WEEK 6

1) WAP to Implement Single Link List with following operations

a) Sort the linked list.

b) Reverse the linked list.

c) Concatenation of two linked lists

Program:

```
#include<stdio.h>
#include<stdlib.h>
struct NODE
{
 int data;
 struct NODE *link;
};
typedef struct NODE node;
node *start = NULL,*start1,*start2,*start3;

node* create()
{
 int choice;
 node *new, *curr;
 start = (node*)malloc(sizeof(node));
 curr = start;
 printf("Enter element:\n");
 scanf("%d", &start->data);
 while(1)
 {
```

```
 printf("Do you want to add an element? press 1 for yes\n");
 scanf("%d", &choice);
 if(choice)
 {
 new = (node*)malloc(sizeof(node));
 printf("Please enter element:\n");
 scanf("%d", &new->data);
 curr->link=new;
 curr = new;
 }
 else
 {
 curr->link=NULL;
 break;
 }
 }
 return start;
}
void sort()
{
      int t,n,count=0,i,j;
      node *a,*b,*temp;
      temp=start;
      while(temp!=NULL)
      {
      count++;
      temp=temp->link;
      }
      n=count;
      a=start;
```

```
        b=start->link;
        for(i=0;i<n-1;i++)
        {
        for(j=0;j<n-i-1;j++)
        {
        if(a->data>b->data)
        {
        t=a->data;
        a->data=b->data;
        b->data=t;
        }
        a=b;
        b=b->link;
        }
        a=start;
        b=start->link;
        }
}
void reverse()
{
        node*a=start,*b=NULL,*c=NULL;
        while(a!=NULL)
        {
        c=b;
        b=a;
        a=a->link;
        b->link=c;
        }
        start=b;
}
```

```
void display()
{
 node *temp;
 temp = start;
 if(start==NULL)
 {
 printf("Linked list is empty\n");
 return;
 }
 while(temp!=NULL)
 {
 printf("%d\t", temp->data);
 temp= temp->link;
 }
}

void concatenate(node *start1,node *start2)
{
      node *temp;
      if(start1==NULL)
      {
      start=start2;
      return;
      }
      if(start2==NULL)
      {
      start=start1;
      return;
      }
```

```
      else
      {
      temp=start1;
      while(temp->link!=NULL)
      {
      temp=temp->link;
      }
      temp->link=start2;
      start=start1;
      }
}

void main()
{
    int choice,c1,c2;
    while(1)
    {
        printf("\n1.Create 2.Sort 3.Reverse 4.Concatenate 5.Display
6.Exit\n");
        printf("Enter choice:");
        scanf("%d", &choice);
        switch(choice)
        {
            case 1: create();
            break;
            case 2: sort();
            break;
            case 3: reverse();
            break;
            case 4: printf("Do ypu want to create the first linked list if
```

```
yes press 1\n");
            scanf("%d",&c1);
            if(c1==1)
            start1=create();
            else
            start1=NULL;
            printf("Do ypu want to create the second linked list if yes
press 2\n");
            scanf("%d",&c2);
            if(c2==2)
            start2=create();
            else
            start2=NULL;
            concatenate(start1,start2);
            break;
            case 5:display();
            break;
            case 6:exit(0);
            break;
            default: printf("Invalid choice\n");
        }
    }
}
```

Output:

```
1.Create 2.Sort 3.Reverse 4.Concatenate 5.Display 6.Exit
Enter choice:1
Enter element:
72
Do you want to add an element? press 1 for yes
1
Please enter element:
36
Do you want to add an element? press 1 for yes
1
Please enter element:
46
Do you want to add an element? press 1 for yes
1
Please enter element:
23
Do you want to add an element? press 1 for yes
1
Please enter element:
85
Do you want to add an element? press 1 for yes
1
Please enter element:
9
Do you want to add an element? press 1 for yes
1
Please enter element:
67
Do you want to add an element? press 1 for yes
0

1.Create 2.Sort 3.Reverse 4.Concatenate 5.Display 6.Exit
Enter choice:5
72      36      46      23      85      9       67
1.Create 2.Sort 3.Reverse 4.Concatenate 5.Display 6.Exit
Enter choice:2

1.Create 2.Sort 3.Reverse 4.Concatenate 5.Display 6.Exit
Enter choice:5
9       23      36      46      67      72      85
1.Create 2.Sort 3.Reverse 4.Concatenate 5.Display 6.Exit
Enter choice:3

1.Create 2.Sort 3.Reverse 4.Concatenate 5.Display 6.Exit
Enter choice:5
85      72      67      46      36      23      9
1.Create 2.Sort 3.Reverse 4.Concatenate 5.Display 6.Exit
Enter choice:6
```

