# VISVESVARAYA TECHNOLOGICAL UNIVERSITY

**"JnanaSangama", Belgaum -590014, Karnataka.**

**LAB REPORT**
**on**

# DATA STRUCTURES

***Submitted by***
**AMULYA S A(1BM21CS020)**

***in partial fulfillment for the award of the degree of***
**BACHELOR OF ENGINEERING**
***in***
**COMPUTER SCIENCE AND ENGINEERING**

**B.M.S. COLLEGE OF ENGINEERING**
**(Autonomous Institution under VTU)**
**BENGALURU-560019**
**Oct 2022-Feb 2023**

# B. M. S. College of Engineering,

**Bull Temple Road, Bangalore 560019**

(Affiliated To Visvesvaraya Technological University, Belgaum)

## Department of Computer Science and Engineering

## CERTIFICATE

This is to certify that the Lab work entitled "**DATA STRUCTURES**" carried out by **AMULYA S A(1BM21CS020),** who is bonafide student of **B. M. S. College of Engineering.** It is in partial fulfillment for the award of **Bachelor of Engineering in Computer Science and Engineering** of the Visvesvaraya Technological University, Belgaum during the year 2022-23. The Lab report has been approved as it satisfies the academic requirements in respect of Data structures Lab **- (22CS3PCDST)** work prescribed for the said degree.

LOHITH J J
Designation
Department of CSE
BMSCE, Bengaluru

**Dr. Jyothi S Nayak**
Professor and Head
Department of CSE
BMSCE, Bengaluru

`

# Index Sheet

| | | |
|---|---|---|
| 7 | WAP to Implement Single Link List with following operations<br>a) Sort the linked list.<br>b) Reverse the linked list.<br>c) Concatenation of two linked lists | 38-51 |
| 8 | WAP to implement Stack & Queues using Linked Representation. | 51-60 |
| 9 | WAP to Implement doubly link list with primitive operations<br>a) Create a doubly linked list.<br>b) Insert a new node to the left of the node.<br>c) Delete the node based on a specific value<br>d) Display the contents of the list | 60-67 |

## Course Outcome

| | |
|---|---|
| **CO1** | Apply the concept of linear and nonlinear data structure. |
| **CO2** | Analyse data structure operations for a given problem. |
| **CO3** | Design and develop solutions using the operations of linear and nonlinear data structure for a given specification |
| **CO4** | Conduct practical experiments for demonstrating the operations of different data structures. |

## LAB PROGRAM 1:

Write a program to simulate the working of stack using an array with the following:
a) Push
b) Pop
c) Display
The program should print appropriate messages for stack overflow, stack underflow.

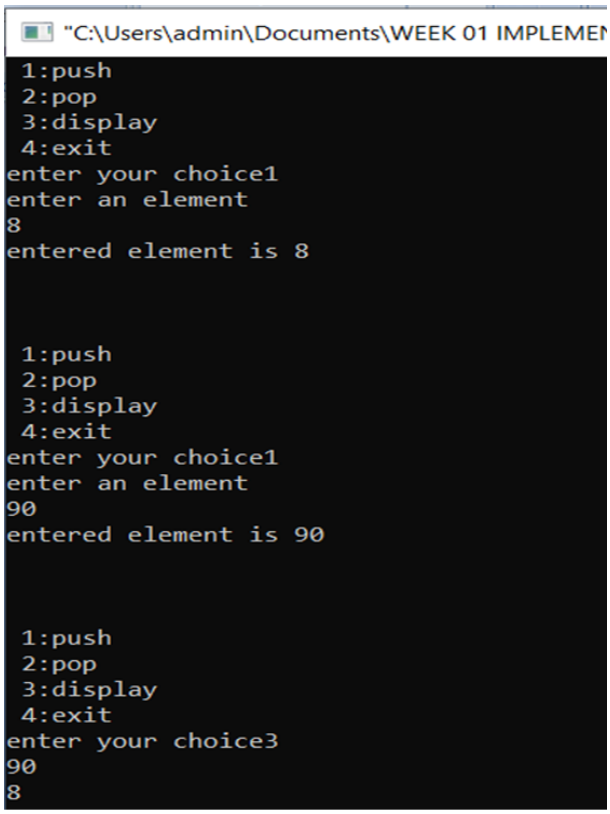
```
#include<stdio.h>

#include<conio.h>

#define SIZE 3

int STACK[SIZE],TOP=-1,ITEM;

void push();

void pop();

void display();

void main()

{

 int choice;

while(1)

          {

      printf("\n\n 1:push\n 2:pop\n 3:display\n 4:exit\n");

      printf("enter your choice");

      scanf("%d",&choice);

      switch(choice)

          {

           case 1:push();

          break;

           case 2: pop();
```

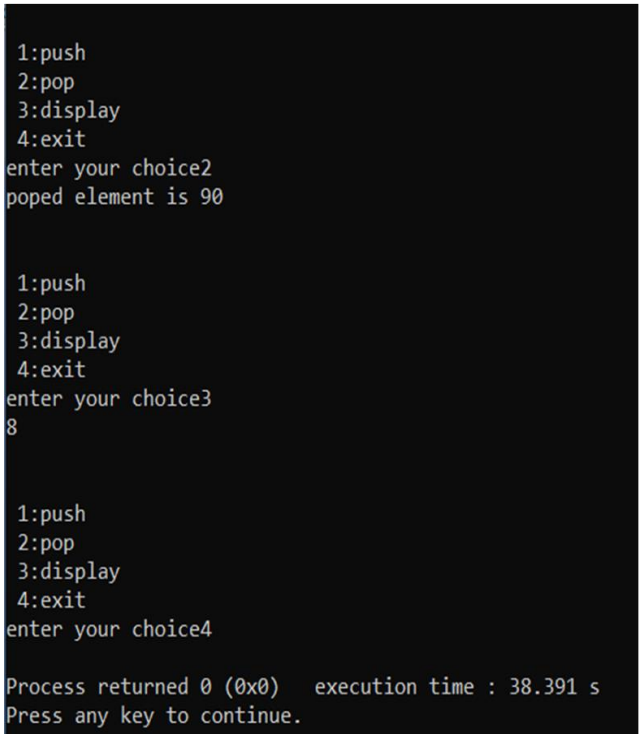
```
            break;
             case 3: display();
            break;
              case 4: exit(0);
            break;
            default: printf("wrong choice");
            }
             }
             getch();
}
void push()
{
   if(TOP==SIZE-1)
             {
       printf("stack overflow");
             return;
             }
             else
             {
       printf("enter an element\n");
       scanf("%d",&ITEM);
       printf("entered element is %d\n\n",ITEM);
       TOP=TOP+1;
       STACK[TOP]=ITEM;
```

```
        }
}
void pop()
{
        int del;
        if(TOP==-1)
        {
    printf("stack underflow\n");
        return;
        }
        else
        {
    del=STACK[TOP];
    printf("poped element is %d\n",del);
    TOP=TOP-1;
        }
}
void display()
{
int i;
if(TOP==-1)
{
printf("STACK IS EMPTY\n");
return;
```

```
}
else
{
   for(i=TOP;i>=0;i--)
         {
      printf("%d\n",STACK[i]);
         }
}
}
```

OUTPUT:






## LAB PROGRAM 2:

WAP to convert a given valid parenthesized infix arithmetic expression to postfix expression. The expression consists of single character operands and the binary operators + (plus), - (minus), * (multiply) and / (divide)

```
#include<stdio.h>

#include<conio.h>

#include<string.h>

int index=0,pos=0,top=-1,length;

char symbol,temp,infix[20],postfix[20],stack[20];

voidinfix_postfix();

void push(char);

char pop();

intpred(char symbol);

void main()
```

