# NAME - Arka Sinha

**USN - 1BM19CS024** 

**SUBJECT – Data Structures Lab Report** 

**ACADEMIC YEAR - 2020-2021** 

#### LAB PROGRAM 1

#### Source code-

```
#include<stdio.h>
                    #include<stdlib.h>
                    struct Student
                        int age;
                        int marks;
                    };
                    int validateAge(int age)
                    {
                        if(age<0)
                            return -1;
                        else
                            return 1;
                    }
                    int validateMarks(int marks)
                    {
                        if(marks<0 || marks>100)
                            return -1;
                        else
                            return 1;
                    }
                    int qualify(int age,int marks)
                        if(age>20 && marks>=65)
                            return 1;
                        else
                            return 0;
                    }
                    int main()
                    {
                        struct Student s;
                        int checkAge,checkMarks,qualified;
                        printf("Enter age and marks:\n");
                        scanf("%d%d",&s.age,&s.marks);
                        checkAge=validateMarks(s.age);
                        checkMarks=validateMarks(s.marks);
                        if(checkAge==1 && checkMarks==1){
                            qualified=qualify(s.age,s.marks);
                            if (qualified==1)
                                 printf("Qualified!!");
                            else
```

```
printf("Not Qualified");
}
else
    printf("Invalid age or marks");
return 0;
}
```

```
Tabl 18419CSO24 Page pullin
# include allio to
Hindurde additions
street Student
intage;
int marks;
int validate Age (int age)
if (age = 0)
neturn -1;
 dre
 int qualify [ind age, int marks)

if [age > 20 & L marks > = 65)

return 0;
```

#### ScreenShot-

```
input

Enter age and marks:

21

87

Qualified!!

...Program finished with exit code 0

Press ENTER to exit console.
```

```
input

Enter age and marks:

32

61

Not Qualified

...Program finished with exit code 0

Press ENTER to exit console.
```

#### **LAB PROGRAM 2**

#### Source code-

```
int stack[size];
int top=-1;
int main()
    int c,n,a=1;
   while(a==1)
        printf("\n1-Push\n2-Pull\n3-Display\n");
        printf("Enter your choice: ");
        scanf("%d",&c);
        switch(c)
        {
            case 1:
                printf("Enter number to push into stack: ");
                scanf("%d",&n);
                push(n);
                break;
            case 2:
                n=pop();
                if(n!=INT_MIN)
                    printf("Number popped is %d\n",n);
                break;
            case 3:
                display();
                break;
            default:
                printf("Enter valid choice\n");
        }
        printf("\nPress 1 to continue: ");
        scanf("%d",&a);
    }
    return 0;
}
void push(int ele)
{
    if(top>=size)
        printf("Stack overflow\n");
        return;
    top++;
    stack[top]=ele;
    printf("Number stacked\n");
```

```
}
int pop()
{
   if(top<0)
        printf("Stack empty\n");
       return INT_MIN;
   return stack[top--];
}
void display()
{
   if(top==-1)
       printf("Stack is empty\n");
    }
    int i;
    for(i=top;i>=0;i--)
       printf("%d\n",stack[i]);
}
```

```
Week 2
                 pseudo code
int size=100;
int stack [ eige ];
Int top =- 1;
told push (ele)
  if (top>= size)

i print(" stack overflow \n");
    tob ++
  stack[top]=ele;
prints("Data stacked \n");
  if (topeo)
      printf ("Atack empty In");
return INT_MIN;
     return stack [top-]
      for (i=0; i < top+1; i+e)
print("/d", atach[i]);
```

Screen Shot-

```
V 2 3
                                   input
1-Push
2-Pull
3-Display
4-Exit
Enter your choice: 1
Enter number to push into stack: 12
Number stacked
1-Push
2-Pull
3-Display
4-Exit
Enter your choice: 1
Enter number to push into stack: 15
Number stacked
1-Push
2-Pull
3-Display
4-Exit
Enter your choice: 1
Enter number to push into stack: 52
Number stacked
1-Push
2-Pull
3-Display
4-Exit
Enter your choice: 2
Number popped is 52
1-Push
2-Pull
3-Display
4-Exit
```

```
input
Number popped is 52
1-Push
2-Pull
3-Display
4-Exit
Enter your choice: 3
15
12
1-Push
2-Pull
3-Display
4-Exit
Enter your choice: 1
Enter number to push into stack: 52
Number stacked
1-Push
2-Pull
3-Display
4-Exit
Enter your choice: 3
52
15
12
1-Push
2-Pull
3-Display
4-Exit
Enter your choice: 4
...Program finished with exit code 0
Press ENTER to exit console.
```

## **Lab Program 3**

#### Source Code-

```
#include<stdio.h>
                     #include<stdlib.h>
                     #include<string.h>
                     #define SIZE 20
                     char stack[SIZE];
                     int top = -1;
                     void push(char ele)
                     {
                             if(top >= SIZE)
                             {
                                     printf("\nStack Overflow.");
                             }
                             else
                             {
                                     top = top+1;
                                     stack[top] = ele;
                             }
                     char pop()
                     {
                             char ele ;
                             if(top==-1)
                                     printf("stack under flow: invalid infix expression");
                                     getchar();
                                     exit(1);
                             }
                             else
                             {
                                     ele = stack[top];
                                     top = top-1;
                                     return(ele);
                             }
                     }
                     int is_operator(char symbol)
                             if(symbol == '^' \mid \mid symbol == '*' \mid \mid symbol == '/' \mid \mid symbol
                     == '+' || symbol =='-')
                             {
```

```
return 1;
       }
       else
       {
       return 0;
}
int higher(char symbol)
    switch(symbol)
    {
        case '^':
            return(3);
            break;
        case '*':
        case '/':
            return(2);
            break;
        case '+':
        case '-':
            return(1);
            break;
        default:
            return(0);
            break;
    }
void InfixToPostfix(char infix_exp[], char postfix_exp[])
       int i=0, j=0;
       char ele;
       char x;
       push('(');
       strcat(infix_exp,")");
       ele=infix_exp[i];
       while(ele != '\0')
       {
              if(ele == '(')
                      push(ele);
              else if(ele=='A' || ele=='B' || ele=='C' || ele=='D'
|| ele=='E' || ele=='F' || ele=='G' || ele=='H' || ele=='I' ||
ele=='J' || ele=='K' || ele=='L' || ele=='M' || ele=='N' || ele=='O'
|| ele=='P' || ele=='Q' || ele=='R' || ele=='S' || ele=='T' ||
ele=='U' || ele=='V' || ele=='W' || ele=='X' || ele=='Y' || ele=='Z'
```

```
|| ele=='0' || ele=='1' || ele=='2' || ele=='3' || ele=='4' ||
ele=='5' || ele=='6' || ele=='7' || ele=='8' || ele=='9')
              {
                      postfix_exp[j] = ele;
                      j++;
              else if(is_operator(ele) == 1)
                      x=pop();
                      while(is_operator(x) == 1 && higher(x)>=
higher(ele))
                      {
                             postfix_exp[j] = x;
                             j++;
                             x = pop();
                      }
                      push(x);
                      push(ele);
              }
              else if(ele == ')')
              {
                      x = pop();
                      while(x != '(')
                      {
                             postfix_exp[j] = x;
                             j++;
                             x = pop();
                      }
              }
              else
              {
                      printf("\nInvalid infix Expression.\n");
                      getchar();
                      exit(1);
              }
              i++;
        ele = infix_exp[i];
       postfix_exp[j] = '\0';
int main()
{
       char infix[SIZE], postfix[SIZE];
       printf("\nEnter Infix expression : ");
       gets(infix);
       InfixToPostfix(infix,postfix);
```

```
printf("Postfix Expression is: ");
puts(postfix);
return 0;
```

