Lab-5

Q. WAP to Implement Singly Linked List with the following operations
(a) Create a linked list
(b) Insertion of a node at first position, at any position and at end of list
(c) Display the contents of the linked list

```
#include <stdio.h>
#include <stdlib.h>

typedef struct Node{
    int data;
    struct Node* next;
} Node;

typedef struct LinkedList{
    Node* head;
} LinkedList;

Node* createNode(int data){
    Node* newNode = (Node*)malloc(sizeof(Node));
    newNode->data = data;
    newNode->next = NULL;
    return newNode;
}
```

```
LinkedList * createLinkedList (){
    LinkedList *list = (LinkedList *) malloc (sizeof
                        (LinkedList));
    list ->head = NULL;
    return list;
}

void insertAtBeginning (LinkedList *list, int data){
    Node* newNode = createNode (data);
    newNode ->next = list -> head;
    list -> head = newNode;
}

void insertAtPosition (LinkedList *list, int pos, int data
    if (pos <=0){
        printf ("Position should be greater than
                zero \n");
        return;
    }
    Node* newNode = createNode (data);
    Node* curr = list -> head;
    int count = 0;
```

```
    while (cur) {
        if (count == pos-1) {
            newNode -> next = cur -> next;
            cur -> next = newNode;
            return;
        }
        cur = cur -> next;
        count += 1;
    }
    printf("Position out of range\n");
}

void insertAtEnd(LinkedList* list, int data) {
    Node* newNode = createNode(data);
    if (list -> head == NULL) {
        list -> head = newNode;
        return;
    }
    Node* cur = list -> head;
    while (cur -> next) {
        cur = cur -> next;
    }
    cur -> next = newNode;
}
```

```
void display (LinkedList * list){
    Node* curNode = list->head;
    while (curNode){
        printf("%d ->", curNode->data);
        curNode = curNode->next;
    }
    printf("NULL\n");
}

int main(){
    LinkedList *LinkedList = createLinkedList();
    insertAtBeginning(linked list, 1);
    insertAtBeginning(linked list, 2);
    insertAtBeginning(linked list, 3);
    display(linked list);
    insertAtBeginning(linked list, 4);
    display(linked list);
    insertAtPosition(linkedlist, 2, 5);
    display(linked list);
    insertAtEnd(linkedlist, 6);
    display(linked list);
    return 0;
}
```

Output-

3 → 2 → 1 → NULL

4 → 3 → 2 → 1 → NULL

4 → 3 → 5 → 2 → 1 → NULL

4 → 3 → 5 → 2 → 1 → 6 → NULL

Q. WAP to implement singly linked list with the following operations -

(a) Create a linked list.

(b) Deletion of first element, specified element and the last element in the list.

(c) Display the contents of the linked list.

```
#include <stdio.h>
#include <stdlib.h>
// Define the structure for a node in the linked list
struct Node {
    int data;
    struct Node* next;
};
// Function to create a new node with the given data
struct Node* newNode(int data) {
    struct Node* node = (struct node*)malloc
                        (sizeof(struct Node));
    node -> data = data;
    node -> next = NULL;
    return node;
}
```

```
// Function to insert a new node at the beginning
   of the list
void push (struct Node** head, int data){
    struct Node* node = new Node (data);
    node->next = *head;
    *head = node;
}
// Function to delete the first occurance of a key in
   the list
void deleteNode (struct Node** head, int key){
    struct Node* temp = *head, * prev;
// If the head node itself holds the key to be deleted
    if (temp != NULL && temp->data == key){
        *head = temp->next;
        free (temp);
        return;
    }
// Delete the key in the list
    while(temp != NULL && temp->data != key){
        prev = temp;
        temp = temp->next;
    }
```

```
// If the key was not present in the list
    if (temp == NULL)
        return;
// Unlink the node from the list
    prev -> next = temp -> next;
    free (temp);
}
// Function to display the contents of the list
void printList (struct Node * node ){
    while (node != NULL) {
        printf ("%d ->", node -> data);
        node = node -> next;
    }
    printf ("NULL\n");
}
// Function to delete the first node in the list
void deleteFirst (struct Node ** head) {
    if (*head != NULL) {
        struct Node* temp = *head;
        *head = (*head) -> next;
        free (temp);
    }
}
```

```
// Function to delete the last node in the list
void deleteLast(struct Node** head) {
    if (*head == NULL)
        return;
    if ((*head)->next == NULL) {
        free(*head);
        *head = NULL;
        return;
    }
    struct Node* temp = *head;
    while (temp->next->next != NULL)
        temp = temp->next;
    free(temp->next);
    temp->next = NULL;
}
int main() {
    struct Node* head = NULL;
// Insert elements at the beginning of the list
    push(&head, 4);
    push(&head, 3);
    push(&head, 2);
    push(&head, 1);
    printf("Linked list: ");
    printList(head);
```

```
// Delete elements from the list
deleteNode(&head, 2);
printf("After deleting 2: ");
printList(head);
deleteFirst(&head);
printf("After deleting the first element: ");
printList(head);
deleteLast(&head);
printf("After deleting the last element: ");
printList(head);
return 0;
}
```

Output -

Linked list: 1 -> 2 -> 3 -> 4 -> NULL

After deleting 2: 1 -> 3 -> 4 -> NULL

After deleting the first element: 3 -> 4 -> NULL

After deleting the last element: 3 -> NULL

22.01.24