ACADEMY OF TECHNOLOGY

DEPARTMENT OF CSE

Faculty Name: Dilip Kr. Maity and Prasenjit Das

Subject with code : Data Structure and Algorithms Lab [PCC-CS391]

Semester / Branch : 3rd/ COMPUTER SCIENCE AND ENGINEERING

CREDIT :2

ASSIGNMENT LIST

ASSIGNMENT 1:

Outcome: Students will be able to understand organization of data in an array, Advantage and Disadvantage of Array. Search an element using linear and binary search techniques.

Problem Statement: Write a menu driven program to perform the following operations on an array.

(a) *insert* an element *x* at position *k* in the array.

(b) *remove* an element *x* from the array.

(c) *search* an element *x* from the array using linear search(check no. of comparison).

(d) *search* an element *x* from the array using binary search(check no. of comparison).

(e) *display* the array.

ASSIGNMENT 2:

Outcome: Students will be able to understand ADT as Stack, Different operations on Stack.

Problem Statement: Write a menu driven program in C to implement a Stack using array and perform the following operations.

(a) *isF ull*() function to check whether the *Stack* is full or not.

(b) *isEmpty*() function to check whether the *Stack* is empty or not.

(c) *peek*() function to read the stack top element without deleting it.

(d) *push*(*item*) function to insert an element *item* in the *Stack*.

(e) *pop*() function to read and remove an element from the *Stack*.

(f) *display*() function to display the entire stack.

ASSIGNMENT 3:

Outcome: Students will be able to understand ADT as Linear Queue, Different operations on Queue and its disadvantages.

Problem Statement: Write a menu driven program in C to implement a *Queue* using array and perform the following operations.

(a) *isF ull*() function to check whether the *Queue* is full or not.

(b) *isEmpty*() function to check whether the *Queue* is empty or not.

(c) *insert*(*item*) function to insert an element *item* in the *Queue*.

(d) *delete*() function to read and remove an element from the *Queue*.

(e) *display*() function to display the entire *Queue*.

ASSIGNMENT 3:

Outcome: Students will be able to understand different operations of Cicular and how tpo resove the problem occurs linear Queue.

Problem Statement: Write a menu driven program in C to implement a *Circular Queue* using array and perform the following operations.

(a) *isF ull*() function to check whether the *Circular Queue* is full or not.

(b) *isEmpty*() function to check whether the *Circular Queue* is empty or not.

(c) *insert*(*item*) function to insert an element *item* in the *Circular Queue*.

(d) *delete*() function to read and remove an element from the *Circular Queue*.

(e) *display*() function to display the entire *Circular Queue*.

ASSIGNMENT 4:

Outcome: Students will be able to understand different Applications of Stack.

Problem Statement: Write a C program to Convert Infix to Postfix Expression using Stack. Assume that there are only four operators (∗, /, +, −,) in an *infix* expression and operand may be an alphabet or a digit.

ASSIGNMENT 5:

Outcome: Students will be able to understand different Applications of Stack.

Problem Statement: Write a C program to evaluate a given postfix expression. Assume that there are only four operators ($*$, /, +, −,) in a *postfix* expression and operand is single digit only.

ASSIGNMENT 6:

Outcome: Students will be able to understand Dynamic memory allocation and different operations of linked list.

Problem Statement: Write a menu driven program in C or C++ to perform the following operations on single linked list.

(a) *insert* a node at the beginning of the list.
(b) *insert* a node at the end of the list.
(c) *insert* a node at k^{th} position of the list
(d) *delete* a node from the beginning of the list.
(e) *delete* a node at the end of the list.
(e) *display* the whole list.
(f) *search* an element *x* in the list.
(g) reverse the list.

ASSIGNMENT 7:

Outcome: Students will be able to understand Dynamic memory allocation and different operations of double linked list and Circular Linked list.

Problem Statement: Write a menu driven program in C to perform the following operations on double linked list and circular linked list..

(a) *insert* a node at the k^{th} position of the list.
(b) *delete* the k^{th} node at the end of the list.
(c) *display* the whole list.
(d) *search* an element *x* in the list.

ASSIGNMENT 8:

Outcome: Students will be able to understand Dynamic memory allocation and different operations of double linked list.

Problem Statement: Write a menu driven program in C to implement a *stack and Queue* using single linked list and perform the following operations.

(a) *isEmpty()* is to check whether the stack is empty or not.

(b) *push*() is to insert an item in the stack.

(c) *pop*() is to delete an item from the stack.

(d) *display*() is to show the entire stack.

Problem Statement: Write a menu driven program in C to implement a *queue* using single linked list and perform the following operations.

(a) *isEmpty*() is to check whether the queue is empty or not.

(b) *insert*() is to insert an item in the queue.

(c) *del*() is to delete an item from the queue.

(d) *display*() is to show the entire queue.

ASSIGNMENT 8:

Outcome: Students will be able to understand how a polynomial (single variable) can be stored in a linked list and perform addition of two polynomial single Linked list.

Problem Statement: Write a program in C to add two polynomials using linked list.

ASSIGNMENT 9:

Outcome: Students will be able to understand and apply sorting algorithm to sort list of elements stored in an array and analyze number of swaps, number comparison, and stability properties of sorting algorithms.

Problem Statement:(a) Write a program in C to sort a given array using bubble sort ,selection sort and insertion sort algorithm. Show the number of comparison required for a given input.

ASSIGNMENT 10:

Outcome: Students will be able to understand and apply sorting algorithm to sort list of elements stored in an array and analyze number of swaps, number comparison, and stability properties of sorting algorithms.

Problem Statement :(a) Write a program in C to sort a given array using Quick sort and merge sort algorithm. Show the number of comparison required for a given input.

ASSIGNMENT 11:

Outcome: Students will be able to understand organization of data in BST and its benefits.

Problem Statement : Write a menu driven program in C to perform the following operations on Binary Search Tree.

(a) *insert* a node.

(b) *inorder* traversal.

(c) *preorder* traversal.

(d) *search* an given *key*.

(e) Find the smallest element.

(f) Count the total number of nodes.

ASSIGNMENT 12:

Outcome: Students will be able to apply hash table to store data and understand its benefits and drawbacks.

Problem statement: Write a c program to perform the following operations (use division method for hash function)

(a) Insert an item x into a hash table (Resolve the collision using open Addressing with linear Probing).

(b) Search an element from hash table

ASSIGNMENT 13:

Outcome: Students will be able to understand how to store a graph in computer in the form of adjacency matrix and will be able apply DFS and BFS traversal to search an item. They will also understand how stack and queue are used in DFS and BFS.

Problem statement: Write a c program to perform the following operations

(a) Read the adjacency matrix from a file.

(b) DFS function to traverse the graph.

(c) BFS function to traverse the graph.