2)WAP to implement Stack & Queues using Linked Representation

Program:

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

struct NODE
{
    int data;
    struct NODE *link;
};
typedef struct NODE node;
node *front=NULL,*rear=NULL,*new=NULL;

void disp()
{
    node *temp;
    if(front==NULL)
    {
        printf("Empty");
        return;
    }
    temp=front;
    while(temp!=NULL)
    {
        printf("%d\t",temp->data);
        temp=temp->link;
    }
}
```

```
void ins_beg()
{
    new=(node*)malloc(sizeof(node));
    printf("Enter element:");
    scanf("%d",&new->data);
    if(front==NULL)
    {
        front=new;
        rear=new;
        new->link=NULL;
        return;
    }
    new->link=front;
    front=new;
}

void ins_end()
{
    node *temp;
    temp=rear;
    new=(node*)malloc(sizeof(node));
    printf("Enter element:");
    scanf("%d",&new->data);
    if(front==NULL)
    {
        front=new;
        rear=new;
        new->link=NULL;
        return;
```

```
    }
    new->link=NULL;
    temp->link=new;
    temp=temp->link;
    rear=temp;
}

void del_beg()
{
    node *temp;
    temp=front;
    if(front==NULL)
    {
        printf("Empty");
        return;
    }
    front=front->link;
    free(temp);
}

void stackop()
{
    int c1;
    while(1)
    {
        printf("\n1.Push 2.Pop 3.Display 4.Exit");
        printf("\nEnter your choice:");
        scanf("%d",&c1);
        switch(c1)
        {
```

```
            case 1:ins_beg();
                   break;
            case 2:del_beg();
                   break;
            case 3:disp();
                   break;
            case 4:exit(0);
                   break;
            default:printf("Wrong choice!");
        }
    }
}

void queueop()
{
    int c1;
    while(1)
    {
        printf("\n1.Insert 2.Delete 3.Display 4.Exit");
        printf("\nEnter your choice:");
        scanf("%d",&c1);
        switch(c1)
        {
            case 1:ins_end();
                   break;
            case 2:del_beg();
                   break;
            case 3:disp();
                   break;
            case 4:exit(0);
```

```
                    break;
            default:printf("Wrong choice!");
        }
    }
}

void main()
{
    int c1;
    printf("1.Stack 2.Queue");
    printf("\nEnter your choice:");
    scanf("%d",&c1);
    switch(c1)
    {
        case 1:stackop();
               break;
        case 2:queueop();
               break;
        default:printf("Wrong Choice!");
    }
}
```

Output:

```
1.Stack 2.Queue
Enter your choice:1

1.Push 2.Pop 3.Display 4.Exit
Enter your choice:1
Enter element:10

1.Push 2.Pop 3.Display 4.Exit
Enter your choice:1
Enter element:20

1.Push 2.Pop 3.Display 4.Exit
Enter your choice:1
Enter element:30

1.Push 2.Pop 3.Display 4.Exit
Enter your choice:3
30      20      10
1.Push 2.Pop 3.Display 4.Exit
Enter your choice:2

1.Push 2.Pop 3.Display 4.Exit
Enter your choice:3
20      10
1.Push 2.Pop 3.Display 4.Exit
Enter your choice:2

1.Push 2.Pop 3.Display 4.Exit
Enter your choice:3
10
1.Push 2.Pop 3.Display 4.Exit
Enter your choice:2

1.Push 2.Pop 3.Display 4.Exit
Enter your choice:3
Empty
1.Push 2.Pop 3.Display 4.Exit
Enter your choice:4
```

```
1.Stack 2.Queue
Enter your choice:2

1.Insert 2.Delete 3.Display 4.Exit
Enter your choice:1
Enter element:10

1.Insert 2.Delete 3.Display 4.Exit
Enter your choice:1
Enter element:20

1.Insert 2.Delete 3.Display 4.Exit
Enter your choice:1
Enter element:30

1.Insert 2.Delete 3.Display 4.Exit
Enter your choice:1
Enter element:40

1.Insert 2.Delete 3.Display 4.Exit
Enter your choice:3
10      20      30      40
1.Insert 2.Delete 3.Display 4.Exit
Enter your choice:2