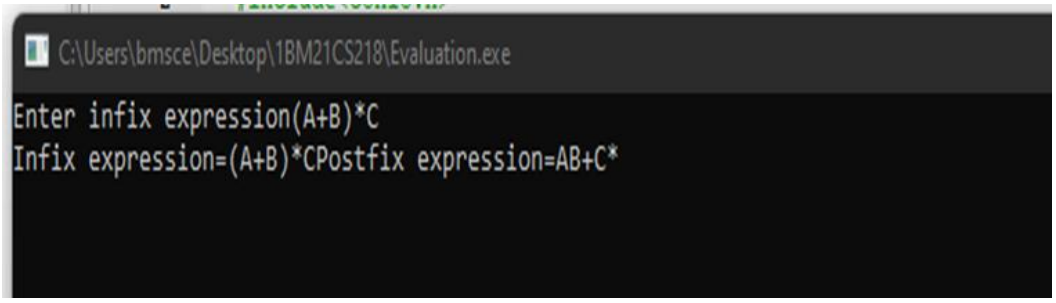
```
{


printf("Enter infix expression");
scanf("%s",infix);
infix_postfix();
printf("Infix expression=%s",infix);
printf("Postfix expression=%s",postfix);
getch();


}
voidinfix_postfix()
{
 length=strlen(infix);
push('#');
while(index<length)
{
symbol=infix[index];
switch(symbol)
{
case '(':push(symbol);
break;
case')':temp=pop();
while(temp!='(')
```

```
{
postfix[pos]=temp;
pos++;
temp=pop();
 }
break;
case '+':
case '-':
case '*':
case '/':
case '^':while(pred(stack[top])>=pred(symbol))
          {
           temp=pop();
           postfix[pos++]=temp;
          }
push(symbol);
break;
default:postfix[pos++]=symbol;
 }
index++;
}
while(top>0)
          {
```

```
temp=pop();
postfix[pos++]=temp;
        }
}
void push(char symbol)
{
top=top+1;
stack[top]=symbol;
}
char pop()
{
char symb;
symb=stack[top];
top=top-1;
return(symb);
}
int(pred(char symbol))
{
int p;
switch(symbol)
{
case '^':p=3;
break;
```

```
case '*':
case '/':p=2;
break;
case '+':
case '-':p=1;
break;
case '(':p=0;
break;
case '#':p=-1;
break;
 }
return(p);
}
```

OUTPUT:




## LAB PROGRAM 3:

3.WAP to simulate the working of a queue of integers using an array. Provide the following operations

a) Insert

b) Delete

c) Display

The program should print appropriate messages for queue empty and queue overflow conditions.

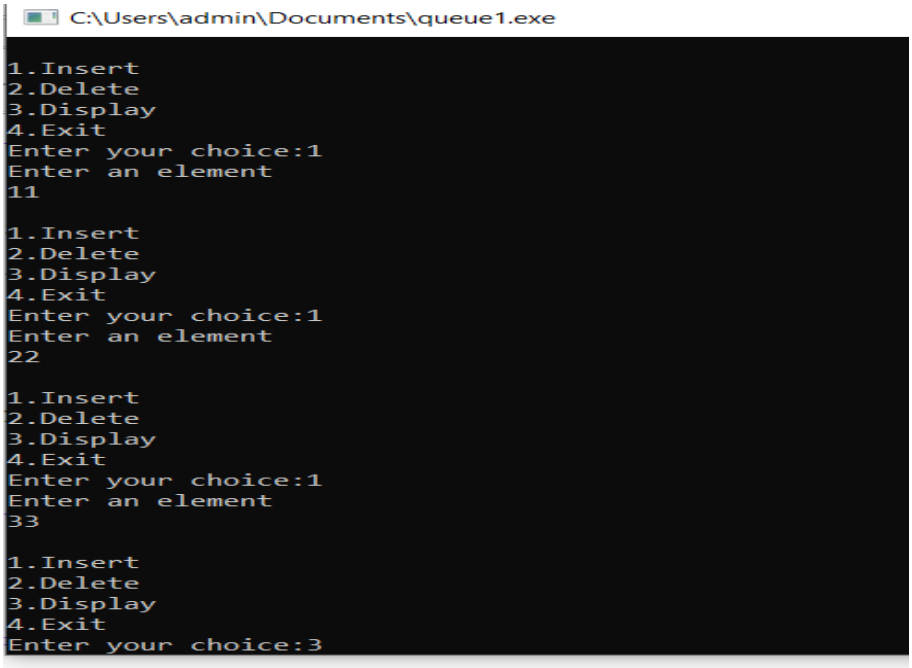
```
#include<stdio.h>

#include<conio.h>

#define size 3

int queue[size],rear=-1,front=0,item;

void insert();

void  delete();

void display();

void main()

{

int choice;

while(1)

        {


printf("\n1.Insert\n2.Delete\n3.Display \n4.Exit\n");

printf("Enter your choice:");

scanf("%d",&choice);

switch(choice)

        {

case 1: insert();

break;

case 2: delete();
```

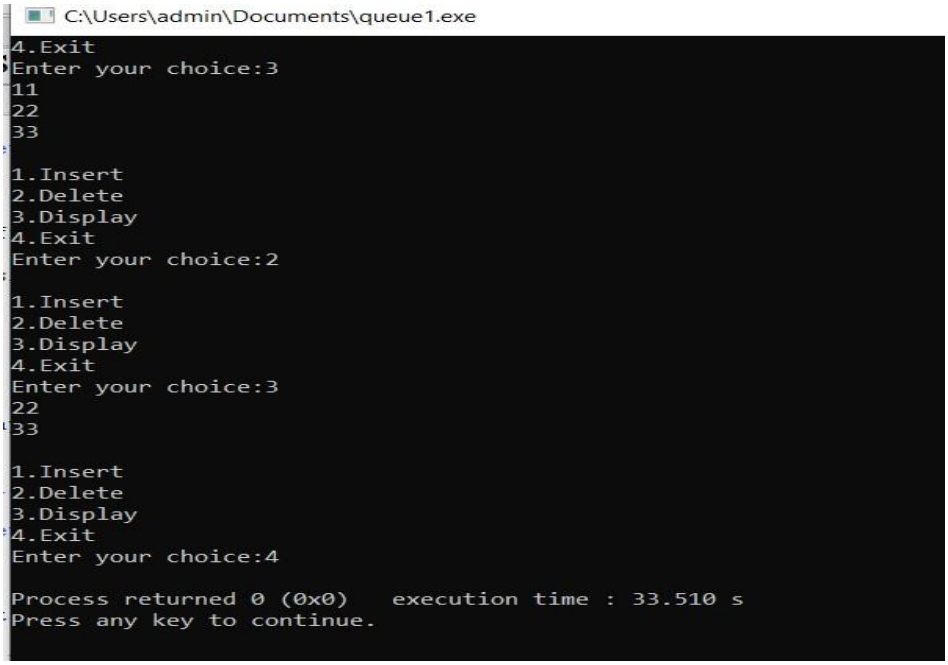
```
break;
case 3: display();
break;
case 4: exit(0);
break;
default: printf("WRONG CHOICE\n");
        }
        }
getch();
}
void insert()
{
if(rear==size-1)
        {
printf("Queue is full");
        }
else
        {
printf("Enter an element\n");
scanf("%d",&item);
rear++;
queue[rear]=item;
```

```
        }
}
void delete()
{
int del;
if(rear==-1)
        {
printf("Queue is empty");
        }
else
        {
del=queue[front];
front++;
if(front==size)
        {
front=0;
rear=-1;
        }
        }
}
void display()
{
inti;
```

```
if(rear==-1)
        {
printf("Queue is empty");
        }
else
        {
for(i=front;i<=rear;i++)
        {
printf("%d",queue[i]);
        }
        }
}
```

OUTPUT:






## LAB PROGRAM 4:

WAP to simulate the working of a circular queue of integers using an array. Provide the following operations.

a) Insert

b) Delete

c) Display

The program should print appropriate messages for queue empty and queue overflow conditions.

```
#include<stdio.h>

#include<conio.h>

#define size 3


int Queue[size],front=-1,rear=-1,item;

void insert();
```

```
void  delete();
void display();
void main()
{
intch;
while(1)
        {
printf("1:Insert\n2:Delete\n3:Display\n4.Exit\n");
printf("Enter your choice:");
scanf("%d",&ch);
switch(ch)
        {
case 1: insert();
break;
case 2: delete();
break;
case 3: display();
break;
case 4: exit(0);
break;
default: printf("WRONG CHOICE\n");
        }
getch();
```

