**Batch:\_A2 Roll No.:16010122041**

**Experiment No.**

**Grade: AA / AB / BB / BC / CC / CD /DD**

|  |
| --- |
| **Title: Implementation of different operations on Linked List – creation, insertion, deletion, traversal, searching an element** |

**Objective:** To understand the advantage of linked list over other structures like arrays in implementing the general linear list

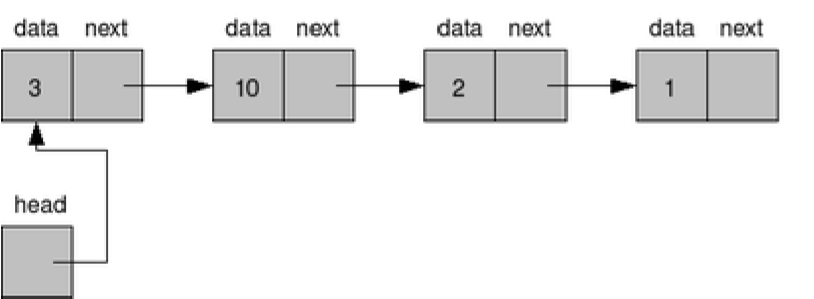
**Expected Outcome of Experiment:**

|  |  |
| --- | --- |
| **CO** | **Outcome** |
| **CO 1** | To understand the advantage of linked list over other structures like arrays in implementing the general linear list |

**Books/ Journals/ Websites referred:**

**Introduction:**

A linear list is a list where each element has a unique successor. There are four common operations associated with linear list: insertion, deletion, retrieval, and traversal. Linear list can be divided into two categories: general list and restricted list. In general list the data can be inserted or deleted without any restriction whereas in restricted list there is restrictions for these operations. Linked list and arrays are commonly used to implement general linear list. A linked list is simply a chain of structures which contain a pointer to the next element. It is dynamic in nature. Items may be added to it or deleted from it at will.



A list item has a pointer to the next element, or to NULL if the current element is the tail (end of the list). This pointer points to a structure of the same type as itself. This Structure that contains elements and pointers to the next structure is called a Node.

**Related Theory: -**

In computer science, a linked list is a linear collection of data elements, whose order is not given by their physical placement in memory. Instead, each element points to the next. It is a data structure consisting of a collection of nodes which together represent a sequence. In its most basic form, each node contains: data, and a reference to the next node in the sequence. This structure allows for efficient insertion or removal of elements from any position in the sequence during iteration.

Like arrays, Linked List is a linear data structure. Unlike arrays, linked list elements are not stored at contiguous location; the elements are linked using pointers

**Linked List ADT:**

*Algorithm for creation, insertion, deletion, traversal and searching an element in assigned linked list type:*

Implementation of an application using linked lists:

#include <stdio.h>

#include <stdlib.h>

struct Node {

    int data;

    struct Node\* next;

};

struct Node\* insertAtBeginning(struct Node\* head, int data) {

    struct Node\* newNode = (struct Node\*)malloc(sizeof(struct Node));

    newNode->data = data;

    newNode->next = head;

    return newNode;

}

struct Node\* deleteNode(struct Node\* head, int key) {

    struct Node \*current = head, \*prev = NULL;

    if (current != NULL && current->data == key) {

        head = current->next;

        free(current);

        return head;

    }

    while (current != NULL && current->data != key) {

        prev = current;

        current = current->next;

    }

    if (current == NULL) {

        printf("Element %d not found in the list.\n", key);

        return head;

    }

    prev->next = current->next;

    free(current);

    return head;

}

void displayList(struct Node\* head) {

    struct Node\* current = head;

    while (current != NULL) {

        printf("%d -> ", current->data);

        current = current->next;

    }

    printf("NULL\n");

}

int main() {

    struct Node\* head = NULL;

    int choice, data;

    choice = data = 0;

    while (choice != 4) {

        printf("1. Insert at the beginning\n");

        printf("2. Delete a node\n");

        printf("3. Display the list\n");

        printf("4. Exit\n");

        printf("Enter your choice: ");

        scanf("%d", &choice);

        switch (choice) {

            case 1:

                printf("Enter data to insert: ");

                scanf("%d", &data);

                head = insertAtBeginning(head, data);

                break;

            case 2:

                printf("Enter data to delete: ");

                scanf("%d", &data);

                head = deleteNode(head, data);

                break;

            case 3:

                printf("Linked List: ");

                displayList(head);

                break;

            case 4:

                exit(0);

            default:

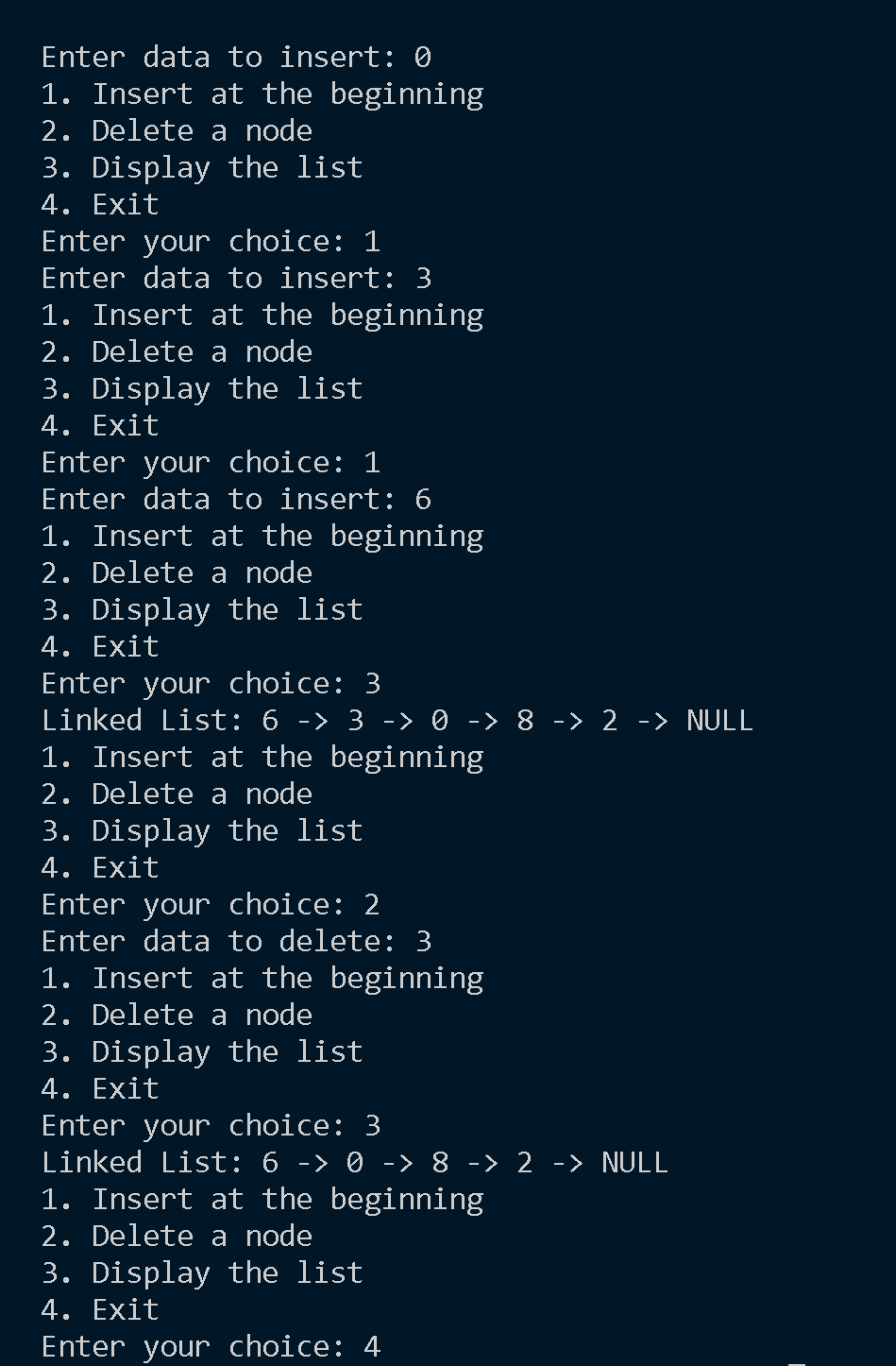
                printf("Invalid choice! Please enter a valid option.\n");

        }

    }

    return 0;

}



**Conclusion:-** Learnt about Linked List, its various operations and its implementation.

**Post lab questions:**

1. Write the differences between linked list and linear array

2. Name some applications which uses linked list.