5, 32, 40, 2, 4, 27, 23

* * * * * * * * * *