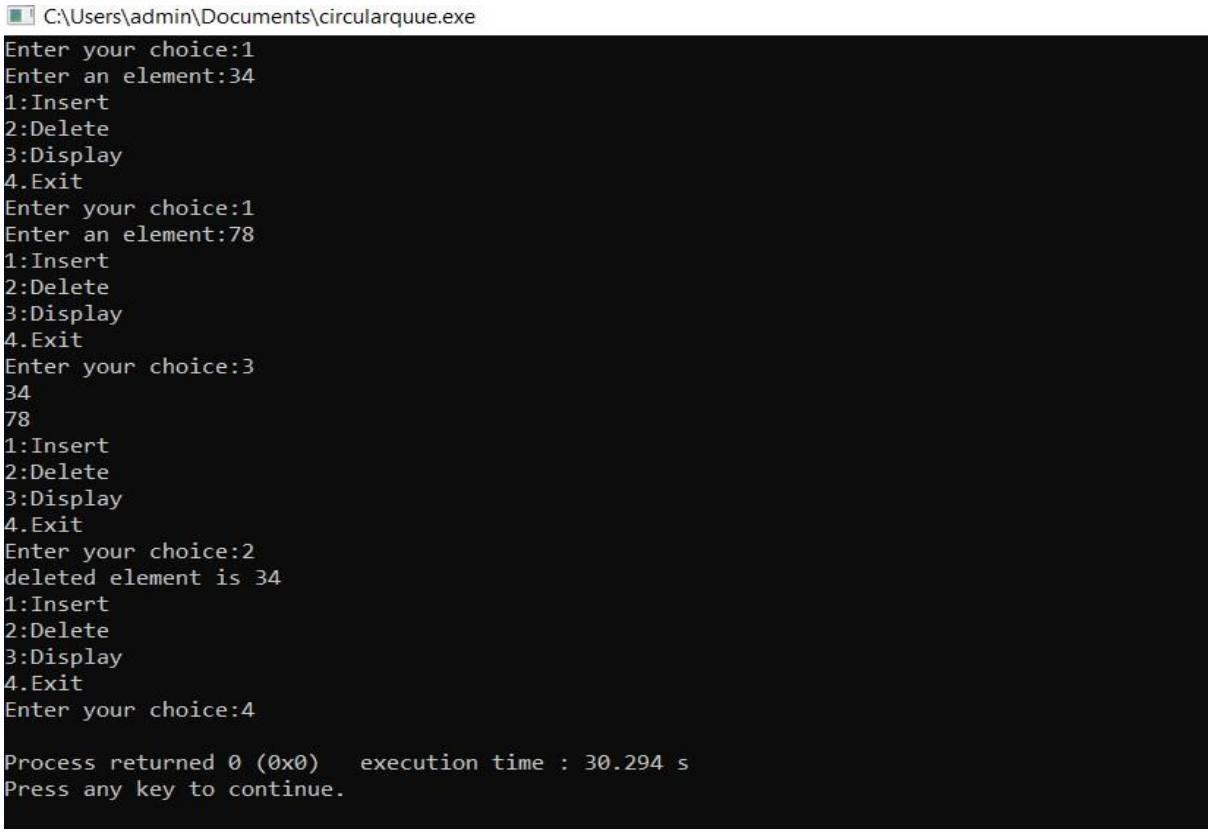
```
        }
}
void insert()
{

if(((front==0)&&(rear==size-1))||(front==rear+1))
        {
printf("Queue is full\n");
        }
else
        {
printf("Enter an element:");
scanf("%d",&item);
if(front==-1&&rear==-1)
        {
front=0;
rear=0;
        }
else
         {
rear=(rear+1)%size;
        }
Queue[rear]=item;
```

```
        }

return;
}
void delete()
{
int del;
if ((front==-1)&&(rear==-1))
        {
printf("Queue is empty\n");
        }
else
        {
del=Queue[front];
printf("deleted element is %d\n",del);
if(front==rear)
        {
front=-1;
rear=-1;
        }
else
        {
front=(front+1)%size;
```

```
        }
        }

return;
}
void display()
{
inti;
if ((front==-1)&&(rear==-1))
        {
printf("Queue is empty\n");
        }
else
        {
if(front<=rear)
        {
for(i=front;i<=rear;i++)
printf("%d\n",Queue[i]);
        }
else
        {
for(i=front;i<=size-1;i++)
        {
```

```
printf("%d\n",Queue[i]);
        }
for(i=0;i<=rear;i++)
        {
printf("%d\n",Queue[i]);
        }
        }
        }
return;
}
```

OUTPUT:



## LAB PROGRAM 5:

WAP to Implement Singly Linked List with 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<conio.h>
#include<stdlib.h>
struct NODE
{
   int data;
   struct NODE *link;
};
typedef struct NODE node;
node*start=NULL,*new,*curr,*temp;
void create();
void display();
void insert_beg();
void insert_end();
void insert_pos();
void main()
{
   int choice;
```

```
   while(1)
      printf("1:Create\\n 2:Display\\n 3:insert beginning\\n 4:insert end\\n 5:insert at required
position\\n 6:Exit\\n");
      printf("Enter your choice:");
      scanf("%d",&choice);
      switch(choice)
         case 1:create();
         break;
         case 2:display();
         break;
         case 3:insert_beg();
         break;
         case 4:insert_end();
         break;
         case 5:insert_pos();
         break;
         case 6:exit(0);
         break;
         default:printf("Invalid choice");
}
        getch();
}
void create()
   int ch;
```

```
    start=(node*)malloc(sizeof(node));
    curr=start;
    printf("Enter an element:");
    scanf("%d",&start->data);
    while(1)
    {
       printf("Do you want to add an other element(1/2)1 for yes any other number for no");
       scanf("%d",&ch);
       if(ch==1)
       {
          new=(node*)malloc(sizeof(node));
          printf("Enter an element:");
          scanf("%d",&new->data);
          curr->link=new;
          curr=new;
       }
       else
          curr->link=NULL;
          break;
void display()
{
    node *temp;
    if(start==NULL)
```

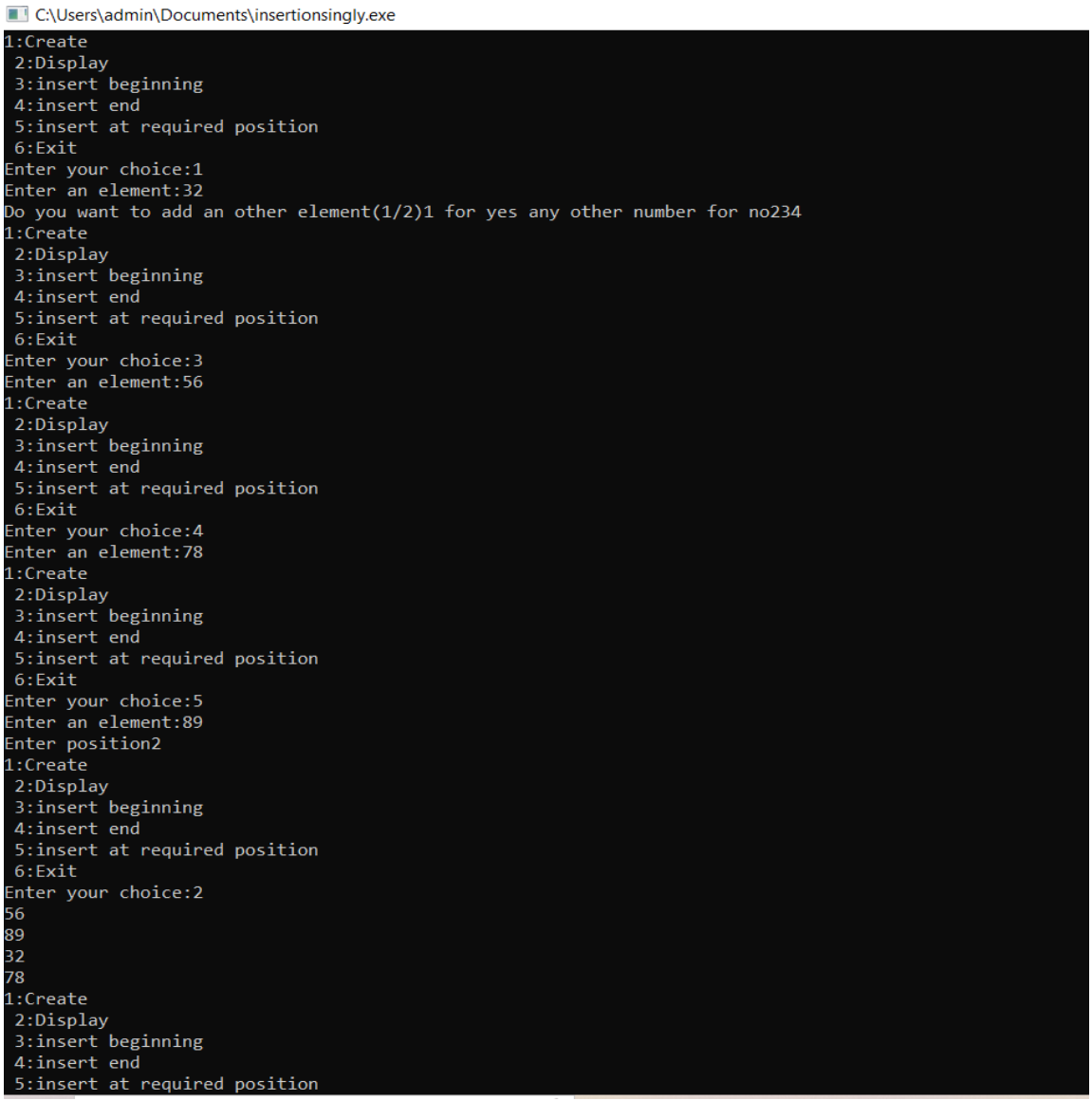
```
    {
        printf("Linked list is empty");
        return;
    }
    temp=start;
    while(temp!=NULL)
    {
        printf("%d\\n",temp->data);
        temp=temp->link;
    }
}
void insert_beg()
{
    new=(node*)malloc(sizeof(node));
    printf("Enter an element:");
    scanf("%d",&new->data);
    if(start==NULL)
    {
      start=new;
      new->link=NULL;
      return;
    }
    new->link=start;
```

```
    start=new;
}
void insert_end()
{
    new=(node*)malloc(sizeof(node));
    printf("Enter an element:");
    scanf("%d",&new->data);
    if(start==NULL)
    {
      start=new;
      new->link=NULL;
      return;
    }
    temp=start;
    while(temp->link!=NULL)
    {
       temp=temp->link;
    }
    temp->link=new;
    new->link=NULL;
}
void insert_pos()
{
```

```
int pos,i=1;
new=(node*)malloc(sizeof(node));
printf("Enter an element:");
scanf("%d",&new->data);
printf("Enter position");
scanf("%d",&pos);
if(pos==1)
{
   new->link=start;
   start=new;
   return;
}
temp=start;
while(i<(pos-1)&&temp!=NULL)
{
   temp=temp->link;
   i++;
}
if(i==(pos-1))
{
   new->link=temp->link;
   temp->link=new;
   return;
```

```
    }
    if(temp==NULL)
    {
        printf("Invalid position\\n");
        return;
}
```

OUTPUT:



## LAB PROGRAM 6:

WAP to Implement Singly Linked List with following operations

a) Create a linked list.

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

c) Display the contents of the linked list.

```
#include<stdio.h>
#include<conio.h>
#include<stdlib.h>
struct NODE
{
   int data;
   struct NODE *link;
};
typedef struct NODE node;
node*start=NULL,*new,*curr,*temp,*next,*prev;
void create();
void display();
void delete_beg();
void delete_end();
void delete_pos();
void main()
{
   int choice;
```

```
   while(1)
   {
      printf("1:Create\\n 2:Display\\n 3:delete beginning\\n 4:delete end\\n 5:delete at required
position\\n 6:Exit\\n");
      printf("Enter your choice:");
      scanf("%d",&choice);
      switch(choice)
      {
         case 1:create();
         break;
         case 2:display();
         break;
         case 3:delete_beg();
         break;
         case 4:delete_end();
         break;
         case 5:delete_pos();
         break;
         case 6:exit(0);
         break;
         default:printf("Invalid choice");
      }
   }
   getch();
```

```
}
void create()
{
   int ch;
   start=(node*)malloc(sizeof(node));
   curr=start;
   printf("Enter an element:");
   scanf("%d",&start->data);
   while(1)
   {
      printf("Do you want to add an other element(1/2)1 for yes any other number for no");
      scanf("%d",&ch);\par
      if(ch==1)\par
      {
         new=(node*)malloc(sizeof(node));
         printf("Enter an element:");
         scanf("%d",&new->data);
         curr->link=new;
         curr=new;
      }
      else
      {
         curr->link=NULL;
```

```
            break;\par
        }


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


}
void delete_beg()
{
```

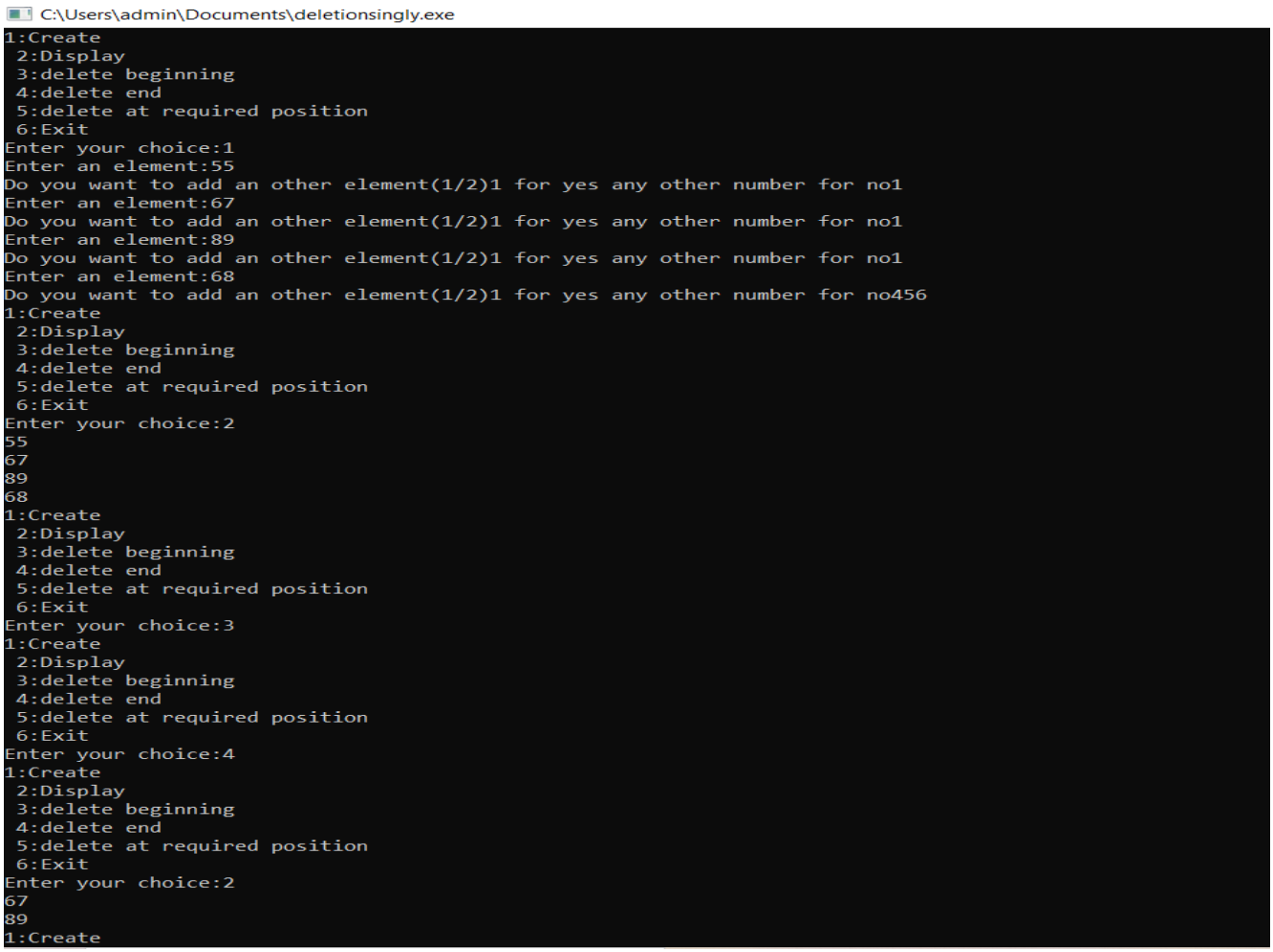
```
    if(start==NULL)
    {
        printf("Linked list is empty");
        return;
    }
    temp=start;
    start=start->link;
    free(temp);

}
void delete_end()
{
    prev=(node*)malloc(sizeof(node));
    next=(node*)malloc(sizeof(node));
    if(start==NULL)
    {
      start=new;
      new->link=NULL;
      return;
    }
     if(start->link==NULL)
     {
        free(start);
```

```
        start==NULL;
        return;
      }
      prev=start;
      next=start->link;
      while(next->link!=NULL)
      {
         prev=next;
         next=next->link;
      }
      prev->link=NULL;free(next);

}
void delete_pos()
{
    int ele;
    prev=(node*)malloc(sizeof(node));
    next=(node*)malloc(sizeof(node));
    if(start==NULL)
    {
       start=new;
       new->link=NULL;
       return;
```

```
  }
  printf("Enter an element to be deleted");
  scanf("%d",&ele);
  if(start->data==ele)
  {

     free(start);
     start=NULL;
     printf("Element deleted");
     return;

  }
   prev=start;
  next=start->link;
while(next->data!=ele && new!= NULL)
  {
     prev=next;
     next=next->link;
  }
  if(next->data==ele)
 {      prev->link=next->link;
    free(next);
     return;
```

```
  }

   printf("Element not found");

}
```

OUTPUT:




## LAB PROGRAM 7:

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

```
#include<stdio.h>
#include<stdlib.h>
struct NODE{
        int data;
        struct NODE *link;
};
typedef struct NODE node;
node *start1=NULL,*start2=NULL,*start3=NULL,*new,*curr,*temp=NULL;
void create_list1();
void create_list2();
void sort();
void search();
void concatenate();
void reverse();
void main()
{
        int choice;
        while(1)
        {

    printf("1.Createlist1 \n 2.Createlist2 \n 3.sort \n 4.search \n 5.concatenate \n 6.Reverse \n
7.Exit \n");
    printf("Enter the choice:");
    scanf("%d",&choice);
```

```
        switch(choice)
        {

        case 1: create_list1();
            break;
        case 2: create_list2();
            break;
        case 3: sort();
            break;
        case 4: search();
            break;
        case 5: concatenate();
            break;
        case 6:reverse();
            break;
        case 7: exit(0);
            break;
       default:printf("Wrong Choice");
        }
        }
        getch();
}
void create_list1()
```

```
{

        int ch;

        start1=(node*) malloc(sizeof(node));

        curr=start1;

        printf("Enter an element");

   scanf("%d",&start1->data);

        while(1)

        {

     printf("Do you want to enter a new element (1 for yes,any other number for no)");

     scanf("%d",&ch);

        if(ch==1)

        {

        new=(node*) malloc(sizeof(node));

        printf("Enter an element");

        scanf("%d",&new->data);

        curr->link=new;

        curr=new;


        }

        else

        {

       curr->link=NULL;
```

```
            break;
            }


            }
            temp=start1;
            printf("the elements in the list 1 are:\n");
   while(temp!=NULL)
            {
        printf("%d\n",temp->data);
        temp=temp->link;
            }



}
void create_list2()
{


            int ch;
            start2=(node*) malloc(sizeof(node));
            curr=start2;
            printf("Enter an element");
   scanf("%d",&start2->data);
            while(1)
```

```
    {
printf("Do you want to enter a new element (1 for yes,any other number for no)");
scanf("%d",&ch);
    if(ch==1)
    {
   new=(node*) malloc(sizeof(node));
    printf("Enter an element");
         scanf("%d",&new->data);
    curr->link=new;
    curr=new;
    }
    else
    {
   curr->link=NULL;
    break;
    }



    }
    temp=start2;
    printf("the elements in the list 2 are:\n");
    while(temp!=NULL)
    {
```

```
        printf("%d\n",temp->data);
        temp=temp->link;
           }


}
void search()
{
           int x,ele;
           printf("Enter 1 to search list 1,enter 2 to search list 2 ");
   scanf("%d",&x);
           if(x==1)
           {
           temp=start1;
           }
           else
           {
           temp=start2;
           }
           if(temp==NULL)
           {
      printf("Linked list is empty");
           return;
```

```
            }
            printf("Enter element to be searched");
    scanf("%d",&ele);
            while(temp!=NULL)
            {
            if(temp->data==ele)
            {
           printf("Element found\n");
            return;
            }
        temp=temp->link;

            }
    printf("Element not found\n");

}
void sort()
{
            node *a,*b,*temp1;
            int x,count=0,t,i,j,n;
            printf("Enter 1 to sort list 1,enter 2 to sort list 2 ");
    scanf("%d",&x);
            if(x==1)
```

```
        {
        temp=start1;
        temp1=start1;
        }
        else
        {
        temp=start2;
        temp1=start2;
        }
        while(temp!=NULL)
        {
        count++;
    temp=temp->link;
        }
        n=count;
        a=temp1;
        b=temp1->link;
for(i=0;i<n-1;i++)
        {
    for(j=0;j<n-i-1;j++)
        {
       if(a->data>b->data)
        {
```

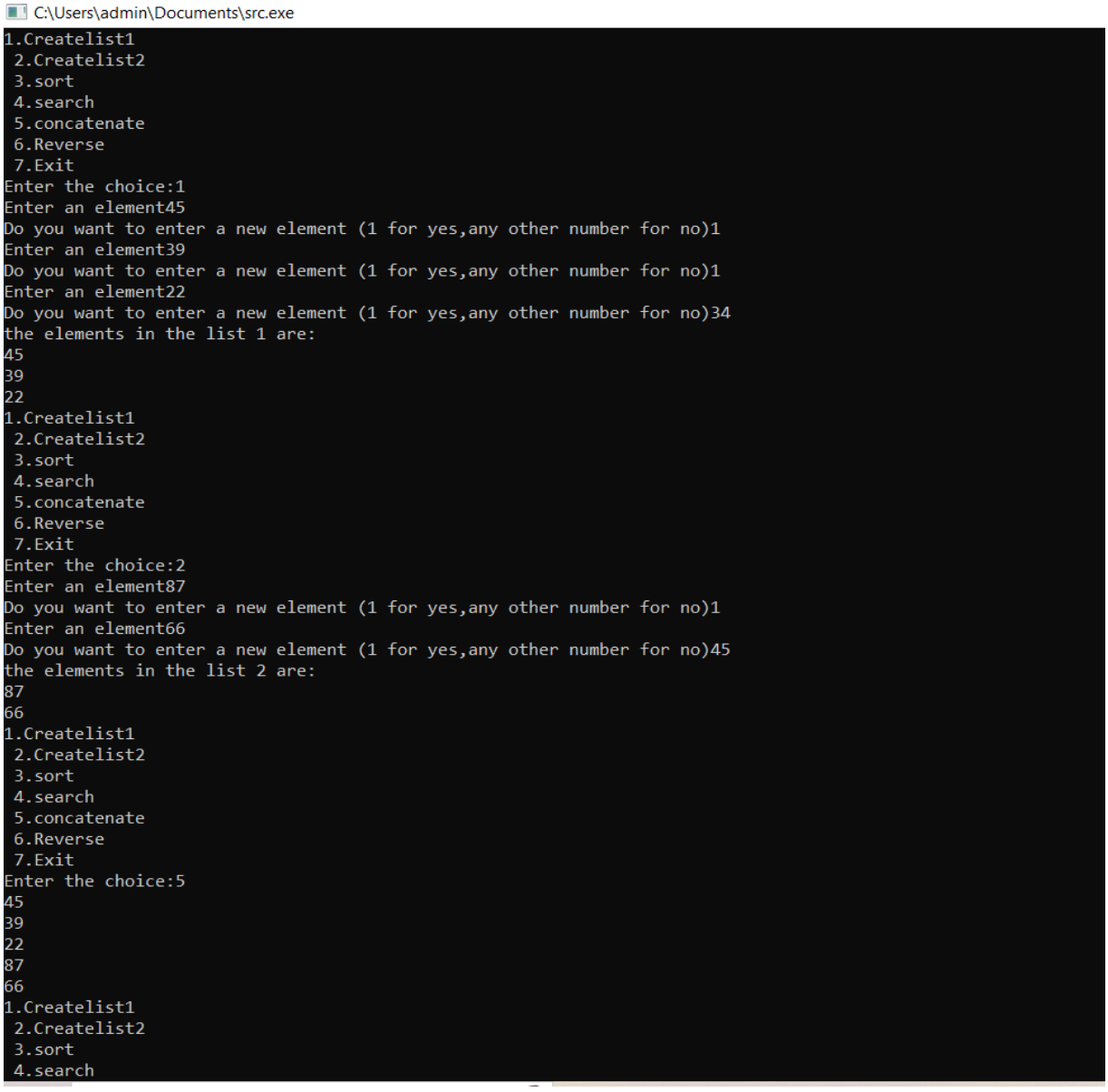
```
            t=a->data;
            a->data=b->data;
            b->data=t;
                }
           a=b;
          b=b->link;
           }
           a=temp1;
      b=temp1->link;
           }
           while(temp1!=NULL)
           {
      printf("%d\n",temp1->data);
      temp1=temp1->link;
           }
}
void concatenate()
{
           if(start1==NULL)
           {
           start3=start2;
           return;
           }
```

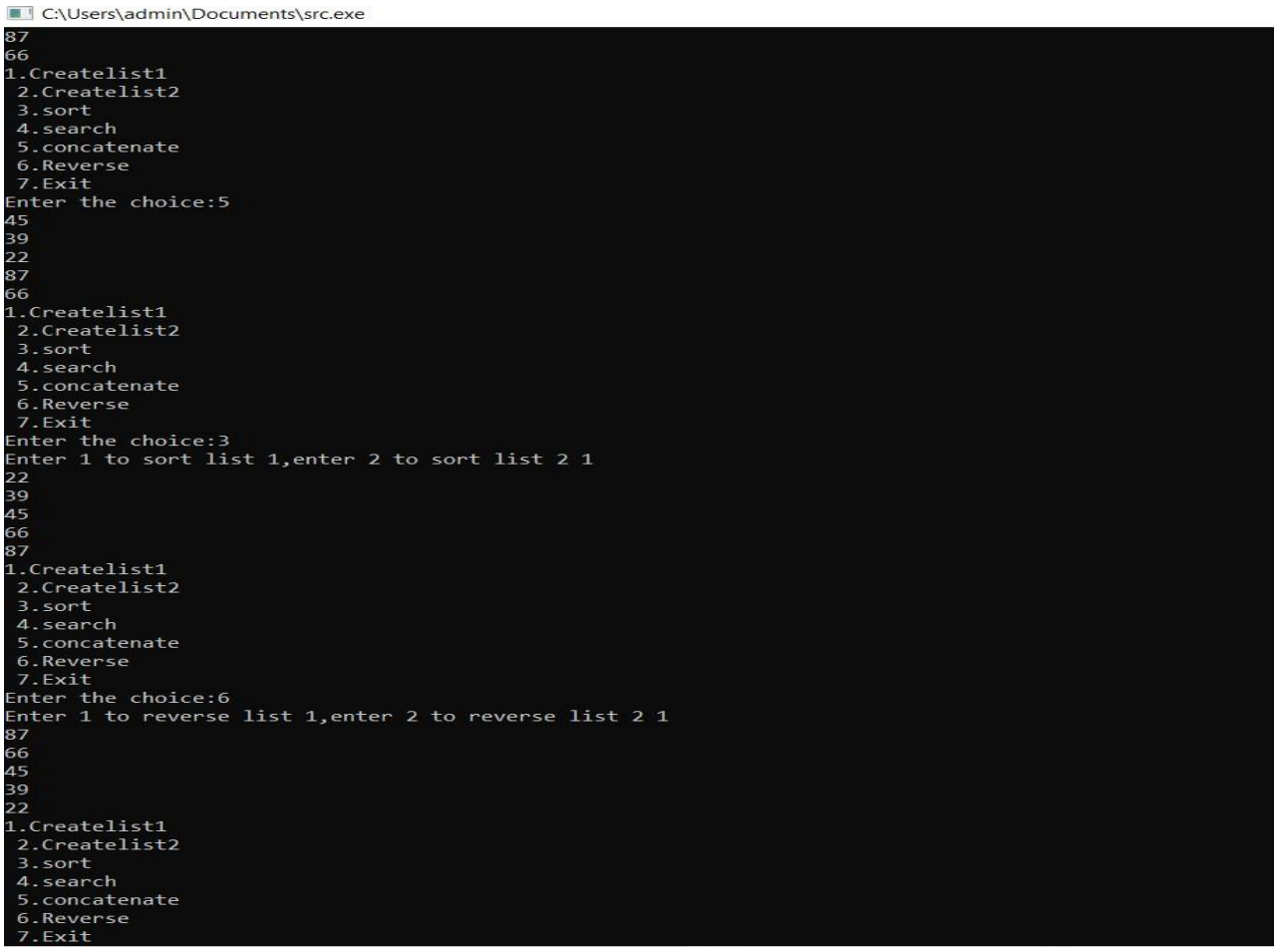
```
            if(start2==NULL)
            {
            start3=start1;
            return;
            }
            start3=start1;
             temp=start1;
    while(temp->link!=NULL)
            {
       temp=temp->link;
            }
    temp->link=start2;
            temp=start3;
            while(temp!=NULL)
            {
       printf("%d\n",temp->data);
       temp=temp->link;
            }

}
void reverse()
{
            int x;
```

```
        node *a,*b=NULL,*c=NULL;
        printf("Enter 1 to reverse list 1,enter 2 to reverse list 2 ");
scanf("%d",&x);
        if(x==1)
        {
        temp=start1;
        }
        else
        {
        temp=start2;

        }
        a=temp;
        while(a!=NULL)
        {
        c=b;
        b=a;
        a=a->link;
        b->link=c;
        }
        temp=b;
        while(temp!=NULL)
        {
```

```
        printf("%d\n",temp->data);
        temp=temp->link;
            }
}
```

OUTPUT:






## LAB PROGRAM 8:

WAP to implement Stack & Queues using Linked Representation.

