Lab Pgm-5a

5a) WAP to implement singly Linked List with following operations

a) create a linked list.

b) deletion of first element
   Specified element
   last element

c) Display the contents of the linked List

Pseudo code:-

b) void deletefirst():

```
Struct node * temp
if (head==NULL). then
        Print "list is empty"
        return.
END if
temp=head.
 head=head->next
 Print " Deleted element"
 free temp
END deleteFirst
Void delete last() :
    if (head==NULL) then
        Print "Deleted element"
        return.
    END IF
     if (head->next==NULL):
        Print ("Deleted element)
        free (head)
        head=NULL
     Endif
     temp=head
     while(temp->next!=NULL)
          Prev=temp;
          temp=temp->next;
```

```
End deletelast
Void deleteSpecific(int value)
    if(head->data==value) then
        deleteFirst
    End If
    If(head==NULL)
        Print "List is empty"
    End if
    while (temp!=NULL and temp->data!=value)
        Prev=temp
        temp=temp->next
    If (temp==NULL) then "value not found return"
    End if
        Prev->next=temp->next;
        free(temp)
End deleteSpecific();
```

codes-

```
#include <stdio.h>
#include <stdlib.h>
struct node{
    int data;
    struct node *next;
};
struct node *head=NULL;
void createlist(int n){
    struct node *newnode,*temp;
    int data,i;
    if(n<0){
        Printf("n should be greater than 0");
        return;
    }
    for(i=1;i<=n;i++){
        newnode=(struct node*)malloc(sizeof(struct node));
        if(newnode==NULL){
            printf("memory allocation failed!");
            return;
        }
        Printf("enter data:");
        scanf("%d",&data);
        new node->data=data;
        newnode->next=NULL;
        if(head==NULL){
            head=newnode;
        }
        else{
            temp->next=newnode;
        }
        temp=newnode;
    }
    Printf("Linked list is created");
}
```

```
Void deletefirst(){
    struct node *temp;
    if(head==NULL){
        Printf("List is Empty! Nothing to delete");
        return;
    }
    temp=head;
    head=head->next;
    printf("The Deleted element is %d", temp->data);
    free(temp)
}

Void deletelast(){
    struct node *temp, *prev;
    if(head==NULL){
        Printf("The list is empty");
        return;
    }
    if(head->next==NULL){
        Printf("Deleted element: %d", head->data);
        free(head);
        head=NULL;
        return;
    }
    temp=head;
    while(temp->next!=NULL){
        prev=temp;
        temp=temp->next;
    }
    Prev->next=NULL;
    printf("Deleted element: %d", temp->data);
    free(temp);
}
```

```
Void deletespecific(int value){
    struct node *temp=head, *prev=NULL;
    if(head==NULL){
        Printf("The list is empty!");
        return;
    }
    if(head->data==value){
        deletefirst();
        return;
    }
    while(temp!=NULL && temp->data!=value){
        Prev=temp;
        temp=temp->next;
    }
    if(temp==NULL){
        Printf("value not found!");
        return;
    }
    prev->next=temp->next;
    free(temp);
}

void display(){
    struct node *temp=head;
    if(head==NULL){
        printf("List is empty");
    }
    else{
        Printf("Linked list: \n");
        while(temp!=NULL){
            Printf("%d", temp->data);
            temp=temp->data;
        }
    }
}
```

```
int main(){
    int ch, value, n;
    do{
        printf("\n 1)create linked list 2) Delete at
beginning 3) Delete at any position 4) Delete at end
5) Display list");
    printf("enter your choice: ");
    scanf("%d", &ch);
    switch(ch) {
        case 1:
            printf("Enter no. of nodes : ");
            scanf("%d", &n);
            createlist(n);
            break;
        case 2:
            deletefirst();
            break;
        case 3:
            printf("Enter value: ");
            scanf("%d", &value);
            deletespecific(value);
            break;
        case 4:
            deletelast();
            break;
        case 5:
            display();
            break;
        default: printf("Invalid input");
    }
}while(ch!=-1);
}
```

o/p:-

1) Create linked list 2) Delete at beginning 3) Delete at position 4) Delete at last 5) Display list

Enter your choice: 1
Enter no. of nodes: 6
Enter data: 1
Enter data: 2
Enter data: 3
Enter data: 4
Enter data: 5
Enter data: 6
Linked list is created

1) Create linked list 2) Delete at beginning
3) Delete at position 4) Delete at last 5) Display list
Enter your choice: 2
the Deleted element is 1

1) Create linked list 2) Delete at beginning
3) Delete at position 4) Delete at last 5) Display list
enter your choice: 4
Deleted element: 6

1) Create linked list 2) Delete at beginning
3) Delete at position 4) Delete at last 5) Display list
Enter your choice: 3
Enter value: 3

1) Create linked list 2) Delete at beginning 3) Delete at position 4) Delete at last 5) Display list
Enter your choice: 5
Linked list:
2 4 5

1) Create list 2) Delete at beginning 3) Delete at position 4) Delete at last 5) Display list
Enter your choice: -1