Module 2 : Stacks & Queues

1. Given an array of size 4. Perform the following operations on the ***Stack.***
    a. Push 3 values 98, 67, 54
    b. Pop
    c. Push values 32, 59, 13
2. Illustrate the ***Parenthesis Matching*** using Stacks for the given expression :
   (A + B) * C - (D - E) * (F + G)
3. Illustrate the conversion of ***Infix Expression to Postfix Expression*** using Stack :
   (A – B / C) * (A / K – L)
4. ***Evaluate the Postfix Expression*** evaluation using Stacks : **2 5 3 6 + * * 5 / 2 -**
5. Given an array of size 5. Illustrate the status of a ***Queue*** for the given scenario.
    a. Enqueue the values 23, 56, 89, 45.
    b. Perform 1 dequeue
    c. Enqueue the values 67, 83, 98
6. Given an array of size 5. Illustrate the status of a ***Circular Queue*** for the given scenario.
    a. Enqueue the values 23, 56, 89, 45.
    b. Perform 1 dequeue
    c. Enqueue the values 67, 83, 98
7. Given an array of size 5. Illustrate the status of a ***Double Ended Queue*** for the given scenario
    a. Enqueue the values 23, 56 from the beginning of the Queue
    b. Enqueue the values 89, 45 from the End of the Queue
    c. Perform 1 dequeue from the Beginning
    d. Perform 1 dequeue from the End
    e. Enqueue the values 67, 83 from the End
    f. Enqueue the values 98 from the Beginning
8. Given an array of size 5. Perform the following operations on a ***Priority Queue*** assuming, 1-highest priority and 5-least priority:
    a. Enqueue (20, 1), (30,3)
    b. Enqueue (40,2)
    c. Dequeue
    d. Enqueue (60,1), (70,4), (80,2), (87,5)
    e. Dequeue

Module - 3 : Linked List

1. Diagrammatically demonstrate the following operations on a ***Singly Linked List***
    a. Insert at Beg. values 23, 45, 67, 89
    b. Perform 2 Delete from End
    c. Insert at Beg. 56, 67
2. Diagrammatically demonstrate the following operations on a ***Singly Linked List***
    a. Insert at End. values 23, 45, 67, 89
    b. Perform 2 Delete at Beg
    c. Insert at End. 56, 67
3. Diagrammatically demonstrate the following operations on a ***Singly Linked List***
    a. Insert at Beg, values 23, 45, 67, 89
    b. Delete at specific node 45
    c. Insert at Beg. 46, 67
4. Perform the following operations on the ***Stack implemented using a Singly Linked List***
    a. Push 3 values 98, 67, 54
    b. Pop
    c. Push values 32, 59, 13
5. Perform the following operations on a ***Queue implemented using Singly Linked List***
    a. Enqueue the values 23, 56, 89, 45.
    b. Perform 1 dequeue
    c. Enqueue the values 67, 83
6. Diagrammatically demonstrate the following operations on a ***Circular Linked List***
    a. Insert at Beg. values 23, 45, 67, 89
    b. Perform 2 Delete from End
    c. Insert at Beg. 56, 67
7. Diagrammatically demonstrate the following operations on a ***Circular Linked List***
    a. Insert at End. values 23, 45, 67, 89
    b. Perform 2 Delete at Beg
    c. Insert at End. 56, 67
8. Diagrammatically demonstrate the following operations on a ***Doubly Linked List***
    a. Insert at Beg, values 23, 45, 67, 89
    b. Delete at specific node 45

9. Diagrammatically demonstrate the following operations on a ***Doubly Linked List***
    a. Insert at Beg. values 23, 45, 67, 89
    b. Perform 2 Delete from End
    c. Insert at Beg. 56, 67
10. Diagrammatically demonstrate the following operations on a ***Doubly Linked List***
    a. Insert at End. values 23, 45, 67, 89
    b. Perform 2 Delete at Beg
    c. Insert at End. 56, 67
11. Demonstrate the representation of the given ***Polynomial Expression using SLL***
    Given, $82x^2y + 15xy^2 + 2xy + 15$
12. Demonstrate how ***Polynomial Addition*** is performed using Singly Linked List
    Given, P1 : $2x^4 + 15x^3 + 20$ and P2 : $12x^3 + 7x^2 + 9x + 12$

Module - 4 : Tree

1. Construct a ***Binary Search Tree*** with values 23, 11, 67, 19, 56, 10, 89. Show all the intermediary trees while inserting values. Perform the deletion of an internal node & root node
2. Construct a ***Binary Tree*** with values 23, 11, 67, 19, 56, 10, 89. Show all the intermediary trees while inserting values
3. Construct an ***Expression Tree*** for the given expression : **a+b - c / d ^ f**
4. The characters a to h have the set of frequencies based on the first 8 Fibonacci numbers as follows: a : 1, b : 1, c : 2, d : 3, e : 5, f : 8, g : 13, h : 21 . Huffman code is used to represent the characters. What is the sequence of characters corresponding to the following code?
   110111100111010 (A) fdheg (B) ecgdf (C) dchfg (D) fehdg
5. How many bits may be required for encoding the message ‘mississippi’?
6. Huffman Encoding for the given data

character	Frequency
a	5
b	9
c	12
d	13
e	16
f	45

Module 5 : Types of Tree

1. Construct an ***AVL Tree*** with values 23, 11, 67, 19, 56, 10, 89. Show all the intermediary trees while inserting values
2. Construct a ***Max Heap*** with values 23, 11, 67, 19, 56, 10, 89. Show all the intermediary trees while inserting values
3. Construct a ***Min Heap*** with values 23, 11, 67, 19, 56, 10, 89. Show all the intermediary trees while inserting values
4. Construct a ***Red Black Tree*** with values 23, 11, 67, 19, 56, 10, 89. Show all the intermediary trees while inserting values
5. Construct a ***B Tree*** with values 23, 11, 67, 19, 56, 10, 89. Show all the intermediary trees while inserting values
6. Construct a ***B+ Tree*** with values 23, 11, 67, 19, 56, 10, 89. Show all the intermediary trees while inserting values

Module - 6 : Searching & Sorting

1. Given an array of size 7 with the following values - 78, 89, 90, 150 167, 183, 203. Perform a Binary Search for the values - 89 and 202. Compute the comparisons required.
2. Use the Hashing Function - Division Method (m = 10) for hashing the values 397, 786, 988, 344, 764, 237, 886. Count the number of Collisions and solve the Collision using Linear Probing
3. Use the Hashing Function - Mid Square Method (r = 1) for hashing the values 397, 786, 988, 344, 764, 237, 886. Count the number of Collisions and solve the Collision using Linear Probing
4. Use the Hashing Function - Multiplication Method (A = 0.618) for hashing the values 397, 786, 988, 344, 764, 237, 886. Count the number of Collisions and solve the Collision using Linear Probing
5. Perform Selection Sort with the following values - 23, 11, 67, 19, 56, 10, 89. Count the number of comparisons and swaps required.
6. Perform Insertion Sort with the following values - 23, 11, 67, 19, 56, 10, 89. Count the number of comparisons and swaps required
7. Perform Merge Sort with the following values - 23, 11, 67, 19, 56, 10, 89. Count the number of comparisons and swaps required

8. Perform Quick Sort with the following values - 23, 11, 67, 19, 56, 10, 89. Count the number of comparisons and swaps required
9. Perform Heap Sort with the following values - 23, 11, 67, 19, 56, 10, 89. Count the number of comparisons and swaps required
10. Perform Bucket Sort with the following values - 23, 11, 67, 19, 56, 10, 89. Count the number of comparisons and swaps required