```
#include<stdio.h>

#include<stdlib.h>

struct NODE{

        int data;

        struct NODE *link;
```

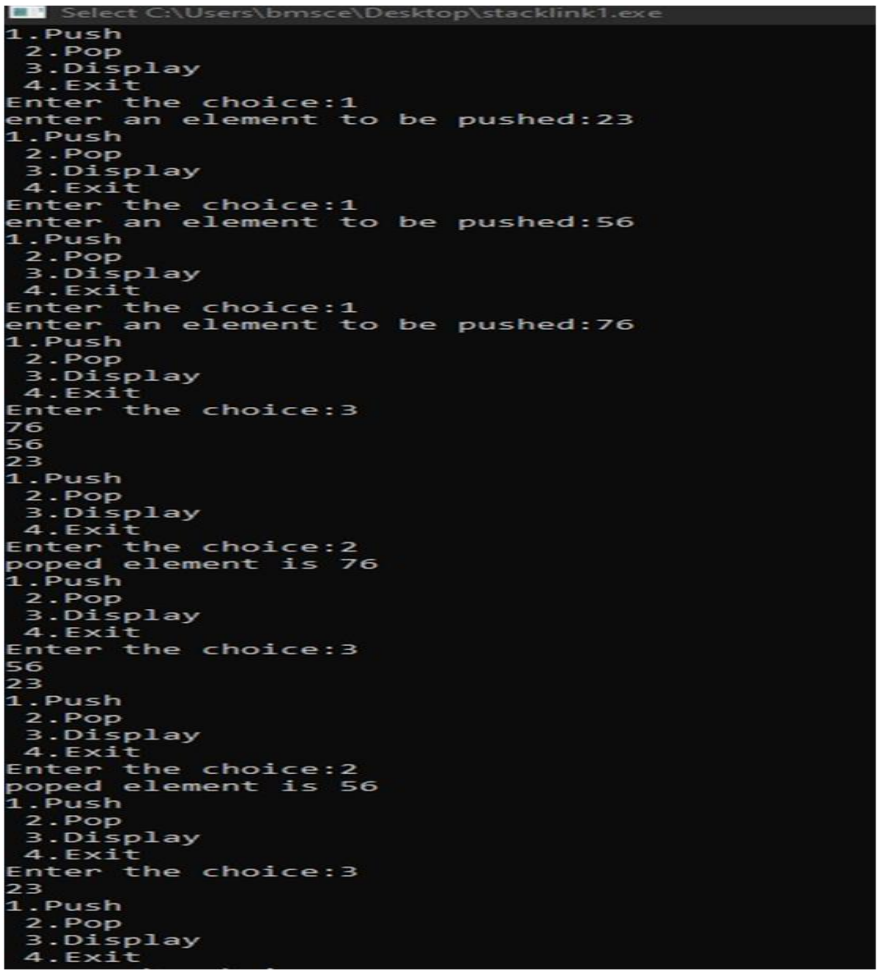
```
};
typedef struct NODE node;
node *new,*top=NULL,*temp;
void push();
void pop();
void display();
void main()
{

        int choice;
        while(1)
        {

        printf("1.Push \n 2.Pop \n 3.Display \n 4.Exit \n");
        printf("Enter the choice:");
        scanf("%d",&choice);
        switch(choice)
        {

        case 1: push();
                break;
        case 2:pop();
                break;
```

```
        case 3: display();
                break;
        case 5: exit(0);
                break;
        default:printf("Wrong Choice");
        }
        }
        getch();
}
void push()
{
  new=(node*)malloc(sizeof(node));
  printf("enter an element to be pushed:");
  scanf("%d",&new->data);
  if(top==NULL)
  {
        top=new;
        top->link=NULL;
  }
  else
  {
        new->link=top;
        top=new;
```

```
  }
}
void pop()
{
        int del;
        if(top==NULL)
        {
        printf("stack is empty\n");
        return;
        }
        del=top->data;
        printf("poped element is %d\n",del);
        top=top->link;

}
void display()
{
        if(top==NULL)
        {
        printf("stack is empty\n");
        return;
        }
        temp=top;
```

```
        while(temp!=NULL)
        {
        printf("%d\n",temp->data);
        temp=temp->link;
        }
}
```

OUTPUT STACKS:




QUEUES

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

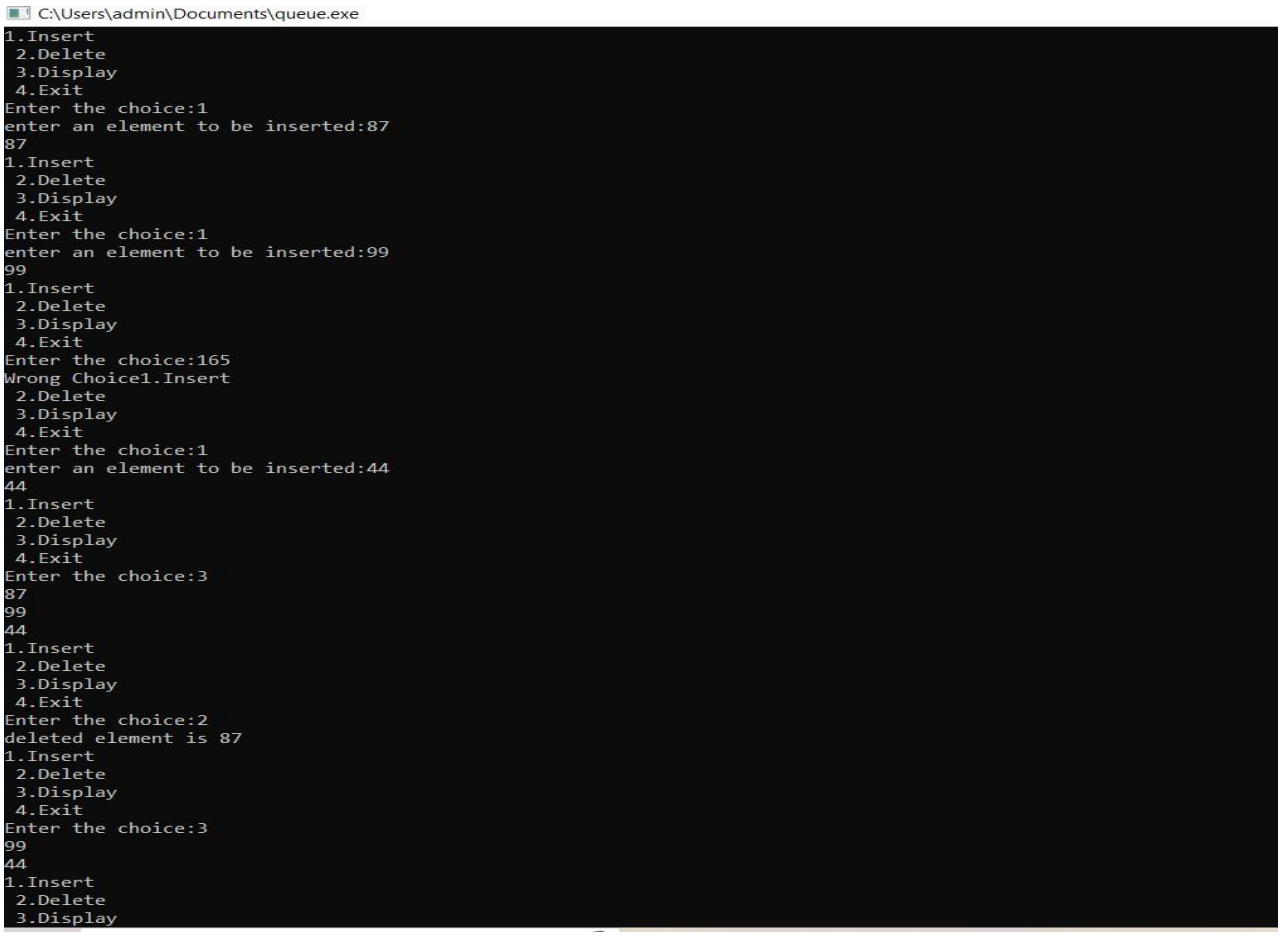
```
struct NODE{
        int data;
        struct NODE *link;
};
typedef struct NODE node;
node *new,*front=NULL,*rear=NULL,*temp,*temp1;
void insert();
void  delete();
void display();
void main()
{

        int choice;
        while(1)
        {

    printf("1.Insert \n 2.Delete \n 3.Display \n 4.Exit \n");
    printf("Enter the choice:");
    scanf("%d",&choice);
        switch(choice)
        {

        case 1: insert();
```

```
            break;
        case 2:delete();
            break;
        case 3: display();
            break;
        case 4: exit(0);
            break;
        default:printf("Wrong Choice");
        }
        }
        getch();
}
void insert()
{
   new=(node*)malloc(sizeof(node));
  printf("enter an element to be inserted:");
  scanf("%d",&new->data);
  if(front==NULL&&rear==NULL)
  {
        front=new;
        rear=new;
     front->link=NULL;
  }
```