}

```
Week-3
      return (3);
return (2);

Ineak;

case '1';

cas '-1;
```

Date\_1\_1\_ Page Puffin default:
return 0;
reak; void infints portion (char

## ScreenShot-

#### Lab Program 4

#### Source Code-

```
#include
<stdio.h>
            #define MAX 3
            void insert();
            void delete();
            void display();
            int queue[MAX];
            int rear = -1;
            int front = - 1;
            void main()
                int choice;
                while (1)
                    printf("1.Insert element to queue \n");
                    printf("2.Delete element from queue \n");
                    printf("3.Display all elements of queue \n");
                    printf("4.Quit \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);
                        default:
                            printf("Enter a valid choice\n\n");
                    }
                }
            }
            void insert()
```

```
int add;
    if (rear == MAX - 1)
    printf("Queue Overflow\n\n");
    {
        if (front == -1)
            front = 0;
        printf("Enter element: ");
        scanf("%d", &add);
        printf("\n");
        rear = rear + 1;
        queue[rear] = add;
    }
}
void delete()
{
    if (front == - 1 || front > rear)
        printf("Queue Underflow\n\n");
        return ;
    }
    else
    {
        printf("Deleted element: %d\n\n", queue[front]);
        front = front + 1;
    }
}
void display()
    int i;
    if (front == - 1)
        printf("Queue is empty\n\n");
    else
    {
        printf("Queue is : \n");
        for (i = front; i <= rear; i++)</pre>
            printf("%d\n", queue[i]);
        printf("\n");
    }
}
```

```
Week 4
  int queu [nax]=;
  int rear =-1;
  int front = 1;
 # define MAX 50
 void insert ()
  int add;
   if ( near == hAx-1)
   print[ 1" Queus Overflow \n \n"];
     4 ( front == -1)
    front = 0;
prints ("Enter element: ");
    pearfilled ", badd);
    print("(h");
    Sean = sean +1;
   queen ( rear) = ads;
void delete!)
 If (front == -1 | front > rean)
 print/1- Queue Underflow In \h ");
else
```

Date Page Puffin krint ("Deleted element: igh \n't, queue (front)) void display () int i;
if (front = =-1)
printf("Queeu is empty \n\n"); for (iz front; i == rear; i++)

print("' 'Id \n", queue [i]);

print(" \n"),

#### ScreenShot-

#### C:\Users\arkas\OneDrive\Desktop\c\C\main.exe

```
4.Quit
Enter your choice : 1
Enter element: 1
1.Insert element to queue
2.Delete element from queue
3.Display all elements of queue
4.Quit
Enter your choice : 1
Enter element: 2
1.Insert element to queue
2.Delete element from queue
3.Display all elements of queue
4.Quit
Enter your choice : 1
Enter element: 3
1.Insert element to queue
2.Delete element from queue
3.Display all elements of queue
4.Quit
Enter your choice : 1
Queue Overflow
1.Insert element to queue
2.Delete element from queue
3.Display all elements of queue
4.Quit
Enter your choice :
```

## **Lab Program 5**

#### Source code-

```
#include<stdio.h>
                    # define MAX 5
                    int cqueue[MAX];
                    int front = -1;
                    int rear = -1;
                    void insert(int item)
                    {
                        if((front == 0 && rear == MAX-1) || (front == rear+1))
                             printf("Queue Overflow\n");
                            return;
                        else if(front == -1)
                        {
                            front = 0;
                            rear = 0;
                        }
                        else
                        {
                             if(rear == MAX-1)
                                rear = 0;
                             else
                                rear = rear+1;
                        cqueue[rear] = item ;
                    void delete()
                    {
                        if(front == -1)
                             printf("Queue Underflow\n");
                            return ;
                        printf("Element deleted from queue is : %d\n",cqueue[front]);
                        if(front == rear)
                            front = -1;
                            rear=-1;
                        }
                        else
                             if(front == MAX-1)
```

```
front = 0;
        else
            front = front+1;
    }
}
void display()
    int front_pos = front,rear_pos = rear;
    if(front == -1)
    {
        printf("Queue is empty\n");
        return;
    }
    printf("Queue elements :\n");
    if( front_pos <= rear_pos )</pre>
        while(front_pos <= rear_pos)</pre>
        {
            printf("%d ",cqueue[front_pos]);
            front_pos++;
        }
    else
    {
        while(front_pos <= MAX-1)</pre>
        {
            printf("%d ",cqueue[front_pos]);
            front_pos++;
        }
        front_pos = 0;
        while(front_pos <= rear_pos)</pre>
        {
            printf("%d ",cqueue[front_pos]);
            front_pos++;
        }
    printf("\n");
}
int main()
{
    int choice,item;
    do
    {
        printf("1.Insert\n2.Delete\n3.Display\n4.Quit\n");
        printf("Enter your choice : ");
        scanf("%d",&choice);
        switch(choice)
        {
```

```
case 1 :
                printf("Input the element for insertion in queue :
");
                scanf("%d", &item);
                printf("\n");
                insert(item);
                break;
            case 2 :
                delete();
                break;
            case 3:
                display();
                break;
            case 4:
                break;
            default:
                printf("Enter a valid choice!!\n");
        }
    }while(choice!=4);
    return 0;
}
```

```
Date /-/-
Page
Puffin
              LAB-S
            Cincular Queue
  # include < stolio h=
  # define MAX 5
  int copieur [HAX];
  int front =-1;
 int rear = -1;
 void insert (int item) &
  if (16ront == 0 18 near == HAX-1) | (front == rear +1))
  prints ("Queue overflow \n");
return;
  else if (bront = = -1)
 if ( nean==HAX-1)
  rean = 0;
  near=near+1};
3 cqueue [near] = ilem;
if (front = = -1)
```

```
Page Pullin
 print ("Queue Underflow In");
return;
 print (" Element deleted from queue is 10 m, question)
if (front = near)
 front = -1;
 near = -1;
else
 4 (front == HAX-1)
front = front +1;
void display ()
int front poo = front, rear poo = rear,
if ( bront ==-1)
 print/ (" ause Queue emply 'n");
neturn;
printy ( Queu elements: \n");
Uf ( front-pioc= near - pio)
     while (bront-pos= near-pos)
```

Page Puffin print ("1d", (queue (front-poo) while ( front - poo <= MAX -1 print[[]d" cqueue [front - peo]) front\_po=0; while (front-per = rear - per) prints ("1d", equeue [ pront-pro] front-poot+; int choice, item; prints ("Insent In2. Delete In3. Display In4. Quit In"); prints ("Enter your choice: "); Scanf ("10", & choice); switch (choice)

Date\_/\_/\_
Page
Puffin Care 13 print ["Edement for insertion: "); Scant ("' 1 d", & item); print(("\h");
incert(item); break; case 2: delete (); break; display (); break, case 4: Ireah; default

prints ("Entervalid choice!! \n'!);

}

Swhole (choice!=4), return 0;

#### ScreenShot-

```
C:\Users\arkas\OneDrive\Desktop\c\C\main.exe
Input the element for insertion in queue : 11
 1.Insert
 1.Insert
2.Delete
3.Display
4.Quit
Input the element for insertion in queue : 22
  2.Delete
3.Display
  .Orsping)
4.Quit
Enter your choice : 1
Input the element for insertion in queue : 33
 1.Insert
 2.Delete
3.Display
 4.Quit
Enter your choice : 1
Input the element for insertion in queue : 44
 2.Delete
3.Display
  4.Quit
Enter your choice : 1
Input the element for insertion in queue : 55
 1.Insert
 2.Delete
3.Display
  4.Quit
Enter your choice : 1
Input the element for insertion in queue : 66
   ueue Overflow
Queue Overflow

1.Insert

2.Delete

3.Display

4.Quit
Enter your choice : 3

Queue elements :

11 22 33 44 55
```

#### C:\Users\arkas\OneDrive\Desktop\c\C\main.exe

```
2.Defet
3.Display
4.Quit
Enter your choice : 1
Input the element for insertion in queue : 33
1.Insert
2.Delete
3.Display
4.Quit
Enter your choice : 1
Input the element for insertion in queue : 55
1.Insert

2. Delete
3. Display
4. Quit
Enter your choice : 2
Element deleted from queue is : 33
1.Insert
2. Delete
3. Display
4. Quit
Enter your choice : 2
Element deleted from queue is : 55
1.Insert
2. Delete
3. Display
4. Quit
Enter your choice : 2
Element deleted from queue is : 55
1.Insert
2. Delete
3. Display
 2.Delete
3.Display
4.Quit
Enter your choice : 2
Queue Underflow
1.Insert
2.Delete
3.Display
3.Display
4.Quit
Enter your choice : 3
Queue is empty
1.Insert
2.Delete
3.Display
4.Quit
Enter your choice : 4
Process returned 0 (0x0) execution time : 53.632 \text{ s} Press any key to continue.
```

## Lab Program 7

#### Source Code-

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

```
struct node
    int data;
    struct node *next;
};
struct node *head=NULL;
int length=0;
void inend(int ele)
{
    struct node *newnode,*temp;
    newnode=(struct node*)malloc(sizeof(struct node));
    newnode->data=ele;
    newnode->next=NULL;
    if(head==NULL)
    {
        head=newnode;
        length=1;
    }
    else
    {
        temp=(struct node*)malloc(sizeof(struct node));
        temp=head;
        while(temp->next!=NULL)
            temp=temp->next;
        temp->next=newnode;
        length++;
    }
}
void infro(int ele)
    struct node *temp;
    temp=(struct node*)malloc(sizeof(struct node));
    temp->data=ele;
```

```
temp->next=head;
    head=temp;
    length++;
}
void inran(int ele,int pos)
    if(pos==1)
        infro(ele);
    else if(pos>=length)
        inend(ele);
    else
    {
        struct node *inst;
        inst=(struct node*)malloc(sizeof(struct node));
        struct node *temp;
        temp=(struct node*)malloc(sizeof(struct node));
        temp=head;
        for(int i=1;i<pos-1;i++)</pre>
        {
               temp=temp->next;
        }
        inst->data=ele;
        inst->next=temp->next;
        temp->next=inst;
        length++;
    }
}
void del(int ele)
{
     struct node *temp,*del;
     temp=(struct node*)malloc(sizeof(struct node));
     del=(struct node*)malloc(sizeof(struct node));
     del=NULL;
     if(head->data==ele)
     {
         del=head;
         head=head->next;
         del->next=NULL;
     }
     else
     {
         temp=head;
```

```
while(temp->next!=NULL)
         {
            if(temp->next->data==ele)
            {
                del=temp->next;
                temp->next=del->next;
                del->next=NULL;
                length--;
                break;
            }
            else
            {
                temp=temp->next;
            }
         }
     }
     if(del==NULL)
     {
         printf("\nElement not found.\n");
     }
}
void display()
{
    struct node *temp;
    temp=(struct node*)malloc(sizeof(struct node));
    temp=head;
    if(temp==NULL)
        printf("\n List is empty \n");
    }
    else
    {
        printf("\nThe contents of the list are :\n");
       while(temp!=NULL)
        {
            printf("%d\n",temp->data);
            temp=temp->next;
        }
    }
}
int main()
```

```
{
    int choice,ele,pos;
    char ch;
    do
    {
    printf("\n1. Inset at end \n2.Insert at front \n3.Insert in
between \n4. Display \n5. Delete \n6.Exit");
    printf("\nEnter your choice : ");
    scanf("%d",&choice);
    switch(choice)
    {
        case 1: printf("Enter the element to be inserted\n");
                scanf("%d",&ele);
                inend(ele);
                break;
        case 2: printf("Enter the element to be inserted\n");
                scanf("%d",&ele);
                infro(ele);
                break;
        case 3: printf("Enter the element to be inserted\n");
                scanf("%d",&ele);
                printf("Enter the position \n");
                scanf("%d",&pos);
                inran(ele,pos);
                break;
        case 4: display();
                break;
        case 5: printf("Enter the element to be deleted\n");
                scanf("%d",&ele);
                del(ele);
                break;
    }
    }while(choice!=6);
    return 0;
}
```

```
1ab-7
#include < stdio h>
struct mode
int data;
struct mode + ment;
struct mode * head= NULL;
int longth = 0;
wind hend (int de)
struct modo * newrode, * demp)
newwoode = ( shout much +) mallor (sign of shout model,
newwoode - data = ment ple;
newrode - ment = NULL;
if (head == NULL)
head = new node;
lengts = 1;
temp = (struct mode *) mallor (size of (struct mode))
temp - ment = neumode;
lengte ++;
void infro (int ele, int pos)
if (poz==1)
```

```
Page
infra (ele);
 elecif (pro>=longth)
 Inend (ele);
 struct mode + inst:
 inst (stauct mode ") mallot (sizeof, (struct node));
struct node "temp,
temp = (struct mode +) mallor (size of (street mode)).
 temp=read;
 for (int i=1; i < per-1; i++)
 temp=temp- next;
Inst - duta = elo;
int - ment = temp - nent;
lamp - ment = int;
Longth +++,
void del (int ele)
struct node * temp, *del;
temp (Street mode") mallor (size of (street mode))
del = (struct mode + ) mallor ( size of Istruct norde) li
del=NULL)
if (reus-data == ele)
dol = head;
```

```
Puffin
  head = head - next;
  del - ment = NULL;
  elle
 temp=head;
 while (temp - nent != NULL)
  if (temp - nent -> data == ele)
 del=temp= ment;
temp= nent = del= nent;
 del -> ment = NULLS
 else
 temp = temp = nest;
hrbnt/ ("Ih Element not found. \n');
void display()
```

struct hode " temp; temp = (struct mode ) mallor (riger (struct mode)); temp = head, if (temp == NULL) frints ("In List is emply in"); else print[ l'In The contents of the list are: \n"); temp = temp -> next; 3

## ScreenShot-

```
exit
ter your choice : 1
ter the element to be inserted
```

## **Lab Program 8**

#### Source Code-

```
#include<stdio.h>
                    #include<stdlib.h>
                    struct node
                        int data;
                        struct node *next;
                    };
                    struct node *head1=NULL;
                    struct node *head2=NULL;
                    int length1=0;
                    int length2=0;
                    void ins_rear1(int ele)
                    {
                        struct node *newnode, *temp;
                        newnode=(struct node*)malloc(sizeof(struct node));
                        newnode->data=ele;
                        newnode->next=NULL;
                        if(head1==NULL)
                        {
                            head1=newnode;
                            length1=1;
                        }
                        else
                        {
                            temp=(struct node*)malloc(sizeof(struct node));
                            temp=head1;
                            while(temp->next!=NULL)
                                 temp=temp->next;
                            temp->next=newnode;
                            length1++;
                        }
                    void ins_rear2(int ele)
                    {
                        struct node *newnode, *temp;
                        newnode=(struct node*)malloc(sizeof(struct node));
                        newnode->data=ele;
                        newnode->next=NULL;
                        if(head2==NULL)
```

```
{
        head2=newnode;
        length2=1;
    }
    else
    {
        temp=(struct node*)malloc(sizeof(struct node));
        temp=head2;
        while(temp->next!=NULL)
            temp=temp->next;
        temp->next=newnode;
        length2++;
    }
}
void del_front1()
{
    struct node *temp;
    temp=(struct node*)malloc(sizeof(struct node));
    if(head1==NULL)
        printf("List is empty!!\n");
    }
    else
    {
        temp=head1;
        head1=temp->next;
        printf("%d is removed front front!!\n",temp->data);
        free(temp);
    }
}
void del_front2()
    struct node *temp;
    temp=(struct node*)malloc(sizeof(struct node));
    if(head2==NULL)
    {
        printf("List is empty!!\n");
    }
    else
    {
        temp=head2;
        head2=temp->next;
        printf("%d is removed front front!!\n",temp->data);
        free(temp);
```

```
}
}
void del_rear1()
    struct node *temp, *run;
    temp=(struct node *)malloc(sizeof(struct node));
    run=(struct node*)malloc(sizeof(struct node));
    temp=head1->next;
    if(run==NULL)
    {
        printf("List is empty!!\n");
    }
    else
    {
       while(temp->next!=NULL)
            run=temp;
            temp=temp->next;
        printf("%d is deleted from rear\n",temp->data);
        run->next=NULL;
        free(temp);
    }
}
void del_rear2()
{
    struct node *temp, *run;
    temp=(struct node *)malloc(sizeof(struct node));
    run=(struct node*)malloc(sizeof(struct node));
    temp=head2->next;
    run=head2;
    if(run==NULL)
        printf("List is empty!!\n");
    }
    else
    {
       while(temp->next!=NULL)
        {
            run=temp;
            temp=temp->next;
        printf("%d is deleted from rear\n",temp->data);
        run->next=NULL;
        free(temp);
    }
```

```
}
void del1(int ele)
    struct node *temp, *del;
    temp=(struct node*)malloc(sizeof(struct node));
    del=(struct node*)malloc(sizeof(struct node));
    del=NULL;
    if(head1->data==ele)
        del=head1;
        head1=head1->next;
        del->next=NULL;
    }
    else
    {
        temp=head1;
       while(temp!=NULL)
            if(temp->next->data==ele)
            {
                del=temp->next;
                temp->next=del->next;
                del->next=NULL;
                length1--;
                break;
            }
            else
                temp=temp->next;
            }
        }
    }
    if(del==NULL)
        printf("\nElement not found!!\n");
    }
void del2(int ele)
{
    struct node *temp, *del;
    temp=(struct node*)malloc(sizeof(struct node));
    del=(struct node*)malloc(sizeof(struct node));
    del=NULL;
    if(head2->data==ele)
    {
        del=head2;
```

```
head1=head2->next;
        del->next=NULL;
    }
    else
    {
        temp=head2;
       while(temp->next!=NULL)
        {
            if(temp->next->data==ele)
                del=temp->next;
                temp->next=del->next;
                del->next=NULL;
                length2--;
                break;
            }
            else
            {
                temp=temp->next;
        }
    }
    if(del==NULL)
        printf("\nElement not found!!\n");
    }
}
void sort1(struct node *h)
{
    int i,j,a;
    struct node *temp1;
    struct node *temp2;
    for(temp1=h;temp1!=NULL;temp1=temp1->next)
      {
        for(temp2=temp1->next;temp2!=NULL;temp2=temp2->next)
          {
            if(temp2->data < temp1->data)
                a = temp1->data;
                temp1->data = temp2->data;
                temp2->data = a;
              }
           }
       }
```

```
}
void sort2(struct node *h)
{
    int i,j,a;
    struct node *temp1;
    struct node *temp2;
    for(temp1=h;temp1!=NULL;temp1=temp1->next)
        for(temp2=temp1->next;temp2!=NULL;temp2=temp2->next)
          {
            if(temp2->data < temp1->data)
              {
                a = temp1->data;
                temp1->data = temp2->data;
                temp2->data = a;
              }
           }
       }
}
void rev1(struct node *h)
{
    int i,j,a;
    struct node *temp1;
    struct node *temp2;
    for(temp1=h;temp1!=NULL;temp1=temp1->next)
    {
        for(temp2=temp1->next;temp2!=NULL;temp2=temp2->next)
        {
            a = temp1->data;
            temp1->data = temp2->data;
            temp2->data = a;
        }
    }
}
void rev2(struct node *h)
    int i,j,a;
    struct node *temp1;
    struct node *temp2;
    for(temp1=h;temp1!=NULL;temp1=temp1->next)
```

```
{
        for(temp2=temp1->next;temp2!=NULL;temp2=temp2->next)
            a = temp1->data;
            temp1->data = temp2->data;
            temp2->data = a;
        }
    }
}
void conc()
{
    struct node *temp, *run;
    temp=(struct node*)malloc(sizeof(struct node));
    run=(struct node*)malloc(sizeof(struct node));
    run=head1;
    if(length1==0 && length2==0)
    {
        printf("\nBoth lists empty\n");
    }
    else
    {
        temp=head1;
       while(temp->next!=NULL)
            temp=temp->next;
        temp->next=head2;
    printf("\nThe elements in the concatenated list is-\n");
    while(run!=NULL)
    {
        printf("%d\n",run->data);
        run=run->next;
    }
}
void display1()
    struct node *temp;
    temp=(struct node*)malloc(sizeof(struct node));
    temp=head1;
    if(temp==NULL)
        printf("\nList is empty!!\n");
    }
    else
    {
```

```
printf("\nThe contents in list 1 are-\n");
        while(temp!=NULL)
            printf("%d\n",temp->data);
            temp=temp->next;
        }
    }
}
void display2()
    struct node *temp;
    temp=(struct node*)malloc(sizeof(struct node));
    temp=head2;
    if(temp==NULL)
        printf("\nList is empty!!\n");
    }
    else
    {
        printf("\nThe contents in list 2 are-\n");
        while(temp!=NULL)
            printf("%d\n",temp->data);
            temp=temp->next;
        }
    }
}
int main()
{
    int ch, item;
    printf("Choose any option-");
    do{
        printf("\n1.Insert at rear in list 1\n2.Insert at rear in
list 2\n3.Delete random from list 1\n4.Delete random from list
2\n5.Remove from front in list 1\n6.Remove from front in list
2\n7.Remove from rear in list 1\n8.Remove from rear in list
2\n9.Display list 1\n10.Display list 2\n11.Sort list 1\n12.Sort list
2\n13.Reverse list 1.\n14.Reverse list 2\n15.Concatenate list 1 and
list 2\n16.Exit\n");
        scanf("%d",&ch);
        switch(ch)
        {
        case 1:
            printf("\nEnter the number to be inserted: ");
            scanf("%d",&item);
            printf("\n");
```

```
ins_rear1(item);
   break;
case 2:
   printf("\nEnter the number to be inserted: ");
   scanf("%d",&item);
    printf("\n");
   ins_rear2(item);
   break;
case 3:
   printf("\nEnter the number to be deleted: ");
    scanf("%d",&item);
    printf("\n");
   del1(item);
   break;
case 4:
   printf("\nEnter the number to be deleted: ");
    scanf("%d",&item);
    printf("\n");
   del2(item);
   break;
case 5:
   del_front1();
   break;
case 6:
   del_front2();
   break;
case 7:
   del_rear1();
   break;
case 8:
   del_rear2();
   break;
case 9:
   display1();
   break;
case 10:
   display2();
   break;
case 11:
   sort1(head1);
   break;
case 12:
    sort2(head2);
   break;
case 13:
   rev1(head1);
```

## Writeup-

```
Late 8 18414(5024 Page Puffin
  # implude < etdio h >
  # include c std lib h >
  struct medo
  struct mode "ment;
  struct mode + head = NULL;
  void in rear ( int ele)
  struct mode * newnode, * temp;
 new rood = (struct rode =) mallor (sped (struct node))
 neurode -> data = ele;
 neurode > nent = NULL;
 if (head == NULL)
 head = neunode;
demp = (struct mode *) malla (size of (struct mode))
While (semp-) ment = ! = NULL)
temp > meat = newrod;
```

Date Page Puffin void del- front () struct rode \* temp; temp: (struct mode &) mallor (sized (struct mode)). if (head = = NULL) printy ("List is Enably \n"); elee temp = head; head & temp > hear next; printl (" "Id is removed In", temp - data ); dree (temp); void del-near () struct mode \* temp, \*run; Jemp = fatreet node +) mallor (size of (struct mode)) New = (struct mode \*) mallor (size of (struct mode)) temp = head -> next if ( sun= 2 NULL) prints ("List is Empty \n"); else while (temp-) next! = NULL)

Date 1\_1\_ Page Pullin New - temp temp = temp > mont; punt ("1-d , deleted In", temp = data) Dem - ment = NULLE free (temp); world del ( int ele) struct mode \* temp, & del; temp=[struct mode \*) malloc (size of (struct mode)); del = (struct mode \*) malloc (size of (struct mode)); del = NOLL if ( head -) data = = ele) del = head; head = head - ment; del = next = NULL; temp = head; while temp! = NULZ) if (temp=) ment -> data = = ele) del=temp-) ment;

```
Page Puffin
temp => nent = del >> ment;
del > ment = NULL;
break;
glee
temp = temp > ment;
if (del = = NULL)
printy 1" In Element not found In");
void sort (struct mode + h)}
Int i, isas
struct mode + temp!;
strud nort * temp?;
for (lemp 1= h; templ= NULL; temp 1= Fatement
 top (temp2 = temp); temp2:=NULL; temp2 = temp2 - ) ment
if I sompl > data < templ > data)
a - temp 1 > data;
temp 1 >data = temp 2 - data)
temp) > date = a;
```

Date 1\_1 Page Puffin void new (struct mode th) street mode \* sempl, + sampl; for (ramp)=h; temp1!=NULL; temp1=temp1 > ment) for (tem 2 = temp1; telmp2 != NULL; temp2 = temp2 -> next) a =templ > data; temp 1 = data = lemp 2 > data; temp 2 -> data = a; void conc smuct mode \* temp, \* run;

temp: [smuct mode \*) omallor (size of (struct mode));

run = (struct mode \*) mallor (size of (struct mode)); Alin=head; if head == NULL & read 2 == NULL) print[l'Both lists empts \n"); temp = head; While temp - ment! = NULL) Jemp = temp = nent)

Page Puffin temp => ment = head?;

print( 1" In The concatenated that - In");

while ( run != NULL)

{ frints ("/d In", rem-> data);

#### Screen Shots-

```
Choose any option-
1.Insert at rear in list 1
                                                                          input
2.Insert at rear in list 2
3.Delete random from list 1
4.Delete random from list 2
5.Remove from front in list 1
6.Remove from front in list 2
7.Remove from rear in list 1
8.Remove from rear in list 2
9.Display list 1
10.Display list 2
11.Sort list 1
12.Sort list 2
13.Reverse list 1.
14.Reverse list 2
15.Concatenate list 1 and list 2
16.Exit
Enter the number to be inserted: 11
1.Insert at rear in list 1
2.Insert at rear in list 2
3.Delete random from list 1
4.Delete random from list 2
5.Remove from front in list 1
6.Remove from front in list 2
7.Remove from rear in list 1
8.Remove from rear in list 2
9.Display list 1
10.Display list 2
11.Sort list 1
12.Sort list 2
13.Reverse list 1.
14.Reverse list 2
```

```
✓ ✓ .

14.Reverse list 2
15.Concatenate list 1 and list 2
16.Exit
Enter the number to be inserted: 12
1.Insert at rear in list 1
2.Insert at rear in list 2
3.Delete random from list 1
4.Delete random from list 2
5.Remove from front in list 1
6.Remove from front in list 2
7.Remove from rear in list 1
8.Remove from rear in list 2
9.Display list 1
10.Display list 2
11.Sort list 1
12.Sort list 2
13.Reverse list 1.
14.Reverse list 2
15.Concatenate list 1 and list 2
16.Exit
Enter the number to be inserted: 13
1.Insert at rear in list 1
2.Insert at rear in list 2
3.Delete random from list 1
4.Delete random from list 2
5.Remove from front in list 1
6.Remove from front in list 2
7.Remove from rear in list 1
```

```
5.Remove from front in list 1
6.Remove from front in list 2
7.Remove from rear in list 1
8.Remove from rear in list 2
9.Display list 1
10.Display list 2
11.Sort list 1
12.Sort list 2
13.Reverse list 1.
14.Reverse list 2
15.Concatenate list 1 and list 2
16.Exit
The contents in list 1 are-
11
12
13
1.Insert at rear in list 1
2.Insert at rear in list 2
3.Delete random from list 1
4.Delete random from list 2
5.Remove from front in list 1
6.Remove from front in list 2
7.Remove from rear in list 1
8.Remove from rear in list 2
9.Display list 1
10.Display list 2
11.Sort list 1
12.Sort list 2
13.Reverse list 1.
14.Reverse list 2
15.Concatenate list 1 and list 2
16.Exit
```

```
→ ∠′ ,g
                                                                       input
13 is deleted from rear
1.Insert at rear in list 1
2.Insert at rear in list 2
3.Delete random from list 1
4.Delete random from list 2
5.Remove from front in list 1
6.Remove from front in list 2
7.Remove from rear in list 1
8.Remove from rear in list 2
9.Display list 1
10.Display list 2
11.Sort list 1
12.Sort list 2
13.Reverse list 1.
14.Reverse list 2
15.Concatenate list 1 and list 2
16.Exit
The contents in list 1 are-
11
12
```

```
Choose any option
                                                                         input
2.Add to list 2
3.Delete from list 1
4.Delete from list 2
5.Display list 1
6.Display list 2
7.Sort list 1
8.Sort list 2
9.Reverse list 1.
10.Reverse list 2
11.Concatenate list 1 and list 2
12.Exit
Enter the number to be inserted: 11
1.Add to list 1
2.Add to list 2
3.Delete from list 1
4.Delete from list 2
5.Display list 1
6.Display list 2
7.Sort list 1
8.Sort list 2
9.Reverse list 1.
10.Reverse list 2
11.Concatenate list 1 and list 2
12.Exit
Enter the number to be inserted: 12
1.Add to list 1
2.Add to list 2
3.Delete from list 1
```

```
9.Concatenate list 1 and list 2
Enter the number to be inserted: 63
1.Add to list 1
2.Add to list 2
3.Delete from list 1
4.Delete from list 2
5.Sort list 1
6.Sort list 2
7.Display list 1
8.Display list 2
9.Concatenate list 1 and list 2
10.Exit
Enter the number to be inserted: 32
1.Add to list 1
2.Add to list 2
3.Delete from list 1
4.Delete from list 2
5.Sort list 1
6.Sort list 2
7.Display list 1
8.Display list 2
9.Concatenate list 1 and list 2
10.Exit
The contents in list 1 are-
4213
23
256
```

```
V 2 3
                                                                               input
3.Delete from list 1
4.Delete from list 2
5.Display list 1
6.Display list 2
8.Sort list 2
9.Reverse list 1.
10.Reverse list 2
11.Concatenate list 1 and list 2
12.Exit
Enter the number to be inserted: 13
1.Add to list 1
2.Add to list 2
3.Delete from list 1
4.Delete from list 2
5.Display list 1
6.Display list 2
8.Sort list 2
9.Reverse list 1.
10.Reverse list 2
11.Concatenate list 1 and list 2
12.Exit
The contents in list 1 are-
11
12
1.Add to list 1
2.Add to list 2
3.Delete from list 1
4.Delete from list 2
```

```
3.Delete from list 1
                                                                        input
4.Delete from list 2
5.Display list 1
6.Display list 2
7.Sort list 1
8.Sort list 2
9.Reverse list 1.
10.Reverse list 2
11.Concatenate list 1 and list 2
12.Exit
1.Add to list 1
2.Add to list 2
3.Delete from list 1
4.Delete from list 2
5.Display list 1
6.Display list 2
7.Sort list 1
8.Sort list 2
9.Reverse list 1.
10.Reverse list 2
11.Concatenate list 1 and list 2
12.Exit
The contents in list 1 are-
13
12
11
1.Add to list 1
2.Add to list 2
3.Delete from list 1
4.Delete from list 2
5.Display list 1
6.Display list 2
7.Sort list 1
```

```
Choose any option
2.Add to list 2
3.Delete from list 1
4.Delete from list 2
5.Sort list 1
6.Sort list 2
 7.Display list 1
8.Display list 2
 9.Concatenate list 1 and list 2
10.Exit
Enter the number to be inserted: 123
1.Add to list 1
 2.Add to list 2
 3.Delete from list 1
4.Delete from list 2
5.Sort list 1
6.Sort list 2
7.Display list 1
8.Display list 2
 9.Concatenate list 1 and list 2
10.Exit
Enter the number to be inserted: 153
1.Add to list 1
2.Add to list 2
3.Delete from list 1
4.Delete from list 2
5.Sort list 1
6.Sort list 2
7.Display list 1
```

```
7.Display list 18.Display list 2
                                                                               input
9.Concatenate list 1 and list 2
10.Exit
Enter the number to be inserted: 123
1.Add to list 1
2.Add to list 2
3.Delete from list 1
4.Delete from list 2
5.Sort list 1
6.Sort list 2
7.Display list 1
8.Display list 2
9.Concatenate list 1 and list 2
10.Exit
Enter the number to be inserted: 531
1.Add to list 1
2.Add to list 2
3.Delete from list 1
4.Delete from list 2
5.Sort list 1
6.Sort list 2
7.Display list 1
8.Display list 2
9.Concatenate list 1 and list 2
10.Exit
Enter the number to be inserted: 23
```

```
♥ x . . Enter the number to be inserted: 23
 .Add to list 1
2.Add to list 2
3.Delete from list 1
4.Delete from list 2
5.Sort list 1
6.Sort list 2
7.Display list 1
8.Display list 2
9.Concatenate list 1 and list 2
Enter the number to be inserted: 51
1.Add to list 1
2.Add to list 2
3.Delete from list 1
4.Delete from list 2
5.Sort list 1
 6.Sort list 2
7.Display list 1
8.Display list 2
9.Concatenate list 1 and list 2
10.Exit
Enter the number to be inserted: 9
1.Add to list 1
2.Add to list 2
3.Delete from list 1
4.Delete from list 2
5.Sort list 1
6.Sort list 2
```

```
6.Sort list 2
7.Display list 1
8.Display list 2
9.Concatenate list 1 and list 2
Enter the number to be inserted: 21
1.Add to list 1
2.Add to list 2
3.Delete from list 1
4.Delete from list 2
 5.Sort list 1
 6.Sort list 2
7.Display list 1
8.Display list 2
9.Concatenate list 1 and list 2
10.Exit
The contents in list 2 are-
123
153
123
531
23
51
1.Add to list 1
2.Add to list 2
3.Delete from list 1
4.Delete from list 2
5.Sort list 1
6.Sort list 2
```

```
input
7.Display list 1
8.Display list 2
9.Concatenate list 1 and list 2
10.Exit
1.Add to list 1
2.Add to list 2
3.Delete from list 1
4.Delete from list 2
5.Sort list 1
6.Sort list 2
7.Display list 1
8.Display list 2
9.Concatenate list 1 and list 2
10.Exit
The contents in list 2 are-
21
23
51
123
123
153
531
1.Add to list 1
2.Add to list 2
3.Delete from list 1
4.Delete from list 2
5.Sort list 1
6.Sort list 2
7.Display list 1
8.Display list 2
9.Concatenate list 1 and list 2
```

```
9.Concatenate list 1 and list 2
10.Exit
Enter the number to be inserted: 4213
1.Add to list 1
2.Add to list 2
3.Delete from list 1
4.Delete from list 2
5.Sort list 1
6.Sort list 2
7.Display list 1
8.Display list 2
9.Concatenate list 1 and list 2
10.Exit
Enter the number to be inserted: 23
1.Add to list 1
2.Add to list 2
3.Delete from list 1
4.Delete from list 2
5.Sort list 1
6.Sort list 2
7.Display list 1
8.Display list 2
9.Concatenate list 1 and list 2
10.Exit
Enter the number to be inserted: 256
1.Add to list 1
2.Add to list 2
```

```
2.Add to list 2
                                                                             input
3.Delete from list 1
4.Delete from list 2
5.Sort list 1
6.Sort list 2
7.Display list 1
8.Display list 2
9.Concatenate list 1 and list 2
10.Exit
Enter the number to be inserted: 235
1.Add to list 1
2.Add to list 2
3.Delete from list 1
4.Delete from list 2
5.Sort list 1
6.Sort list 2
7.Display list 1
8.Display list 2
9.Concatenate list 1 and list 2
10.Exit
Enter the number to be inserted: 834
1.Add to list 1
2.Add to list 2
3.Delete from list 1
4.Delete from list 2
5.Sort list 1
6.Sort list 2
7.Display list 1
8.Display list 2
9.Concatenate list 1 and list 2
```

## Lab Program 9

#### Source Code-

```
#include<stdio.h>
                    #include<stdlib.h>
                    struct node
                    {
                        int data;
                        struct node *next;
                        struct node *prev;
                    };
                    struct node *head;
                    void ins_left()
                    {
                        struct node *newnode;
                        newnode=(struct node*)malloc(sizeof(struct node));
                        printf("Enter the value: ");
                        scanf("%d",&newnode->data);
                        newnode->prev=NULL;
                        newnode->next=NULL;
                        if(head==NULL)
                            head=newnode;
                        }
                        else
                        {
                            newnode->next=head;
                            head->prev=newnode;
                            head=newnode;
                        }
                    void ins_end()
                    {
                            struct node *newnode,*temp;
                            newnode=(struct node*)malloc(sizeof(struct node));
                            printf("Enter the number: ");
                            scanf("%d",&newnode->data);
                            newnode->next=NULL;
                            newnode->prev=NULL;
                            if(head==NULL)
                            {
                                   head=newnode;
                            }
```

```
else
       {
                temp=head;
               while(temp->next!=NULL)
                temp=temp->next;
                temp->next=newnode;
                newnode->prev=temp;
       }
}
void ins_ran()
       int ele;
       struct node *newnode,*temp;
       printf("Enter the element in the list: ");
       scanf("%d",&ele);
       newnode=(struct node*)malloc(sizeof(struct node));
       printf("Enter the new node data: ");
       scanf("%d",&newnode->data);
       newnode->next=NULL;
       newnode->prev=NULL;
    if(head==NULL)
    {
        printf("List is Empty/n");
        return;
    }
       temp=head;
       while(temp->data!=ele)
       {
               temp=temp->next;
               if(temp==NULL)
               {
                      printf("Element is not in the list\n");
                      return;
               }
       }
       newnode->next=temp->next;
       temp->next=newnode;
       newnode->prev=temp;
       newnode->next->prev=newnode;
}
void del()
{
       struct node *temp;
       int ele;
```

```
if(head==NULL)
    {
        printf("List is empty!!\n");
        return;
    }
       printf("Enter the element to be deleted: ");
       scanf("%d",&ele);
       temp=head;
       while(temp->data!=ele)
       {
               temp=temp->next;
               if(temp==NULL)
               printf("Element not in list\n");
               break;
               }
        }
        if(temp==head)
               head=head->next;
        else if(temp->next==NULL)
                      temp=temp->prev;
                      temp->next=NULL;
        }
        else
        {
               temp->prev->next=temp->next;
               temp->next->prev=temp->prev;
        }
}
void display()
        struct node *temp;
        temp=head;
        while(temp!=NULL)
        {
               printf("%d\t",temp->data);
               temp=temp->next;
        printf("\n");
}
int main()
```

```
int ch;
    printf("Choose from the following: \n");
        printf("1.Insert at the beginning\n2.Insert at the
end\n3.Insert in between\n4.Delete a number\n5.Display
list\n6.Exit\n");
        scanf("%d",&ch);
        switch(ch)
        case 1:
            ins_left();
            break;
        case 2:
            ins_end();
            break;
        case 3:
            ins_ran();
            break;
        case 4:
            del();
            break;
        case 5:
            display();
            break;
        case 6:
            break;
        default:
            printf("Invalid choice!!/n");
        }
    }while(ch!=6);
    return 0;
}
```

# Writeup-

#include = elder h >  #thinclude < stalib : h >  shreet mode * ment;  street mode * prev;  3:  shreet mode * prev;  3:  shreet mode * prev;  3:  shreet mode * prev;  printy (* 1 d ", & newnode)  mewnode > prev = NULL;  in (head == NULL)  feed = mewnode;  selse  newnode > prev = head)  head - prev = newnode;  head = prevnode;  head = prevnode;  head = prevnode;  head = mewnode;  shreet = mewnode;  head = mewnode;		Lab-9 18119CS	024 Page Puffin
# include < stable h >  shreet mode * ment;  street mode * prev;  3;  shreet mode * prev;  2 street mode * newnode;  newnode = 'street mode *) mallor (inject Intruct mode  printy ("Id", I newnode > data);  newnode > prev = NULL;  newnode > ment = NULL;  if (head == NULL)  head = newnode;  3  else  2  newnode > nent = head;  head - >prev = newnode;  head = newnode;  3;  visit ino-end()  2	#include	e addicina	Mark Austi
int data;  struct mode * mont;  etnect mode * prev;  3;  struct mode * prev;  newnode = (struct mode *) mallor (sized Intruct mode  printy ("Id", & newnode >> data);  mewnode >> prev = NULL;  newnode >> ment = NULL;  if (head == NULL)  head = mewnode;  3  else  2  mewnode >> ment = head;  head -> prev = newnode;	11 .		
int data;  struct mode * mont;  etneut mode * knew;  3;  when feft()  remode = (druct mode *) mallor (expert Intruct north  print ("Id", k newnorde > data);  mewnorde > prew = NULL;  rewrorde > ment = NULL;  il (head == NULL)  head = mewnorde;  } else  2  mewnords > ment = head;  head - prew = newnorde;  head = mewnorde;  head = mewnorde;  head = mewnorde;  head = mewnorde;	11		Printer of the state of the sta
struct mode * mont;  struct mode * prev;  3;  struct mode * meumode;  maurode - (struct mode *) malloc (sized (struct mode print) ("13", & newmode > data);  meumode > prev = NULL;  if (head == NULL)  {  head = meumode; }  else  else  else  neumode > ment = head;  head - prev = neumode;  head = neumode;  head = neumode;  head = neumode;  head = neumode;	5	Can ate.	me labba
struct mode * mont;  struct mode * prev;  3;  struct mode * meumode;  maurode - (struct mode *) malloc (sized (struct mode print) ("13", & newmode > data);  meumode > prev = NULL;  if (head == NULL)  {  head = meumode; }  else  else  else  neumode > ment = head;  head - prev = neumode;  head = neumode;  head = neumode;  head = neumode;  head = neumode;	int data;		
struct mode * prev;  3;  struct mode * prevnode;  rewroode = (struct mode +) mallor (sized Intract mode printf ("Id", & newporte -> data);  mewnode -> prev = NULL;  if (head == NULL)  {  head = mewnode;  }  else  2  mewnode => ment = head;  head -> prev = newrode;  rend -> prev = newrode;  }  void ino end()  d			
struct mode * head;  void ino_left()  I struct mode * newnode;  newnode = (struct mode *) mallor (sized Intruct mode  printf("Id", & newnode > data);  meumode > prew = NULL;  meumode > ment = NULL;  il (head == NULL)  {  head = newnode;  }  else  2  meumode > ment = head;  head -> prew = newnode;  rend = newnode;  3  void ino_end()  d	street no	rde * prev;	BOOK BOOK AND A STATE OF THE PARTY OF THE PA
retruct mode + neumode;  neumode - (struct mode +) mallor (sized Intruct mode  printy ("1d", & newmode -> data);  meumode -> prev = NULL;  neumode -> ment = NULL;  if (head == NULL)  head = neumode;  }  else  2  neumode -> prev = neumode;  head -> prev = neumode;  head -> prev = neumode;  head = neumode;  all ino - end()  delare -> pred -> neumode;	3:		48
retruct mode + newwoode;  newroode = (struct mode +) malloc (sized (struct mode print) ("10", & newwoode -> data);  mewroode -> prew = NULL;  newroode -> ment = NULL;  il (head == NULL)  head = newroode;  else  2  newroode => ment = head;  head -> prew = newroode;  head = newroode;  head = newroode;  head = newroode;	struct nocl	e * heads	
retruct mode + newwoode;  newroode = (struct mode +) malloc (sized (struct mode print) ("10", & newwoode -> data);  mewroode -> prew = NULL;  newroode -> ment = NULL;  il (head == NULL)  head = newroode;  else  2  newroode => ment = head;  head -> prew = newroode;  head = newroode;  head = newroode;  head = newroode;	vaid ins f	eft()	
meumode = (struct mode +) mallor (sized Istruct mode  printy ("10", & newmode > data);  meumode > prew = NULL;  meumode > ment = NULL;  if (head == NULL)  {  head = newmode;  }  else  {  meumode > ment = head;  head -> prew = newmode;  head = newmode;  sold ino end()  {  void ino end()  }	7	*	
meumode = (struct mode +) mallor (sized Istruct mode  printy ("10", & newmode > data);  meumode > prew = NULL;  meumode > ment = NULL;  if (head == NULL)  {  head = newmode;  }  else  {  meumode > ment = head;  head -> prew = newmode;  head = newmode;  sold ino end()  {  void ino end()  }	struct no	rde + newnode;	
meumode > prev = NULL;  meumode > prev = NULL;  il (head == NULL)  {  head = meumode;  }  else  el  meumode > ment = head)  head -> prev = newmode;  head = meumode;  head = prewnode;  all  head = prewnode;  head = meumode;	newnode -	struct mode + mallor	- ( sixed (street morb)
mewnode > prev = NULL;  mewnode > ment = NULL;  if (head == NULL)  head = mewnode;  lese  newnode > ment = head;  head -> prev = newnode;  head = mewnode;	print/"10"	I newporle - date	a);
mewnorde > ment = NULL;  il (head == NULL)  kead = mewnorde;  }  else  2  mewnorde > ment = head;  head -> prev = newnorde;  head = mewnorde;  sold = mewnorde;  d void i'mo end()  d	meunode-	prev = NULL:	
il (head == NULL)  kead = mewnode;  else  else  mewnode = ment = head)  head = prew = newnode;  head = mewnode;  head = mewnode;  yold ino_end()  d	newrode >	ment = NUCL.	
head = newhode;  else  else  newhode = newhode;  head -> prew = newhode;  head = newhode;  head = newhode;  and inc-end()	if (head ==	NULL)	
else  **  mewnorde => ment = head)  head -> prew = newnorde;  head = newnorde;  33  void ino-end()	19		
else  **  menonorde => ment = head)  head -> prew = newrode;  head = newrode;  3}  void ino-end()	head = new	hode;	
newrode = newrode;  head = prew = newrode;  head = newrode;  33  void ino-end()	13		
newnorde = new tead)  head = prew = new node;  head = new node;  33  void ino end()	else		
head => prew = new mode;  head = new mode;  3}  void ino end()	2		
head = prew = newrode;  head = newrode;  3}  void ino_end()	neumarle 2 m	nent= he di	
head = newnode;  33  void ino-end()			
33 void ins-end()			
void ins-end()		room j	
		445	
shurt made * meumante * +	Sol ino en	2()	
wheat made mannerle *1.	18	dx	
Jam D	shuct mode	* newnords * ton	mbi

Date\_\_\_\_ mounodo = (shuct nodo +) mallor (sigeof (struct node)); printl( Enter the number: "); scanf (" 1d", of new rode > data); nounode -> ment = NULL; neurod-) frev= NULL) if (head = = NU LL) head = new node else while I temp > ment! = NULL) temp = temp - ) ment; Temp > nent = neumode) neurode > prev= temp; void ins\_nan() int eles printl("Enter element of list: "); Rearly" 1d" & ele ; newnote = ( snuct node +) mallor ( sizet) ( snuct node printy ("Enter the new data"); scanf ("'Id", & newsode -> data); meunode to - nent : NULL: newnord prev = NULL; is [head = n=NULL].

Puffin printy (" Libt & empty \n") temp = head) while (temp-) data! = ele) temp = temp = nent; if (temp == NOCC) Drinty 1" Element not in list In") newnode - nent = temp - next temp -> ment = newrodes newrod - prev - temp; me new rode - nent prev - new rode void del () struct mode \* hemp; intele; if (read = = NOLL) print (" Yest is Enopty In"); return; prints ("Ender element to be deleted: "); scand ("1.d", & ele); temp = head; while (temp-)data! - ele) il (temp == NOLL) printy (" Element not in liet \n"); if (temp == head) she if (temp=) ment==NULL) temp=temp=kner; temp = next = NOLL; demp - prev -> menot = temp - menot; temp - nent - prev = temp prev; void display () struct node \* temp; temp = head; while (temp! = NUCC) printl("/d \t" temp = date ); }
temp = nemt; & printle ("In"); }

#### ScreenShot-

```
C:\Users\arkas\OneDrive\Desktop\c\C\bin\Debug\C.exe
```

```
Choose from the following:
1.Insert at the beginning
2.Insert at the end
3.Insert in between
4.Delete a number
5.Display list
6.Exit
1
Enter the value: 12
1.Insert at the beginning
2.Insert at the end
3.Insert in between
4.Delete a number
5.Display list
6.Exit
  2
Enter the number: 13
L.Insert at the beginning
2.Insert at the end
3.Insert in between
4.Delete a number
5.Display list
5.Exit
3
  The content of the list: 12 inter the new node data: 5
Linsert at the beginning
Linsert at the end
Sinsert in between
Linete a number
Lipital list
Est
  5.Exit
5.Exit
12 5 11 13
1.Insert at the beginning
2.Insert at the end
3.Insert in between
4.Delete a number
5.Display list
5.Exit
```

#### C:\Users\arkas\OneDrive\Desktop\c\C\bin\Debug\C.exe

```
5.Display list
6.Exit
     3
Enter the element in the list: 12
Enter the new node data: 5
1.Insert at the beginning
2.Insert at the end
3.Insert in between
4.Delete a number
5.Display list
6.Exit
5
   12 5 11 13
1.Insert at the beginning
2.Insert at the end
3.Insert in between
4.Delete a number
5.Display list
6.Exit
      4
Enter the element to be deleted: 11
1.Insert at the beginning
2.Insert at the end
3.Insert in between
4.Delete a number
5.Display list
6.Exit
12 5 13
1.Insert at the beginning
2.Insert at the end
3.Insert in between
4.Delete a number
5.Display list
6.Exit
      Process returned 0 (0x0) execution time : 48.027 s
Press any key to continue.
```

# **Lab Program 10**

#### Source Code-

```
#include<stdio.h</pre>
>
                   #include<stdlib.h>
                   #include<conio.h>
                    struct node
                    {
                     int info;
                     struct node *rlink;
                     struct node *llink;
                     };
                    typedef struct node *NODE;
                   NODE getnode()
                    {
                   NODE x;
                   x=(NODE)malloc(sizeof(struct node));
                    if(x==NULL)
                    {
                     printf("mem full\n");
                     exit(0);
                     }
                     return x;
                    }
                   void freenode(NODE x)
                    {
                   free(x);
                   NODE insert(NODE root, int item)
                   NODE temp, cur, prev;
                   temp=getnode();
                   temp->rlink=NULL;
                    temp->llink=NULL;
                   temp->info=item;
                    if(root==NULL)
                     return temp;
                    prev=NULL;
                    cur=root;
                   while(cur!=NULL)
                    prev=cur;
```

```
cur=(item<cur->info)?cur->llink:cur->rlink;
}
if(item<prev->info)
prev->llink=temp;
else
 prev->rlink=temp;
return root;
}
void display(NODE root,int i)
int j;
if(root!=NULL)
  display(root->rlink,i+1);
  for(j=0;j<i;j++)</pre>
         printf(" ");
  printf("%d\n",root->info);
        display(root->llink,i+1);
 }
}
NODE delete(NODE root,int item)
NODE cur, parent, q, suc;
if(root==NULL)
printf("empty\n");
return root;
parent=NULL;
cur=root;
while(cur!=NULL&&item!=cur->info)
parent=cur;
cur=(item<cur->info)?cur->llink:cur->rlink;
if(cur==NULL)
printf("not found\n");
return root;
if(cur->llink==NULL)
q=cur->rlink;
else if(cur->rlink==NULL)
q=cur->llink;
else
 {
```

```
suc=cur->rlink;
 while(suc->llink!=NULL)
  suc=suc->llink;
 suc->llink=cur->llink;
 q=cur->rlink;
 if(parent==NULL)
  return q;
 if(cur==parent->llink)
  parent->llink=q;
 else
  parent->rlink=q;
 freenode(cur);
 return root;
void preorder(NODE root)
if(root!=NULL)
 {
  printf("%d\n",root->info);
  preorder(root->llink);
  preorder(root->rlink);
  }
 }
void postorder(NODE root)
if(root!=NULL)
 {
  postorder(root->llink);
  postorder(root->rlink);
  printf("%d\n",root->info);
  }
 }
void inorder(NODE root)
if(root!=NULL)
{
  inorder(root->llink);
  printf("%d\n",root->info);
  inorder(root->rlink);
  }
 }
void main()
```

```
{
int item,choice;
NODE root=NULL;
for(;;)
{
printf("\n1.insert\n2.display\n3.pre\n4.post\n5.in\n6.delete\n7.exit\
printf("enter the choice\n");
scanf("%d",&choice);
switch(choice)
{
  case 1:printf("enter the item\n");
               scanf("%d",&item);
               root=insert(root,item);
               break;
  case 2:display(root,0);
               break;
  case 3:preorder(root);
               break;
  case 4:postorder(root);
               break;
  case 5:inorder(root);
               break;
  case 6:printf("enter the item\n");
               scanf("%d",&item);
               root=delete(root,item);
               break;
  default:exit(0);
                break;
         }
       }
 }
```

# Writeup-

	Lat-10 Date Pag	e /_/_ Puffin
	# include < stdioh?	alun i
	Himchele < stdlib.h?	
	# include = cono h>	101
	etnuct mode	
	1	
	int infos;	
	struct mode + slink)	
	struct node + Mint;	
3		
	of typedel struct mode *NODE;	
1	JODE getnode ()	NA I
1	VODE n;	
N	(=(NODE) mallor (sized (struct more	4):
i	( h= NULL)	(-11)
13		
	print( "trem full \n"); 2Mit (o);	7
	2Mit (0):	
3		
15	retern n;	
3		
Val	Nd freenode (NODE n)	1000
19	The work is	
11	nee (u);	
120	ue · · · )	
110	DE : 10+(NODE +)	
No	DE insert (NODE root, int item)	
110	25 1 1	
NO	of temp, cur, prev;	
term	p = setrode ();	
	The state of the s	

```
Page Pullin
temp-stime= NOLL;
temp > llink = NULL;
temp - infor = item;
il ( noot == NULL)
   return temp;
prev = NULL;
cun = noot;
while (eun!= NULL)
 prev= cur;
 cun=(item < cun > imfor)? cun > llink; an > rlink;
 if litem < prev > infor)
   prev > Minh = temp;
 alse
   preu -> rlink = temp;
 return root;
  void display (NODE root, inti)
  display ( noot - ) rlink, (1);

for (j=0; j<1;j+1).

print(("");

frint(("")d)n", root -> infor).

display ( noot -> Ulink, (1));
```

NODE delete (NODE root, int item) NODE, can parent q, suc; if (noct = NULL) printy ("Enply in"); selvin root; parent = NULL; aur = root, while (cur! = NULL & atem != cur-sinfo) parent : cus; cur = litem ( cur sinfor)? cur > llink: cur solmh; if (aur == NULL) printy ( " not found \ n "); if (aus -> llimb = NULL) q: cus → rlink; de if ( aur > rlink= NULL) q= cur > llink; else suc = cur - orlink. Juc = SOL > Climb! = NULL)

```
Due - llimb = cur - llimb.
  q=an-s slink;
if ( parent == NUL)
    setum q;
if (cur == parent = lionh)

g = parent = llinh = g;
     parent - slink=q;
freewood (cury)
return root.
 void presiden (NODE root)
 y ( Noot ) = NOLL)
  print( " /d In", root " into);
previder ( root -> blink);
 grearder (root + rlink);
 void portorder ( NODE root)
 if ( robt! = NOW)
posterder (root-s llimb);
posterder (root-s rlink);
print( !" I.d \n", root-s infor);
3
```

Puffin if (Noot!= NULL) inorder ( root - llink); printy ("1.dln", root - infor); inorder ( root - rlink);

## ScreenShot-

```
\blacksquare   
Select C:\Users\arkas\OneDrive\Desktop\c\C\bin\Debug\C.exeenter the choice
1.insert
2.display
3.pre
4.post
5.in
6.delete
7.exit
enter the choice
1.insert
2.display
3.pre
4.post
5.in
6.delete
7.exit
enter the choice
1.insert
2.display
3.pre
4.post
5.in
6.delete
7.exit
enter the choice
7
Process returned \theta (\theta x \theta) execution time : 1408.482 s
Press any key to continue.
```

#### Select C:\Users\arkas\OneDrive\Desktop\c\C\bin\Debug\C.exe

```
1. insert
2. display
3. pre
4. post
5. in
6. delete
7. exit
enter the item
23
1. insert
2. display
3. pre
4. post
5. in
6. delete
7. exit
enter the choice
enter the item
41
1. insert
2. display
4. post
5. in
6. delete
7. exit
enter the choice
enter the item
41
1. insert
2. display
4. post
5. in
6. delete
7. exit
enter the choice
enter the item
52
1. insert
6. delete
7. exit
enter the choice
enter the item
32
1. insert
```

#### Select C:\Users\arkas\OneDrive\Desktop\c\C\bin\Debug\C.exe