1.Insert 2.Delete 3.Display 4.Exit
Enter your choice:3
20      30      40
1.Insert 2.Delete 3.Display 4.Exit
Enter your choice:2

1.Insert 2.Delete 3.Display 4.Exit
Enter your choice:3
30      40
1.Insert 2.Delete 3.Display 4.Exit
Enter your choice:2

1.Insert 2.Delete 3.Display 4.Exit
Enter your choice:3
40
1.Insert 2.Delete 3.Display 4.Exit
Enter your choice:2

1.Insert 2.Delete 3.Display 4.Exit
Enter your choice:3
Empty
1.Insert 2.Delete 3.Display 4.Exit
Enter your choice:4
```

3) WAP to Implement doubly link list with primitive operations

a) Create a doubly linked list.

b) Insert a new node to the left of the node.

c) Delete the node based on a specific value.

d) Display the contents of the list.

Program:

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

struct NODE
{
    struct NODE *llink;
    int data;
    struct NODE *rlink;
};
typedef struct NODE node;
node *start=NULL,*curr,*new,*temp;

void create()
{
    start=(node*)malloc(sizeof(node));
    printf("Enter element:");
    scanf("%d",&start->data);
    start->llink=NULL;
    curr=start;
    while(1)
```

```
    {
        int choice;
        printf("Do you want to add an element? press 1 for yes\n");
        scanf("%d", &choice);
        if(choice!=0)
        {
            new = (node*)malloc(sizeof(node));
            curr->rlink=new;
            new->llink=curr;
            printf("Enter the element:");
            scanf("%d",&new->data);
            curr=new;
        }
        else
        {
            curr->rlink=NULL;
            break;
        }
    }
}

void insert_beg()
{
    new=(node*)malloc(sizeof(node));
    printf("Enter an element:");
    scanf("%d",&new->data);
    if(start==NULL)
    {
        new->llink=NULL;
        new->rlink=NULL;
```

```
        start=new;
        return;
    }
    new->rlink=start;
    start->llink=new;
    new->llink=NULL;
    start=new;
}

void delete_ele()
{
    node *temp;
    int ele;
    if(start==NULL)
    {
        printf("Linked list is empty\n");
        return;
    }
    printf("Enter element to be deleted:");
    scanf("%d",&ele);
    if(start->data==ele)
    {
        temp=start;
        start=start->rlink;
        start->llink=NULL;
        free(temp);
        return;
    }
    temp=start;
    while(temp->rlink!=NULL&&temp->data!=ele)
```

```
    {
        temp=temp->rlink;
    }
    if(temp->data==ele&&temp->rlink==NULL)
    {
        temp->llink->rlink=NULL;
        free(temp);
        return;
    }
    if(temp->data==ele&&temp->rlink!=NULL)
    {
        temp->llink->rlink=temp->rlink;
        temp->rlink->llink=temp->llink;
        free(temp);
        return;
    }
    printf("Element not found\n");
}

void display()
{
    if(start==NULL)
    {
        printf("Linked list is empty\n");
        return;
    }
    temp=start;
    while(temp!=NULL)
    {
        printf("%d\t",temp->data);
```

```
        temp=temp->rlink;
    }
}


void main()
{
    int choice;
    printf("1.CREATE\n2.INSERT AT BEGINING\n3.DELETE SPECIFIC
ELEMENT\n4.DISPLAY\n5.EXIT\n");
    while(1)
    {
        printf("Enter choice:\n");
        scanf("%d", &choice);
        switch(choice)
        {
            case 1: create();
            break;
            case 2: insert_beg();
            break;
            case 3: delete_ele();
            break;
            case 4:display();
            break;
            case 5:exit(0);
            break;
            default:printf("Invalid choice\n");
        }
    }
    getch();
}
```

Output:

```
1.CREATE
2.INSERT AT BEGINING
3.DELETE SPECIFIC ELEMENT
4.DISPLAY
5.EXIT
Enter choice:
1
Enter element:10
Do you want to add an element? press 1 for yes
1
Enter the element:20
Do you want to add an element? press 1 for yes
1
Enter the element:30
Do you want to add an element? press 1 for yes
1
Enter the element:40
Do you want to add an element? press 1 for yes
0
Enter choice:
4
10      20      30      40      Enter choice:
2
Enter an element:5
Enter choice:
4
5       10      20      30      40      Enter choice:
3
Enter element to be deleted:20
Enter choice:
4
5       10      30      40      Enter choice:
5
```