```
  else
  {
     rear->link=new;
          rear=new;
     rear->link=NULL;
  }
}
void delete()
{
          if(front==NULL)
          {
       printf("queue is empty\n");
          return;
          }
   printf("deleted element is %d\n",front->data);
          temp=front;
   front=front->link;
          free(temp);

}
void display()
{
          if(front==NULL)
```

```
        {
    printf("queue is empty\n");
        return;
        }
        temp1=front;
        while(temp1!=NULL)
        {
    printf("%d\n",temp1->data);
    temp1=temp1->link;
  }
}
```

OUTPUT:




## LAB PROGRAM 9:

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

#include<stdio.h>

```
#include<stdlib.h>
struct NODE{
        int data;
        struct NODE *Llink,*Rlink;
};
typedef struct NODE node;
node *start=NULL,*new,*curr,*temp;
void create();
void display();
void insert();
void delete();
void main()
{

        int choice;
        while(1)
        {

    printf("1.Create \n 2.Display \n 3.Insert \n 4.Delete the specific value \n 5.Exit \n");
    printf("Enter the choice:");
    scanf("%d",&choice);
    switch(choice)
        {
```

```
            case 1: create();
                break;
            case 2: display();
                break;
            case 3: insert();
                break;
            case 4: delete();
                break;
            case 5: exit(0); break;
           default:printf("Wrong Choice");
            }
            }
            getch();
}

void create()
{

            int ch;
    start=(node*) malloc(sizeof(node));
            curr=start;
    printf("Enter an element");
```

```
scanf("%d",&start->data);
        while(1)
        {
        printf("Do you want to enter a new element (1 for yes,any other number for no)");
    scanf("%d",&ch);
    if(ch==1)
        {
        new=(node*) malloc(sizeof(node));
        curr->Rlink=new;
        new->Llink=curr;
        printf("Enter an element");
        scanf("%d",&new->data);
        curr=new;

        }
        else
        {
       curr->Rlink=NULL;
       break;
        }


        }
```

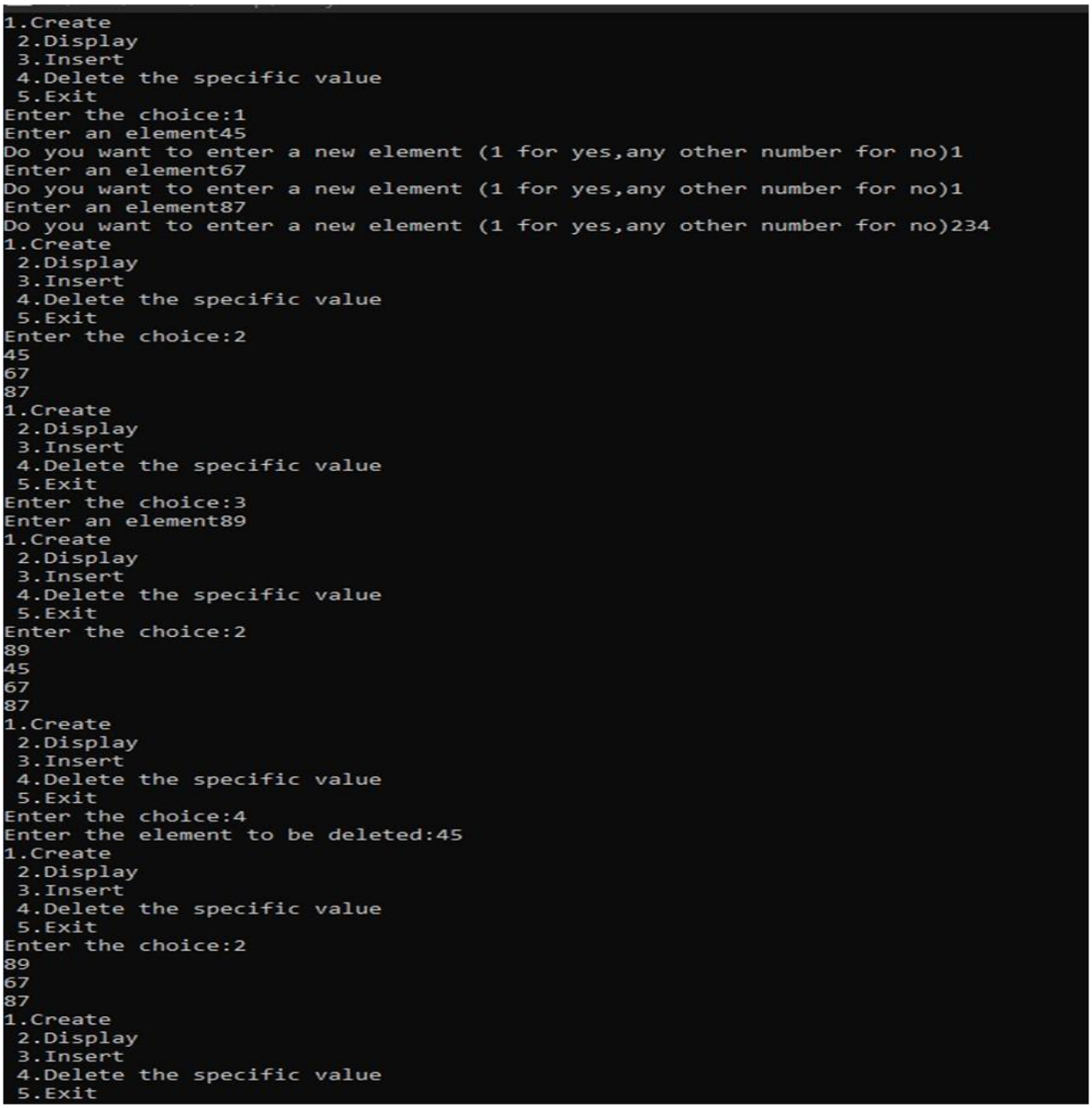
```
}
void display()
{
   if(start==NULL)
          {
      printf("Linked List is empty");
          return;
          }
          temp=start;
   while(temp!=NULL)
          {
      printf("%d\n",temp->data);
      temp=temp->Rlink;
          }
}
void insert()
{
   new=(node*)malloc(sizeof(node));
   printf("Enter an element");
   scanf("%d",&new->data);
   if(start==NULL)
          {
```

```
        new->Llink=NULL;
        new->Rlink=NULL;
        start=new;
            return;
            }
    start->Llink=new;
    new->Rlink=start;
    new->Llink=NULL;
            start=new;




}
void delete()
{
            int ele;
    if(start==NULL)
            {
        printf("Linked list is empty");
            return;
            }
    printf("Enter the element to be deleted:");
    scanf("%d",&ele);
```

```
if(start->Rlink!=NULL&&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&&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;
        }
}
```

OUTPUT:



```
1.Create
 2.Display
 3.Insert
 4.Delete the specific value
 5.Exit
Enter the choice:1
Enter an element45
Do you want to enter a new element (1 for yes,any other number for no)1
Enter an element67
Do you want to enter a new element (1 for yes,any other number for no)1
Enter an element87
Do you want to enter a new element (1 for yes,any other number for no)234
1.Create
 2.Display
 3.Insert
 4.Delete the specific value
 5.Exit
Enter the choice:2
45
67
87
1.Create
 2.Display
 3.Insert
 4.Delete the specific value
 5.Exit
Enter the choice:3
Enter an element89
1.Create
 2.Display
 3.Insert
 4.Delete the specific value
 5.Exit
Enter the choice:2
89
45
67
87
1.Create
 2.Display
 3.Insert
 4.Delete the specific value
 5.Exit
Enter the choice:4
Enter the element to be deleted:45
1.Create
 2.Display
 3.Insert
 4.Delete the specific value
 5.Exit
Enter the choice:2
89
67
87
1.Create
 2.Display
 3.Insert
 4.Delete the specific value
 5.Exit
```