WEEK 5

1.WAP to implement singly linked list operations

#include <stdio.h>

#include <stdlib.h>

struct node

{

int data;

struct node \*next;

};

struct node \*head1=NULL;

struct node \*head2=NULL;

void insert1()

{

int n,i,data;

printf("Enter the no of nodes:");

scanf("%d",&n);

printf("Enter the data:\n");

for(i=0;i<n;i++)

{

struct node \*last1=head1;

struct node \*new\_node1;

new\_node1=(struct node\*)malloc(sizeof(struct node));

scanf("%d",&data);

new\_node1->data=data;

new\_node1->next=NULL;

if(head1==NULL)

{

head1=new\_node1;

}

else

{

while(last1->next!=NULL)

{

last1=last1->next;

}

last1->next=new\_node1;

}

}

}

void insert2()

{

int n,i,data;

printf("Enter the number of nodes:");

scanf("%d",&n);

printf("Enter the data:\n");

for(i=0;i<n;i++)

{

struct node \*last2=head2;

struct node \*new\_node2;

new\_node2=(struct node\*)malloc(sizeof(struct node));

scanf("%d",&data);

new\_node2->data=data;

new\_node2->next=NULL;

if(head2==NULL)

{

head2=new\_node2;

}

else

{

while(last2->next!=NULL)

{

last2=last2->next;

}

last2->next=new\_node2;

}

}

}

void sort()

{

struct node \*curr = head1;

struct node \*ptr = NULL;

int temp;

while(curr != NULL)

{

ptr = curr->next;

while(ptr !=NULL)

{

if(curr->data > ptr->data)

{

temp = curr->data;

curr->data=ptr->data;

ptr->data=temp;

}

ptr = ptr->next;

}

curr = curr->next;

}

display();

}

void reverse()

{

struct node\* prev=NULL;

struct node\* ptr=NULL;

while(head1!=NULL)

{

ptr=head1->next;

head1->next=prev;

prev=head1;

head1=ptr;

}

head1=prev;

display();

}

void concate()

{

struct node\*temp=head1;

if(head1==NULL)

{

struct node \*node=head2;

while(node!=NULL)

{

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

node=node->next;

}

printf("\n");

}

else

{

while(temp->next!=NULL)

{

temp=temp->next;

}

temp->next=head2;

}

display();

}

void display()

{

struct node \*node=head1;

if(head1==NULL)

{

printf("List is empty\n");

}

else

{

while(node!=NULL)

{

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

node=node->next;

}

printf("NULL\n");

}

}

void main()

{

printf("Insert elemnts in first list:\n");

insert1();

printf("Sorted list:\n");

sort();

printf("Reversed list:\n");

reverse();

printf("Insert the elements in second list:\n");

insert2();

printf("Concatenated list:\n");

concate();

}

OUTPUT:

