**WAP to convert a given Infix expression into its equivalent Postfix expression and evaluate it using stack.**

**WAP to convert a given Infix expression into its equivalent Prefix expression and evaluate it using stack.**



**WAP to implement multiple stack i.e. More than two stacks using array and perform following operations on it. A. PUSH, B. POP, C. StackFull D. StackeEmpty E. Display Stack.**



**A person is living in house having 5 rooms. These rooms are adjacent to each other. There is a treasure which is electronically locked and to unlock it we need a code. In last room there is a box in which some decimal number is written. We need to convert that number into binary to open treasure which is kept in room no.1 . We need to move from room no.1 to 2 to 3 and so on and follow the reverse way to come back i.e. from 5 to 4 to 3 etc. Apply suitable logic to implement this scenario by using stacks.**



**WAP to implement following by using stack. A. Factorial of a given number B. Generation of Fibonacci series**



**We Fly Anywhere Airlines (WFAA) is considering redesigning their ticket counters for airline passengers. They would like to have two separate waiting lines, one for regular customers and one for frequent flyers. Assuming there is only one ticket agent available to serve all passengers, they would like to determine the average waiting time for both types of passengers using various strategies for taking passengers from the waiting lines. WAP to simulate this situation.**

**An operating system assigns job to print queues based on the number of pages to be printed (1 to 50 pages). You may assume that the system printers are able to print 10 page per minute. Smaller print jobs are printed before larger print jobs and print jobs are processed from a single print queue implemented as a priority queue). The system administrators would like to compare the time required to process a set of print jobs using 1, 2, or 3 system printers. Write a program which simulates processing 100 print jobs of varying lengths using either 1, 2, or 3 printers. Assume that a print request is made every minute and that the number of pages to print varies from 1 to 50 pages. To be fair, you will need to process the same set of print jobs each time you add a printer. The output from your program should indicate the order in which the jobs were received, the order in which they were printed, and the time required to process the set of print jobs. If more than one printer is being used, indicate which printer each job was printed on.**



**Write a program which simulates the operation of a busy airport which has only two runways to handle all takeoffs and landings. You may assume that each takeoff or landing takes 15 minutes to complete. One runway request is made during each five minute time interval and likelihood of landing request is the same as for takeoff. Priority is given to planes requesting a landing. If a request cannot be honored it is added to a takeoff or landing queue. Your program should simulate 120 minutes of activity at the airport. Each request for runway clearance should be time-stamped and added to the appropriate queue. The output from your program should include the final queue contents, the number of take offs completed, the number of landings completed, and the average number of minutes spent in each queue.**



**Write a Program to implement circular double ended queue where user can add and remove the elements from both front and rear of the queue**



**Write a Program to simulate the following situation. Computer is a multitasking device. We need to download some document as well as listen music and play game simultaneously. There is a system queue which decides which task to be done first. Assume that for download application priority is highest and game playing is having lowest priority. After completion of one type of tasks like all download operations then the second queue will be processed.**



**Write a Program to implement multiple queue i.e. two queues using array and perform following operations on it. A. Addq, B. Delq, C. Display Queue.**



Fifth Assignment

**Given a list, split it into two sublists — one for the front half, and one for the back half. If the number of elements is odd, the extra element should go in the front list. So FrontBackSplit() on the list {2, 3, 5, 7, 11} should yield the two lists {2, 3, 5} and {7, 11}. Getting this right for all the cases is harder than it looks. You should check your solution against a few cases (length = 2, length = 3, length=4) to make sure that the list gets split correctly near the short-list boundary conditions. If it works right for length=4, it probably works right for length=1000. You will probably need special case code to deal with the (length <2) cases.**

**WAP to perform addition of two polynomials using singly linked list.**



**Write an iterative Reverse() function that reverses a list by rearranging all the next pointers and the head pointer. Ideally, Reverse() should only need to make one pass of the list.**



**WAP to create doubly linked list and perform following operations on it. A) Insert (all cases) 2. Delete (all cases).**



**WAP to merge two sorted Doubly linked lists and display their result.**



**Implement Push and POP operations of STACK on Doubly linked lists**



**Implement ADD and DELETE operations of QUEUE on Doubly linked lists**



Seventh Assignment

**Write a Program to create a Binary Tree and perform following nonrecursive operations on it. a. Preorder Traversal b. Postorder Traversal c. Count total no. of nodes d. Display height of a tree.**

**Write a Program to create a Binary Tree and perform following nonrecursive operations on it. a. Levelwise display b. Mirror image c. Display height of a tree.**



**Write a program to illustrate operations on a BST holding numeric keys. The menu must include: • Insert • Delete • Find • Show**



**Write a program to illustrate operations on a BST holding numeric keys. The menu must include: • Insert • Mirror Image • Find • Post order (nonrecursive)**



**Write a Program to create a Binary Tree and perform following Nonrecursive operations on it. a. Inorder Traversal b. Preorder Traversal c. Display Number of Leaf Nodes d. Mirror Image**



**Write a Program to create a Binary Search Tree and perform following nonrecursive operations on it. a. Preorder Traversal b. Inorder Traversal c. Display Number of Leaf Nodes d. Mirror Image**



**Write a Program to create a Binary Search Tree and perform following nonrecursive operations on it. a. Preorder Traversal b. Postorder Traversal c. Display total Number of Nodes d. Display Leaf nodes.**



**Write a Program to create a Binary Search Tree and perform deletion of a node from it. Also display the tree in nonrecursive postorder way.**



**Write a Program to create a Binary Search Tree and display it levelwise. Also perform deletion of a node from it.**



**Write a Program to create a Binary Search Tree and display its mirror image with and without disturbing the original tree. Also display height of a tree using nonrecursion.**



**Write a program to efficiently search a particular employee record by using Tree data structure. Also sort the data on emp-id in ascending order.**



Eighth Assignment

**Write a Program to create Inorder Threaded Binary Tree and Traverse it in Preorder way.**

**Write a Program to create Inorder Threaded Binary Tree and Traverse it in Inorder way.**



**Write a Program to implement AVL tree and perform different rotations on it and display it Levelwise.**



Ninth Assignment

**Write a Program to accept a graph from user and represent it with Adjacency Matrix and perform BFS and DFS traversals on it.**

**Write a Program to implement Prim’s algorithm to find minimum spanning tree of a user defined graph. Use Adjacency List to represent a graph.**



**Write a Program to implement Kruskal’s algorithm to find minimum spanning tree of a user defined graph. Use Adjacency List to represent a graph.**



**Write a Program to implement Dijkstra’s algorithm to find shortest distance between two nodes of a user defined graph. Use Adjacency List to represent a graph.**



**Write a Program to accept a graph from user and represent it with Adjacency List and perform BFS and DFS traversals on it.**



**Write a Program to implement Kruskal’s algorithm to find minimum spanning tree of a user defined graph. Use Adjacency Matrix to represent a graph.**



**Write a Program to implement Dijkstra’s algorithm to find shortest distance between two nodes of a user defined graph. Use Adjacency Matrix to represent a graph.**



**Write a Program to implement Prim’s algorithm to find minimum spanning tree of a user defined graph. Use Adjacency List to represent a graph.**



**Write a Program to implement Kruskal’s algorithm to find minimum spanning tree of a user defined graph. Use Adjacency List to represent a graph.**



**Write a Program to implement Dijkstra’s algorithm to find shortest distance between two nodes of a user defined graph. Use Adjacency List to represent a graph.**



**Write a Program to implement Prim’s algorithm to find minimum spanning tree of a user defined graph. Use Adjacency Matrix to represent a graph.**



|  |
| --- |
| **WAP to implement Heap sort and Quick Sort on 1D array of Student structure (contains student\_name, student\_roll\_no, total\_marks), with key as student\_roll\_no. And count the number of swap performed.** |
| **WAP to implement Heap sort and Merge Sort on 1D array of Student structure (contains student\_name, student\_roll\_no, total\_marks), with key as student\_roll\_no. And count the number of swap performed.** |
| **WAP to implement Quick sort and Merge Sort on 1D array of Employee structure (contains employee\_name, emp\_no, emp\_salary), with key as emp\_no. And count the number of swap performed.** |
| **Assume that an array A with n elements was sorted in an ascending order, but two of its elements swapped their positions by a mistake while maintaining the array. Write a code to identify the swapped pair of elements and their positions in the asymptotically best possible time. [Assume that all given elements are distinct integers.]** |
| **Implement following hashing Techniques by assuming suitable input and Table Size.**   1. **Linear Probing** 2. **Quadratic Probing** 3. **Double Hashing.**   **Also mention number of collisions occurred while inserting a Data in hash table.** |