WEEK 4

LAB 5

1. Deleting elements from linked list

#include <stdio.h>  
#include <stdlib.h>  
  
void pop();  
void end\_delete();  
void delete\_at\_pos();  
void display();  
void append();  
struct node  
{  
    int data;  
    struct node \*next;  
};  
struct node \*head = NULL;  
  
void main()  
{  
    printf("Insert the elements in list\n");  
    append();  
    printf("1.Delete from beginning\n2.Delete at end\n3.Delete at particular position\n4.Display\n5.Exit\n");  
    int ch;  
    while(ch!=5)  
    {  
        printf("Enter choice:");  
        scanf("%d",&ch);  
        switch(ch)  
        {  
            case 1:  
                pop();  
                break;  
            case 2:  
                end\_delete();  
                break;  
            case 3:  
                delete\_at\_pos();  
                break;  
            case 4:  
                display();  
                break;  
            default: printf("Invalid choice");  
                break;  
        }  
    }  
}  
  
void append()  
{  
    int data,n;  
    printf("Enter no. of nodes:");  
    scanf("%d",&n);  
    for(int i=0;i<n;i++)  
    {  
        struct node \*last=head;  
        struct node \*new\_node;  
        new\_node=(struct node\*)malloc(sizeof(struct node));  
        printf("Enter the data:");  
        scanf("%d",&data);  
        new\_node -> data=data;  
        new\_node -> next =NULL;  
        if(head==NULL)  
            head = new\_node;  
        else  
        {  
        while(last->next!=NULL)  
        {      
            last=last->next;  
        }  
        last-> next =new\_node;  
        }  
    }  
}  
  
  
void pop()  
{  
    struct node \*ptr;  
    if(head == NULL)  
        printf("List is empty\n");  
    else  
    {  
        ptr=head;  
        head = ptr->next;  
        free(ptr);  
        printf("Node deleted from beginning\n");  
    }  
}  
  
void end\_delete()  
{  
    struct node \*ptr;  
    struct node \*ptr1;  
    if(head == NULL)  
        printf("List is empty\n");  
    else if(head -> next ==NULL)  
    {  
        free(head);  
        head = NULL;  
    }  
    else  
    {  
        ptr = head;  
        ptr1 = head;  
        while(ptr -> next !=NULL)  
        {  
            ptr1 = ptr;  
            ptr = ptr->next;  
        }  
        ptr1 -> next =NULL;  
        free(ptr);  
        printf("Node deleted from end\n");  
    }  
}  
  
void delete\_at\_pos()  
{  
    struct node \*ptr;  
    struct node \*ptr1;  
    int pos;  
    printf("Enter the position of deletion:\n");  
    scanf("%d",&pos);  
    ptr = head;  
    for(int i=0;i<pos-1;i++)  
    {  
        ptr1 = ptr;  
        ptr = ptr->next;  
        if(ptr == NULL)  
        {  
            printf("There are less elements in the list\n");  
            return;  
        }  
    }  
    ptr1->next = ptr->next;  
    free(ptr);  
    printf("Node deleted from position %d\n",pos);  
}  
  
void display()  
{  
    struct node \*p= head;  
    printf("List:\n");  
     while(p != NULL)  
    {  
        printf("%d ->",p->data);  
        p=p->next;  
    }  
    printf("NULL\n");  
}

OUTPUT:

