PROGRAM -1

Design, Develop and Implement a menu driven Program in C for the following Array operations

a. Creating an Array of N Integer Elements
b. Display of Array Elements with Suitable Headings
c. Inserting an Element (ELEM) at a given valid Position (POS)
d. Deleting an Element at a given valid Position (POS)
e. Exit.

Support the program with functions for each of the above operations

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

int a[20];       // integer array which can hold maximum 20 elements
int n, val, i, pos;

/*Function Prototype*/
void create();      // Function to create an array
void display();     // Function to display elements of  an array
void insert();      // Function to insert an element in an array
void delete();      // Function to delete an element from an array

int main()
{
        int choice;
        while(choice)
        {
                printf("\n\n--------MENU-----------\n");
                printf("1.CREATE\n");
                printf("2.DISPLAY\n");
                printf("3.INSERT\n");
                printf("4.DELETE\n");
                printf("5.EXIT\n");
                printf("----------------------");
                printf("\n ENTER YOUR CHOICE:\t");
                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("\n Invalid choice: \n");
                                break;
                }
        }
    return 0;
}

// Function to create an array
void create()
{
        printf("\n Enter the size of the array :\t");
        scanf("%d", &n);
        printf("\n Enter %d elements  the array:\n", n);
        for(i=0;i<n;i++)
        {
                scanf("%d", &a[i]);
        }
}

// Function to display elements of an array
void display()
{
        int i;
        printf("\nThe array elements are:\n");
        for(i=0;i<n;i++)
        {
                printf("%d\t",a[i]);
```

```
        }
}

// Function to insert an element into array
void insert()
{
        printf("\n Enter the position for the new element:\t");
        scanf("%d", &pos);
        printf("\n Enter the element to be inserted :\t");
        scanf("%d", &val);
        for(i=n-1;i>=pos;i--)
        {
                a[i+1]=a[i];
        }
        a[pos]=val;
        n=n+1;
}

// Function to delete an array element
void delete()
{
        printf("\n Enter the position of the element to be deleted:\t");
        scanf("%d", &pos);
        val=a[pos];
        for(i=pos;i<n-1;i++)
        {
                a[i]=a[i+1];
        }
        n=n-1;
        printf("\nThe deleted element is =%d",val);
}
```

OUTPUT

```
root/DSLAB:  vi lab1.c
root/DSLAB:  cc lab1.c
root/DSLAB:  ./a.out
--------MENU----------
1.CREATE
2.DISPLAY
3.INSERT
4.DELETE
5.EXIT
----------------------
ENTER YOUR CHOICE: 1
Enter the size of the array elements: 3
Enter the elements for the array:

10 25 30
ENTER YOUR CHOICE: 2
The array elements are:
10 25 30
ENTER YOUR CHOICE: 3
Enter the position for the new element: 1
Enter the element to be inserted: 20
ENTER YOUR CHOICE: 2
The array elements are:
10 20 25 30
ENTER YOUR CHOICE: 4
Enter the position of the element to be deleted: 3
The deleted element is =30

enter your choice: 5
```

PROGRAM-2

Design, Develop and Implement a Program in C for the following operations on Strings
a. Read a main String (STR), a Pattern String (PAT) and a Replace String (REP)
b. Perform Pattern Matching Operation: Find and Replace all occurrences of PAT in STR with REP if PAT exists in STR. Report suitable messages in case PAT does not exist in STR.
Support the program with functions for each of the above operations. Don't use Built-in functions.

```
#include<stdio.h>
void main()
{
        char STR[100],PAT[100],REP[100],ans[100];
        int i,j,c,m,k,flag=0;
        printf("\nEnter the MAIN string: \n");
        gets(STR);
        printf("\nEnter a PATTERN string: \n");
        gets(PAT);
        printf("\nEnter a REPLACE string: \n");
        gets(REP);
        i = m = c = j = 0;
        while ( STR[c] != '\0')
        {
                // Checking for Match
                if ( STR[m] == PAT[i] )

                {
                        i++;   m++;
                        flag=1;
                        if ( PAT[i] == '\0')
                        {
                                //copy replace string in ans string
                                for(k=0; REP[k] != '\0';k++,j++)
                                ans[j] = REP[k];
                                i=0;
                                c=m;
```

```
                }
            }
            else        //mismatch
            {
                ans[j] = STR[c];
                j++;
                c++;
                m = c;
                i=0;
            }
        }
        if(flag==0)
        {
            printf("Pattern doesn't found!!!");
        }
        else
        {
            ans[j] = '\0';
            printf("\nThe RESULTANT string is:%s\n" ,ans);
        }
}
```

OUTPUT

```
Enter the MAIN string:
good morning
Enter a PATTERN string:
morning
Enter a REPLACE string:
evening
The RESULTANT string is: good evening
```

PROGRAM-3

Design, Develop and Implement a menu driven Program in C for the following operations on STACK of Integers (Array Implementation of Stack with maximum size MAX)

a. Push an Element on to Stack
b. Pop an Element from Stack
c. Demonstrate how Stack can be used to check Palindrome
d. Demonstrate Overflow and Underflow situations on Stack
e. Display the status of Stack
f. Exit

Support the program with appropriate functions for each of the above operations

```
#include<stdio.h>
#include<stdlib.h>
#include<math.h>
#define max 3

int s[max],stop;
int ele,st[max],sp,ch;
void push(int ele,int s[],int *top);
int pop(int s[],int *top);
void palindrome(int ele,int st[]);
void display(int s[],int *top);

void main()
{
        stop=-1;
        sp=-1;
        while(1)
        {
                printf("Enter the choice\n");
                printf("Enter 1 to insert an element in to stack\n");
                printf("Enter 2 to delete an element from the stack\n");
                printf("Enter 3 to check an element is pailndrome or not\n");
                printf("Ebter 4 to check status of the stack\n");
```

```
            printf("Enter 5 to exit\n");
            scanf("%d",&ch);


            switch(ch)
            {
               case 1: printf("Enter the element to be inserted in to stack\n");
                       scanf("%d",&ele);
                       push(ele,s,&stop);
                       break;
               case 2: ele = pop(s,&stop);
                       if(ele>0)
                            printf("Element poped is %d\n",ele);
                       else
                            printf(" Element can't be popped \n");
                       break;
               case 3:
                       printf("Enter the element to check whether it is
                              palindrome or not\n");
                       scanf("%d",&ele);
                       palindrome(ele,st);
                       break;
               case 4: printf("The status of the stack is \n");
                        display(s,&stop);
                        break;
               case 5: exit(0);
             }
        }
}

void push(int ele,int s[],int *stop)
{
        if(*stop==max-1)
        {
                printf("stack overflow\n");
                return;
        }
```

```
        else
        {
                s[++*stop]=ele;
        }
}

int pop(int s[],int *top)
{
        if(*top == -1)
        {
                printf("stack underflow\n");
                return -1;
        }
        else
                return(s[(*top)--]);
}

void palindrome(int ele,int st[])
{       int rem,rev=0,temp=ele,i=0;

        while(temp!=0)
        {
                rem=temp%10;
                push(rem,st,&sp);
                temp=temp/10;
        }
        while(sp!=-1)
        {

        rev=rev+(pop(st,&sp)*pow(10,i++));
        }
        if(ele==rev)
                printf("Palindrome\n");
        else
                printf("Not palindrome\n");
}
```

```
void display(int s[],int *stop)
{
        int i;
        if(*stop==-1)
                printf("stack empty\n");
        else
                for(i=*stop;i>-1;i--)
                        printf("%d\n",s[i]);
}
```

OUTPUT

```
[tce123@localhost vb]$ cc -lm lab3.c
[tce123@localhost vb]$ ./a.out
Enter the choice
Enter 1 to insert an element in to stack
Enter 2 to delete an element from the stack
Enter 3 to check an element is palindrome or not
Enter 4 to check status of the stack
Enter 5 to exit
5
[tce123@localhost vb]$ vi lab3.c
[tce123@localhost vb]$ cc -lm lab3.c
[tce123@localhost vb]$ ./a.out
Enter the choice
Enter 1 to insert an element in to stack
Enter 2 to delete an element from the stack
Enter 3 to check an element is palindrome or not
Enter 4 to check status of the stack
Enter 5 to exit
1
Enter the element to be inserted in to stack
10
Enter the choice
Enter 1 to insert an element in to stack
Enter 2 to delete an element from the stack
Enter 3 to check an element is palindrome or not
```

Enter 4 to check status of the stack
Enter 5 to exit
1
Enter the element to be inserted in to stack
20
Enter the choice
Enter 1 to insert an element in to stack
Enter 2 to delete an element from the stack
Enter 3 to check an element is palindrome or not
Enter 4 to check status of the stack
Enter 5 to exit
1
Enter the element to be inserted in to stack
30
Enter the choice
Enter 1 to insert an element in to stack
Enter 2 to delete an element from the stack
Enter 3 to check an element is palindrome or not
Enter 4 to check status of the stack
Enter 5 to exit
1
Enter the element to be inserted in to stack
40
stack overflow
Enter the choice
Enter 1 to insert an element in to stack
Enter 2 to delete an element from the stack
Enter 3 to check an element is palindrome or not
Enter 4 to check status of the stack
Enter 5 to exit
4
The status of the stack is
30
20
10
Enter the choice
Enter 1 to insert an element in to stack

```
Enter 2 to delete an element from the stack
Enter 3 to check an element is palindrome or not
Enter 4 to check status of the stack
Enter 5 to exit
2
Element popped is 30
Enter the choice
Enter 1 to insert an element in to stack
Enter 2 to delete an element from the stack
Enter 3 to check an element is palindrome or not
Enter 4 to check status of the stack
Enter 5 to exit
4
The status of the stack is
20
10
Enter the choice
Enter 1 to insert an element in to stack
Enter 2 to delete an element from the stack
Enter 3 to check an element is palindrome or not
Enter 4 to check status of the stack
Enter 5 to exit
3
Enter the element to check whether it is palindrome or not
121
Palindrome
Enter the choice
Enter 1 to insert an element in to stack
Enter 2 to delete an element from the stack
Enter 3 to check an element is palindrome or not
Enter 4 to check status of the stack
Enter 5 to exit
5
[tce123@localhost vb]$
```

PROGRAM -4

Design, Develop and Implement a Program in C for converting an Infix Expression to Postfix Expression. Program should support for both parenthesized and free parenthesized expressions with the operators: +, -, *, /, % (Remainder), ^ (Power) and alphanumeric operands. */

```
#define SIZE 50 /* Size of Stack */
#include <ctype.h>
#include <stdio.h>

char s[SIZE];
int top = -1; /* Global declarations */

push(char elem) /* Function for PUSH operation */
{
        s[++top] = elem;
}

char pop() /* Function for POP operation */
{
        return (s[top--]);
}

int pr(char elem)          /* Function for precedence */
{
        switch (elem)
        {
                case '#': return 0;
                case '(': return 1;
                case '+':
                case '-': return 2;
                case '*':
                case '/':
                case '%': return 3;
                case '^': return 4;
        }
}
void main()              /* Main Program */
{
        char infx[50], pofx[50], ch, elem;
        int i = 0, k = 0;
        printf("\n\n Enter the valid Infix Expression.  ");
        scanf("%s", infx);
```

```
        push('#');
        while ((ch = infx[i++]) != '\0')
        {
                if (ch == '(')
                  push(ch);
                else if (isalnum(ch))
                  pofx[k++] = ch;
                else if (ch == ')')
                {
                  while (s[top] != '(')
                        pofx[k++] = pop();
                  elem = pop(); /* Remove ( */
                }
                else /* Operator */
                {
                  while (pr(s[top]) >= pr(ch))
                        pofx[k++] = pop();
                  push(ch);
                }
        }

        while (s[top] != '#') /* Pop from stack till empty */
                pofx[k++] = pop();
        pofx[k] = '\0'; /* Make pofx as valid string */
        printf("\n\nGiven Infix Expn: %s Postfix Expn: %s\n", infx, pofx);
}
```

OUTPUT

Enter the valid Infix Expression : (a+b)*c/d^5%1
Given Infix Expn: (a+b)*c/d^5%1 Postfix Expn: ab+c*d5^/1%

PROGRAM – 5

Design, develop and Implement a Program in C for the following Stack Applications

a. Evaluation of Suffix expression with single digit operands and operators: +, -, *, /, %, ^
b. Solving Tower of Hanoi problem with n disks

```
#include<stdio.h>
#include<string.h>
#include<math.h>
int count=0, top=-1;
int operate(char symb, int op1, int op2)
{
    switch(symb)
   {
        case '+':return op1+op2;
        case '-':return op1-op2;
        case '/':return op1/op2;
        case '*':return op1*op2;
        case '%':return op1%op2;
        case '^':return pow(op1,op2);
   }
}

void push(int stack[],int d)
{
        stack[++top]=d;
}

int  pop(int stack[])
{
        return(stack[top--]);
}
```

```
void tower( int n,char src, char intr, char des)
{
        if(n)
        {
                tower(n-1,src,des,intr);
                printf("disk %d  moved from %c to %c\n", n, src, des);
                count++;
                tower(n-1,intr,src,des);
                }
        }

void main()
{
  int n, choice,i,op1,op2,ans,stack[50];
  char expr[20],symb;
  while(1)
 {
       printf("\n1. evaluate suffix expression\n 2.Tower of hanoi\n3.Exit\n ");
       printf("\nenter the choice\n");
       scanf("%d",&choice);
       switch(choice)
      {
            case 1: printf("enter the suffix expression : ");
                     scanf("%s",expr);
                     for(i=0;expr[i]!= '\0';i++)
                    {
                       symb=expr[i];
                       if(symb>='0' && symb<='9')   // symb – '0'   => Converts
                            push(stack, symb-'0');        // character symb to its int
                       else                               // form by subtracting
                       {                                  // ASCII value of zero i.e., 48
                            op2=pop(stack);
                            op1=pop(stack);
                            printf("%d %d %c",op2,op1,symb);
                            ans=operate(symb,op1,op2);
                            push(stack,ans);
                       }
```

```
                    }
                    ans=pop(stack);
                    printf("The result of the suffix expression is %d",ans);
                    break;
          case 2: printf("enter the number of disks\n");
                    scanf("%d",&n);
                    tower(n,'a','b','c');
                    printf("number of moves taken to move disks from
                              source to destination %d", count);
                    break;
          case 3:  return;
        }
  }
}
```

OUTPUT

```
[tce123@localhost vb]$ cc -lm lab5.c
[tce123@localhost vb]$ ./a.out

1. evaluate suffix expression
 2.Tower of hanoi
3.Exit

enter the choice
1
enter the suffix expression : 23*52++
3 2 *2 5 +7 6 +The result of the suffix expression is 13
1. evaluate suffix expression
 2.Tower of hanoi
3.Exit

enter the choice
2
enter the number of disks
3
```

```
disk 1  moved from a to c
disk 2  moved from a to b
disk 1  moved from c to b
disk 3  moved from a to c
disk 1  moved from b to a
disk 2  moved from b to c
disk 1  moved from a to c
number of moves taken to move disks from source to destination 7
1. evaluate suffix expression
 2.Tower of hanoi
3.Exit

enter the choice
3
[tce123@localhost vb]$
```

PROGRAM – 6

Design, Develop and Implement a menu driven Program in C for the following operations on **Circular QUEUE** of Characters (Array Implementation of Queue with maximum size **MAX**)

a. Insert an Element on to Circular QUEUE
b. Delete an Element from Circular QUEUE
c. Demonstrate ***Overflow*** and ***Underflow*** situations on Circular QUEUE
d. Display the status of Circular QUEUE
e. Exit

Support the program with appropriate functions for each of the above operations

```
#include<string.h>
#include<stdio.h>
#include<stdlib.h>
#define max 5

int front=0,rear=-1, count=0;
char cq[max];

void cqinsert()
{
        char ele;
        if(count==max)
        {
                printf("Queue is full\n\n");
                return;
        }
        printf("Enter the element to be inserted\n\n");
        scanf("%s",&ele);
        rear=(rear+1)%max;
        cq[rear]=ele;
        count++;
}
```

```
void cqdelete()
{
        if(count==0)
        {
                printf("Queue is empty\n\n");            return;
        }
        printf("Element deleted is = %c \n\n",cq[front]);
        front =(front+1)%max;
        count--;
 }

void cqdisplay()
{
        int j=front,i;
        if(count==0)
        {
                printf("Queue is empty\n\n");      return;
        }

        printf("Circular Queue content is \n\n");
        for(i=1;i<=count;i++)
        {
                printf("%c\t",cq[j]);
                j=(j+1)%max;
        }
        printf("\n\n");
}

void main()
{
int ch;
do
{
                printf("1:insert\t 2:delete\t 3:display\t 4:exit\n\n");
                printf("Enter your choice\n\n");
                scanf("%d",&ch);
                switch(ch)
```

```
            {
                    case 1: cqinsert();
                            break;
                    case 2: cqdelete();
                            break;
                    case 3: cqdisplay();
                            break;
                    case 4: exit(0);
                    default: printf("Invalid choice\n\n");      break;
            }
        }while(1);
}
```

OUTPUT

```
[tce123@localhost vb]$ cc lab6.c
[tce123@localhost vb]$ ./a.out
1:insert        2:delete        3:display       4:exit
Enter your choice
1
Enter the element to be inserted
1
1:insert        2:delete        3:display       4:exit
Enter your choice
1
Enter the element to be inserted
2
1:insert        2:delete        3:display       4:exit
Enter your choice
1
Enter the element to be inserted
3
1:insert        2:delete        3:display       4:exit
Enter your choice
1
Enter the element to be inserted
```

```
4
1:insert        2:delete        3:display       4:exit
Enter your choice
1
Enter the element to be inserted
5
1:insert        2:delete        3:display       4:exit
Enter your choice
1
Queue is full
1:insert        2:delete        3:display       4:exit
Enter your choice
3
Circular Queue content is
1       2       3       4       5
1:insert        2:delete        3:display       4:exit
Enter your choice
2
Element deleted is = 1
1:insert        2:delete        3:display       4:exit
Enter your choice
3
Circular Queue content is
2       3       4       5
1:insert        2:delete        3:display       4:exit
Enter your choice
1
Enter the element to be inserted
6
1:insert        2:delete        3:display       4:exit
Enter your choice
3
Circular Queue content is
2       3       4       5       6
1:insert        2:delete        3:display       4:exit
Enter your choice
4       [tce123@localhost vb]$
```

PROGRAM – 7

Design, Develop and Implement a menu driven Program in C for the following operations on Singly Linked List (SLL) of Student Data with the fields: USN, Name, Programme, Sem, PhNo

a. Create a SLL of N Students Data by using front insertion.
b. Display the status of SLL and count the number of nodes in it
c. Perform Insertion/Deletion at End of SLL
d. Perform Insertion/Deletion at Front of SLL(Demonstration of STACK)
e. Exit

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

struct list
{
        char usn[15],name[20],programme[30],phno[12];
        int  sem;
        struct list *next;
};
typedef struct list node;

node* insertfront(node *first)
{
        node *temp;
        temp=(struct list *)malloc(sizeof(node));
        if(temp!=NULL)
          printf("memory allocated\n");
        printf("Enter the USN,Name,Programme,Sem,Ph.No.\n");
        scanf("%s%s%s%d%s",&temp->usn,&temp->name,&temp->programme,
        &temp->sem, &temp->phno);
        temp->next=NULL;
        if(!first)
                return(temp);
        temp->next=first;
        first=temp;
        return first;
}
```

```
node* delfront(node *first)
{
        int i;
        node *temp=first;
        if(first==NULL)
        {
                printf("List is empty..can't delete\n");
                //getch();
                return first;
        }
        else
        {
                printf("The student record deleted is %s \n",temp->name);
                first=temp->next;
                free(temp);

        }
  return first;
}

void insertend(node *first)
{
        node *ptr,*temp;
        temp=(struct list *)malloc(sizeof(node));
        printf("Enter the USN,Name,Programme,Sem,Ph.No.\n");
        scanf("%s%s%s%d%s",&temp->usn,&temp->name,&temp->programme,
        &temp->sem,&temp->phno);
        temp->next=NULL;
        ptr=first;
        while(ptr->next!=NULL)
            ptr=ptr->next;
        ptr->next=temp;
}

node* delend(node *first)
{
        node *last,*ptr;
```

```
        ptr=last=first;
        if(first==NULL)
        {
                printf("List is empty..can't delete\n");
                //getch();
                return first;
        }
        if(first->next==NULL)
        {
            printf("\nThe record deleted is %s \n",first->name);
            //free(first);
            return NULL;
        }
        while(last->next)
        {
            ptr=last;
            last=last->next;
        }
        printf("\nThe record deleted is %s \n",last->name);
        ptr->next=NULL;
        //free(last);
  return first;
}

void display(node *first)
{
        int count=0;
        node *temp=first;

        if(first==NULL)
        {
            printf("List is Empty : No records to display.. ! \n");
            return;
        }
        while(temp!=NULL)
        {
            count++;
```

```
            printf("\n%s\t%s\t%s\t%d\t%s\n",temp->usn,temp->name, temp->
            programme, temp->sem,temp->phno);
            temp=temp->next;
        }
        printf("\n Total number of records is = %d\n\n",count);
}

void main()
{
   int i,n,choice;
   node *first=NULL;
   //clrscr();
   printf("Singly linked list Implementation .. \n");
   while(1)
   {
     printf("\n 1.Create a List\t 2.Insert front\t 3.Insert end\n\n");
     printf("4.Delete at front\t 5.Delete at end\t 6.Display\t 7.Exit \n\n");
     printf("Enter your choice\n");
     scanf("%d",&choice);
     switch(choice)
     {
          case 1:printf("\n Enter the number of students\n");
                 scanf("%d",&n);
                 for(i=1;i<=n;i++)
                   first=insertfront(first);
                 break;
          case 2:first=insertfront(first);
                 break;
          case 3:insertend(first);
                 break;
          case 4:first=delfront(first);
                 break;
          case 5:first=delend(first);
                 //getch();
                 break;
          case 6:display(first);
                 break;
```

```
            case 7:exit(0);
            default: printf("Invalid choice: Kindly give valid choice\n");
                    break;
          }
    }
}
```

OUTPUT

```
[tce123@localhost final-lab-programs]$ ./a.out
Singly linked list Implementation ..
1.Create a List      2.Insert front      3.Insert end
4.Delete at front    5.Delete at end     6.Display    7.Exit
Enter your choice
1
Enter the number of students
2
memory allocated
Enter the USN,Name,Programme,Sem,Ph.No.
2tg15cs001
Aishwarya
CSE
3
9876543210
memory allocated
Enter the USN,Name, Programme,Sem,Ph.No.
2tg15cs002
Aishwarya-2
CSE
3
0123456789
 1.Create a List      2.Insert front      3.Insert end
4.Delete at front    5.Delete at end     6.Display    7.Exit
Enter your choice
6
2tg15cs002  Aishwarya-2        CSE  3      0123456789
2tg15cs001  Aishwarya          CSE  3      9876543210
```

```
 Total number of records is = 2
 1.Create a List       2.Insert front       3.Insert end
4.Delete at front      5.Delete at end      6.Display     7.Exit
Enter your choice
2
memory allocated
Enter the USN,Name, Programme,Sem,Ph.No.
2tg15cs003
akshata
cse
3
7865434567
1.Create a List       2.Insert front       3.Insert end
4.Delete at front      5.Delete at end      6.Display     7.Exit
Enter your choice
6
2tg15cs003  akshata       cse     3       7865434567
2tg15cs002  Aishwarya-2       CSE   3       0123456789
2tg15cs001  Aishwarya   CSE   3       9876543210
Total number of records is = 3
1.Create a List       2.Insert front       3.Insert end
4.Delete at front      5.Delete at end      6.Display     7.Exit
Enter your choice
3
Enter the USN,Name, Programme,Sem,Ph.No.
2tg15cs004
anjana
cse
3
8976565643
1.Create a List       2.Insert front       3.Insert end
4.Delete at front      5.Delete at end      6.Display     7.Exit
Enter your choice
6
2tg15cs003  akshata       cse     3       7865434567
2tg15cs002  Aishwarya-2       CSE   3       0123456789
2tg15cs001  Aishwarya   CSE   3       9876543210
```

```
2tg15cs004  anjana        cse    3       8976565643
Total number of records is = 4
1.Create a List       2.Insert front       3.Insert end
4.Delete at front     5.Delete at end      6.Display     7.Exit
Enter your choice
4
The student record deleted is akshata
1.Create a List       2.Insert front       3.Insert end
4.Delete at front     5.Delete at end      6.Display     7.Exit
Enter your choice
6
2tg15cs002  Aishwarya-2       CSE   3       0123456789
2tg15cs001  Aishwarya   CSE   3       9876543210
2tg15cs004  anjana        cse    3       8976565643
Total number of records is = 3
1.Create a List       2.Insert front       3.Insert end
4.Delete at front     5.Delete at end      6.Display     7.Exit
Enter your choice
5
The record deleted is anjana
1.Create a List       2.Insert front       3.Insert end
4.Delete at front     5.Delete at end      6.Display     7.Exit
Enter your choice
6
2tg15cs002  Aishwarya-2       CSE   3       0123456789
2tg15cs001  Aishwarya   CSE   3       9876543210
Total number of records is = 2
1.Create a List       2.Insert front       3.Insert end
4.Delete at front     5.Delete at end      6.Display     7.Exit
Enter your choice
8
Invalid choice: Kindly give valid choice
1.Create a List       2.Insert front       3.Insert end
4.Delete at front     5.Delete at end      6.Display     7.Exit
Enter your choice
7
[tce123@localhost final-lab-programs]$
```

PROGRAM – 8

Design, Develop and Implement a menu driven Program in C for the following operations on Doubly Linked List (DLL) of Employee Data with the fields: SSN, Name, Dept, Designation, Sal, PhNo

a. Create a DLL of N Employees Data by using end insertion.
b. Display the status of DLL and count the number of nodes in it
c. Perform Insertion and Deletion at End of DLL
d. Perform Insertion and Deletion at Front of DLL
e. Demonstrate how this DLL can be used as Double Ended Queue
f. Exit

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

struct employee
{
        char ssn[15];
        char name[25];
        char dept[25];
        char desig[20];
        double sal;
        char ph[12];
        struct employee *prev;
        struct employee *next;
};

struct employee * create(struct employee *, int);
void display(struct employee *);
struct employee * insert_front(struct employee *);
struct employee * delete_front(struct employee *);
struct employee * insert_end(struct employee *);
struct employee * delete_end(struct employee *);
void dequeue();

void main()
{
```

```
int choice,n;
struct employee *head=NULL;

printf("\nProgram to illustrate Doubly Linked List operations\n");

while(1)
{
   printf("\nEnter 1=> Create DLL of N employees by using end
   insertion\n\t");
   printf(" 2=> Display the status of DLL and count\n\t");
   printf(" 3=> Perform Insertion at End\n\t 4=> Perform Deletion at
   End\n\t");
  printf(" 5=> Perform Insertion at front\n\t 6=> Perform Deletion at
  Front\n\t");
   printf(" 7=> Demonstration of Dequeue\n\t 8=> Exit\n\t");
   printf("Enter your choice \n");
   scanf("%d",&choice);
   switch(choice)
   {
        case 1: head=NULL;
                printf("Enter the number of Employees\n");
                scanf("%d",&n);
                head=create(head,n);
                break;
        case 2: display(head);
                break;
        case 3: head=insert_end(head);
                break;
        case 4: head=delete_end(head);
                break;
        case 5: head=insert_front(head);
                break;
        case 6: head=delete_front(head);
                break;
        case 7: dequeue();
                break;
        case 8:  return;
```

```
                default: printf("kindly enter valid choice\n");
                        break;
            }
        }
}

struct employee * create(struct employee *head, int n)
{
  int i;
  struct employee *temp,*ptr;
  for(i=0;i<n;i++)
  {
        temp=(struct employee*)malloc(sizeof(struct employee));
        printf("\n Enter the details of Employee %d\n\n",i+1);
        printf("\n SSN, Name, Dept, Designation,Sal, PhNo \n");
        scanf("%s",temp->ssn);
        scanf("%s",temp->name);
        scanf("%s",temp->dept);
        scanf("%s",temp->desig);
        scanf("%lf",&temp->sal);
        scanf("%s",temp->ph);
        temp->prev=temp->next=NULL;
        if(!head)
        {
          head=temp;
          continue;
        }
        ptr=head;
        while(ptr->next) ptr=ptr->next;
            temp->prev=ptr;
        ptr->next=temp;
  }
 return head;
}

void display(struct employee *head)
{
```

```
        struct employee *ptr;
        int count=0;
        for(ptr=head;ptr;ptr=ptr->next)
        {
          count++;
           printf("\nEmployee %d\n\tSSN: %s\n\tNAME: %s\n\tDEPT:
           %s\n\tDESIGNATION: %s\n\tSALARY: %lf\n\tph: %s\n\n",count,
           ptr->ssn,ptr->name,ptr->dept,ptr->desig,ptr->sal,ptr->ph);
        }
        printf("The total number of Employees are %d", count);
}

struct employee * insert_front(struct employee *head)
{
        struct employee *temp;

        temp=(struct employee*)malloc(sizeof(struct employee));
        printf("\nEnter the details of Employee\n\n");
        printf("\n Enterb SSN, Name, Dept, Designation,Sal, PhNo \n");
        scanf("%s%s%s%s%lf%s",temp->ssn,temp->name,temp->dept,
        temp->desig, &temp->sal,temp->ph);
        temp->prev=temp->next=NULL;

        if(!head)
            return temp;

        head->prev=temp;
        temp->next=head;
    return temp;
}

struct employee * delete_front(struct employee *head)
{
        if(!head)
        {
                printf("No records to delete\n");
                return NULL;
```

```
        }
        else if(!head->next)
        {
                return NULL;
        }
        else
                return head->next;
}

struct employee * insert_end(struct employee *head)
{
        struct employee *temp,*ptr;

        temp=(struct employee*)malloc(sizeof(struct employee));
        printf("\n Enter the details of Employee \n\n");
         printf("\n Enter SSN, Name, Dept, Designation,Sal, PhNo \n");
        scanf("%s%s%s%s%lf%s",temp->ssn,temp->name,temp->dept,
         temp->desig,&temp->sal,temp->ph);
        temp->prev=temp->next=NULL;

        if(!head) return temp;

        ptr=head;
        while(ptr->next) ptr=ptr->next;

        temp->prev=ptr;
        ptr->next=temp;

        return head;
}

struct employee * delete_end(struct employee *head)
{
        struct employee *ptr;

        if(!head)
        {
```

```
                printf("No records to delete\n");
                return head;
        }

        if(!head->next)
        return NULL;

      ptr=head;
        while(ptr->next->next)
          ptr=ptr->next;

        free(ptr->next);
        ptr->next=NULL;

        return head;
}

void dequeue()
{
  struct employee *head=NULL;
  int choice;
  printf("\nDemonstration of Double-Ended_Queue operations\n");
  while(1)
  {
    printf("\nEnter 1=> Insert an element to front end\n\t");
    printf("2=> Delete an element from front end\n\t");
    printf("3=> Insert an element to rear end\n\t");
    printf("4=> Delete an element from rear end\n\t");
    printf("5=> Display Deque status\n\t 6=> Exit\n\t");
    printf("Enter your choice\n");
    scanf("%d",&choice);
    switch(choice)
    {
         case 1: head=insert_front(head);
                 break;
         case 2: head=delete_front(head);
                 break;
```

```
            case 3: head=insert_end(head);
                    break;
            case 4: head=delete_end(head);
                    break;
            case 5: display(head);
                    break;
            case 6: return;
           default: printf("Kindly enter valid choice \n");
                     break;
       }
   }
}
```

OUTPUT

```
[tce123@localhost final-lab-programs]$ ./a.out

Program to illustrate Doubly Linked List operations

Enter 1=> Create DLL of N employees by using end insertion
          2=> Display the status of DLL and count
          3=> Perform Insertion at End
          4=> Perform Deletion at End
          5=> Perform Insertion at front
          6=> Perform Deletion at Front
          7=> Demonstration of  Dequeue
          8=> Exit
         Enter your choice
1
Enter the number of Employees
2
 Enter the details of Employee 1
 SSN, Name, Dept, Designation,Sal, PhNo
2051vab
venkatesh
CSE
Asst.Prof.
```

```
90000
9916262655

 Enter the details of Employee 2
 SSN, Name, Dept, Designation,Sal, PhNo
2054apb
Bhandage
CSE
Asst.Prof.
95000
9886503728
Enter 1=> Create DLL of N employees by using end insertion
        2=> Display the status of DLL and count
        3=> Perform Insertion at End
        4=> Perform Deletion at End
        5=> Perform Insertion at front
        6=> Perform Deletion at Front
        7=> Demonstration of Dequeue
        8=> Exit
       Enter your choice
2
Employee 1
        SSN: 2051vab
        NAME: venkatesh
        DEPT: CSE
        DESIGNATION: Asst.Prof.
        SALARY: 90000.000000
        ph: 9916262655
Employee 2
        SSN: 2054apb
        NAME: Bhandage
        DEPT: CSE
        DESIGNATION: Asst.Prof.
        SALARY: 95000.000000
        ph: 9886503728

The total number of Employees are 2
```

```
Enter 1=> Create DLL of N employees by using end insertion
        2=> Display the status of DLL and count
        3=> Perform Insertion at End
        4=> Perform Deletion at End
        5=> Perform Insertion at front
        6=> Perform Deletion at Front
        7=> Demonstration of Dequeue
        8=> Exit
       Enter your choice
3
 Enter the details of Employee
 Enter SSN, Name, Dept, Designation,Sal, PhNo
vIJAYKUMAR
VIJAYAKUMAR
ECE
ASST.PROF.
80000
8976543218

Enter 1=> Create DLL of N employees by using end insertion
        2=> Display the status of DLL and count
        3=> Perform Insertion at End
        4=> Perform Deletion at End
        5=> Perform Insertion at front
        6=> Perform Deletion at Front
        7=> Demonstration of Dequeue
        8=> Exit
       Enter your choice
2
Employee 1
        SSN: 2051vab
        NAME: venkatesh
        DEPT: CSE
        DESIGNATION: Asst.Prof.
        SALARY: 90000.000000
        ph: 9916262655
Employee 2
```

SSN: 2054apb
NAME: Bhandage
DEPT: CSE
DESIGNATION: Asst.Prof.
SALARY: 95000.000000
ph: 9886503728
Employee 3
SSN: vIJAYKUMAR
NAME: VIJAYAKUMAR
DEPT: ECE
DESIGNATION: ASST.PROF.
SALARY: 80000.000000
ph: 8976543218

The total number of Employees are 3
Enter 1=> Create DLL of N employees by using end insertion
2=> Display the status of DLL and count
3=> Perform Insertion at End
4=> Perform Deletion at End
5=> Perform Insertion at front
6=> Perform Deletion at Front
7=> Demonstration of Dequeue
8=> Exit
Enter your choice
5
Enter the details of Employee
Enterb SSN, Name, Dept, Designation,Sal, PhNo
2tg235TGD
Chandrakanth
CSE
asst.prof.
90000
8976567890
Enter 1=> Create DLL of N employees by using end insertion
2=> Display the status of DLL and count
3=> Perform Insertion at End
4=> Perform Deletion at End

```
        5=> Perform Insertion at front
        6=> Perform Deletion at Front
        7=> Demonstration of Dequeue
        8=> Exit
        Enter your choice
2
Employee 1
        SSN: 2tg235TGD
        NAME: Chandrakanth
        DEPT: CSE
        DESIGNATION: asst.prof.
        SALARY: 90000.000000
        ph: 8976567890
Employee 2
        SSN: 2051vab
        NAME: venkatesh
        DEPT: CSE
        DESIGNATION: Asst.Prof.
        SALARY: 90000.000000
        ph: 9916262655
Employee 3
        SSN: 2054apb
        NAME: Bhandage
        DEPT: CSE
        DESIGNATION: Asst.Prof.
        SALARY: 95000.000000
        ph: 9886503728
Employee 4
        SSN: vIJAYKUMAR
        NAME: VIJAYAKUMAR
        DEPT: ECE
        DESIGNATION: ASST.PROF.
        SALARY: 80000.000000
        ph: 8976543218

The total number of Employees are 4
Enter 1=> Create DLL of N employees by using end insertion
```

```
        2=> Display the status of DLL and count
        3=> Perform Insertion at End
        4=> Perform Deletion at End
        5=> Perform Insertion at front
        6=> Perform Deletion at Front
        7=> Demonstration of Dequeue
        8=> Exit
       Enter your choice
4
Enter 1=> Create DLL of N employees by using end insertion
        2=> Display the status of DLL and count
        3=> Perform Insertion at End
        4=> Perform Deletion at End
        5=> Perform Insertion at front
        6=> Perform Deletion at Front
        7=> Demonstration of Dequeue
        8=> Exit
       Enter your choice
2
Employee 1
       SSN: 2tg235TGD
       NAME: Chandrakanth
       DEPT: CSE
       DESIGNATION: asst.prof.
       SALARY: 90000.000000
       ph: 8976567890
Employee 2
       SSN: 2051vab
       NAME: venkatesh
       DEPT: CSE
       DESIGNATION: Asst.Prof.
       SALARY: 90000.000000
       ph: 9916262655
Employee 3
       SSN: 2054apb
       NAME: Bhandage
       DEPT: CSE
```

DESIGNATION: Asst.Prof.
SALARY: 95000.000000
ph: 9886503728
The total number of Employees are 3
Enter 1=> Create DLL of N employees by using end insertion
2=> Display the status of DLL and count
3=> Perform Insertion at End
4=> Perform Deletion at End
5=> Perform Insertion at front
6=> Perform Deletion at Front
7=> Demonstration of Dequeue
8=> Exit
Enter your choice
6
Enter 1=> Create DLL of N employees by using end insertion
2=> Display the status of DLL and count
3=> Perform Insertion at End
4=> Perform Deletion at End
5=> Perform Insertion at front
6=> Perform Deletion at Front
7=> Demonstration of Dequeue
8=> Exit
Enter your choice
2
Employee 1
SSN: 2051vab
NAME: venkatesh
DEPT: CSE
DESIGNATION: Asst.Prof.
SALARY: 90000.000000
ph: 9916262655
Employee 2
SSN: 2054apb
NAME: Bhandage
DEPT: CSE
DESIGNATION: Asst.Prof.
SALARY: 95000.000000

```
        ph: 9886503728

The total number of Employees are 2
Enter 1=> Create DLL of N employees by using end insertion
         2=> Display the status of DLL and count
         3=> Perform Insertion at End
         4=> Perform Deletion at End
         5=> Perform Insertion at front
         6=> Perform Deletion at Front
         7=> Demonstration of Dequeue
         8=> Exit
        Enter your choice
7
Demonstration of Double-Ended_Queue operations

Enter 1=> Insert an element to front end
        2=> Delete an element from front end
        3=> Insert an element to rear end
        4=> Delete an element from rear end
        5=> Display Deque status
         6=> Exit
        Enter your choice
1
Enter the details of Employee
 Enterb SSN, Name, Dept, Designation,Sal, PhNo
The123
Deepak
Business
Leader
500000
9886503728
Enter 1=> Insert an element to front end
        2=> Delete an element from front end
        3=> Insert an element to rear end
        4=> Delete an element from rear end
        5=> Display Deque status
         6=> Exit
```

Enter your choice
1
Enter the details of Employee
Enterb SSN, Name, Dept, Designation,Sal, PhNo
thessn
thename
thedept
thedesig
908978
9876543210
Enter 1=> Insert an element to front end
2=> Delete an element from front end
3=> Insert an element to rear end
4=> Delete an element from rear end
5=> Display Deque status
6=> Exit
Enter your choice
5
Employee 1
SSN: thessn
NAME: thename
DEPT: thedept
DESIGNATION: thedesig
SALARY: 908978.000000
ph: 9876543210
Employee 2
SSN: The123
NAME: Deepak
DEPT: Business
DESIGNATION: Leader
SALARY: 500000.000000
ph: 9886503728
The total number of Employees are 2
Enter 1=> Insert an element to front end
2=> Delete an element from front end
3=> Insert an element to rear end
4=> Delete an element from rear end

```
        5=> Display Deque status
         6=> Exit
        Enter your choice
3
 Enter the details of Employee
 Enter SSN, Name, Dept, Designation,Sal, PhNo
ssn2
name2
dept2
desig2
908978
9089786756
Enter 1=> Insert an element to front end
        2=> Delete an element from front end
        3=> Insert an element to rear end
        4=> Delete an element from rear end
        5=> Display Deque status
         6=> Exit
        Enter your choice
5
Employee 1
        SSN: thessn
        NAME: thename
        DEPT: thedept
        DESIGNATION: thedesig
        SALARY: 908978.000000
        ph: 9876543210
Employee 2
        SSN: The123
        NAME: Deepak
        DEPT: Business
        DESIGNATION: Leader
        SALARY: 500000.000000
        ph: 9886503728
Employee 3
        SSN: ssn2
        NAME: name2
```

```
        DEPT: dept2
        DESIGNATION: desig2
        SALARY: 908978.000000
        ph: 9089786756

The total number of Employees are 3
Enter 1=> Insert an element to front end
        2=> Delete an element from front end
        3=> Insert an element to rear end
        4=> Delete an element from rear end
        5=> Display Deque status
         6=> Exit
        Enter your choice
2
Enter 1=> Insert an element to front end
        2=> Delete an element from front end
        3=> Insert an element to rear end
        4=> Delete an element from rear end
        5=> Display Deque status
         6=> Exit
        Enter your choice
5
Employee 1
        SSN: The123
        NAME: Deepak
        DEPT: Business
        DESIGNATION: Leader
        SALARY: 500000.000000
        ph: 9886503728
Employee 2
        SSN: ssn2
        NAME: name2
        DEPT: dept2
        DESIGNATION: desig2
        SALARY: 908978.000000
        ph: 9089786756
The total number of Employees are 2
```

Enter 1=> Insert an element to front end
2=> Delete an element from front end
3=> Insert an element to rear end
4=> Delete an element from rear end
5=> Display Deque status
6=> Exit
Enter your choice
4
Enter 1=> Insert an element to front end
2=> Delete an element from front end
3=> Insert an element to rear end
4=> Delete an element from rear end
5=> Display Deque status
6=> Exit
Enter your choice
5
Employee 1
SSN: The123
NAME: Deepak
DEPT: Business
DESIGNATION: Leader
SALARY: 500000.000000
ph: 9886503728
The total number of Employees are 1
Enter 1=> Insert an element to front end
2=> Delete an element from front end
3=> Insert an element to rear end
4=> Delete an element from rear end
5=> Display Deque status
6=> Exit
Enter your choice
7
Kindly enter valid choice
Enter 1=> Insert an element to front end
2=> Delete an element from front end
3=> Insert an element to rear end
4=> Delete an element from rear end

```
        5=> Display Deque status
         6=> Exit
        Enter your choice
6
Enter 1=> Create DLL of N employees by using end insertion
         2=> Display the status of DLL and count
         3=> Perform Insertion at End
         4=> Perform Deletion at End
         5=> Perform Insertion at front
         6=> Perform Deletion at Front
         7=> Demonstration of Dequeue
         8=> Exit
        Enter your choice
9
kindly enter valid choice
8
[tce123@localhost final-lab-programs]$
```

PROGRAM-9

Design, Develop and Implement a Program in C for the following operations on **Singly Circular Linked List (SCLL)** with header nodes

a. Represent and Evaluate a Polynomial

$P(x, y, z) = 6x^2y^2z-4yz^5+3x^3yz+2xy^5z- 2xyz^3$

b. Find the sum of two polynomials **POLY1(x, y, z)** and **POLY2(x, y, z)** and store the result in **POLYSUM(x, y, z)**

Support the program with appropriate functions for each of the above operations.

```
#include<stdio.h>
#include<stdlib.h>
struct node
{
    int coeff;
    int expon;
    struct node *link;
};
typedef struct node *NODE;

NODE attach(int coeff,int expon,NODE head)
{
    NODE temp,cur;
    temp=(NODE)malloc(sizeof(struct node));
    temp->coeff=coeff;
    temp->expon=expon;
    cur=head->link;
    while(cur->link!=head)
    {
        cur=cur->link;
    }
    cur->link=temp;
    temp->link=head;

    return head;
}
```

```
NODE read_poly(NODE head)
{
    int i=1, coeff, expon;
    printf("Enter the coefficient as -999 to end the polynomial\n");

    while(1)
    {
        printf("Enter the %d term \n",i++);
        printf("\n Enter the coefficient : ");
        scanf("%d", &coeff);
        if( coeff == -999)   break;
        printf("\n Enter the Exponent : ");
        scanf("%d", &expon);
        head=attach(coeff,expon,head);
    }
    return head;
}

void display(NODE head)
{
    NODE temp;
    if(head->link==head)
    {
        printf("Empty polynomial\n");
        return;
    }
    temp=head->link;
    while(temp!=head)
    {
        if(temp->coeff < 0)  // display -ve coefficient
            printf("%2dx^%2d",temp->coeff,temp->expon);
        else
            printf("+%2dx^%2d",temp->coeff,temp->expon);

        temp=temp->link;
    }
 }
```

```
NODE poly_add(NODE head1,NODE head2,NODE head3)
{
    NODE a,b;
    int coeff;
    a=head1->link;
    b=head2->link;
    while(a!=head1 && b!= head2)
    {
        if(a->expon==b->expon)
        {
            coeff=a->coeff+b->coeff;
            if(coeff!=0)
                head3=attach(coeff,a->expon,head3);

            a=a->link;
            b=b->link;
        }
        else if(a->expon>b->expon)
        {
            head3=attach(a->coeff,a->expon,head3);
            a=a->link;
        }
        else
        {
            head3=attach(b->coeff,b->expon,head3);
            b=b->link;
        }
    } // End of while

    while(a!=head1)
    {
            head3=attach(a->coeff,a->expon,head3);
            a=a->link;
    }
    while(b!=head2)
    {
            head3=attach(b->coeff,b->expon,head3);
```

```
            b=b->link;
    }
  return head3;
}

int eval(NODE head)
{
      NODE p;
      int x,ans=0,t,t1,t2;
      printf("\n\nEnter the value of x=");
      scanf("%d",&x);
      p=head->link;
      do
      {
            //t=p->coeff;
            //t1=p->expon;
            //t2=pow(x,t1);
            //ans=ans+(t*t2);
            ans=ans+p->coeff*pow(x,p->expon);
            p=p->link;
      }
      while(p!=head->link);
   return(ans);
}

void main()
{
    int a,x,ch;
    NODE head1,head2,head3,head4;

    head1=(NODE)malloc(sizeof(struct node));
    head2=(NODE)malloc(sizeof(struct node));
    head3=(NODE)malloc(sizeof(struct node));
    head4=(NODE)malloc(sizeof(struct node));

    head1->link=head1;
    head2->link=head2;
```

```
head3->link=head3;
head4->link=head4;

while(1)
{
 printf("\n\t------------<< MENU >>-----------");
 printf("\n\t Polynomial Operations :");
 printf(" \n\t 1.Addition of  two polynomials");
 printf("\n\t  2.Evaluate a polynomial");
 printf("\n\t  3.Exit");
 printf("\n\t -------------------------------- ");
 printf("\n\n\t Enter your choice==>");
 scanf("%d",&ch);
 switch(ch)
 {
        case 1:printf("\n Enter 1st polunomial:");
               head1=read_poly(head1);
               printf("\n Enter 2nd polynomial:");
               head2=read_poly(head2);
               printf("\n First polynomial is : ");
               display(head1);
               printf("\n Second polynomial is :");
               display(head2);
               head3=poly_add(head1,head2,head3);
               printf("\n Added polynomial is :");
               display(head3);
               break;
         case 2:printf(“Enter the polynomial to be added \n”);
                head4=read_poly(head4);
                printf("Entered polynomial is : \n");
                display(head4);
                a=eval(head4);
                printf("\n\nValue of polynomial is = %d",a);
                break;
         case 3:exit(0);
                break;
        default:printf("\n\n\t Invalid choice");
```

```
                    break;
            }
        }
}
```

OUTPUT

```
[tce123@localhost final-lab-programs]$ ./a.out
        ------------<< MENU >>-----------
        Polynomial Operations :
                                1.Addition of two polynomials
                                2.Evaluate a polynomial
                                3.Exit
        --------------------------------
        Enter your choice==>1
Enter 1st polynomial:Enter the coefficient as -999 to end the polynomial
Enter the 1 term
enter coeff:3
enter power:2
Enter the 2 term
enter coeff:2
enter power:1
Enter the 3 term
enter coeff:-999
Enter 2nd polynomial:Enter the coefficient as -999 to end the polynomial
Enter the 1 term
enter coeff:4
enter power:2
Enter the 2 term
enter coeff:2
enter power:0
Enter the 3 term
enter coeff:-999
1st polynimial is: + 3x^ 2+ 2x^ 1
2nd polynomial is: + 4x^ 2+ 2x^ 0
 Added polynomial: + 7x^ 2+ 2x^ 1+ 2x^ 0
```

```
        ------------<< MENU >>-----------
        Polynomial Operations :
                                1.Addition of two polynomials
                                2.Evaluate a polynomial
                                3.Exit
        --------------------------------
Enter your choice==>2
Enter the coefficient as -999 to end the polynomial
Enter the 1 term
 enter coeff:3
 enter power:2
Enter the 2 term
 enter coeff:2
 enter power:1
Enter the 3 term
 enter coeff:-999
Entered polynomial is
+ 3x^ 2+ 2x^ 1
Enter the value of x=2
Value of polynomial=16
        ------------<< MENU >>-----------
        Polynomial Operations :
          1.Addition of two polynomials
                                2.Evaluate a polynomial
                                3.Exit
        --------------------------------
Enter your choice==>3
[tce123@localhost final-lab-programs]$
```

PROGRAM-10

Design, Develop and Implement a menu driven Program in C for the following operations on Binary Search Tree (BST) of Integers

a. Create a BST of N Integers: 6, 9, 5, 2, 8, 15, 24, 14, 7, 8, 5, 2
b. Traverse the BST in Inorder, Preorder and Post Order
c. Search the BST for a given element (KEY) and report the appropriate message
d. Exit

```
//Binary Search Tree

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

struct tree
{
        int data;
        struct tree *left;
        struct tree *right;
};

struct tree * create(int);
void traverse(struct tree*);
void inorder(struct tree *);
void preorder(struct tree *);
void postorder(struct tree *);
struct tree * search(struct tree*,int);

void main()
{
     int choice, n, key, element;
     struct tree *root=NULL,*temp;
     clrscr();
     printf("\nProgram to illustrate Binary Search Tree operations\n");
     while(1)
     {
```

```
        printf("\nEnter \t 1=> Create BST of N integers\n\t ");
        printf("2=> Traverse the BST in Inorder, Preorder, and Postorder\n\t");
        printf("3=> Search the BST for a Key element\n\t 4=> Exit\n\n\t");
        scanf("%d",&choice);
        switch(choice)
        {
          case 1: printf("Enter the number of elements of BST\n");
                  scanf("%d",&n);
                  root=create(n);
                  break;

          case 2: traverse(root);
                 break;

          case 3: printf("\nEnter the Key element to be searched in BST\n");
                  scanf("%d",&key);
                  temp=search(root,key);
                  if(temp)
                          printf("The Key element is found\n");
                  else
                          printf("The key element is not found\n");
                 break;
          case 4: return;
        }
    }
}

struct tree *create(int n)
{
        struct tree *root=NULL,*prev,*cur,*temp;
        int i, ele;

        printf("Enter the Elements\n");
        for (i=0;i<n;i++)
        {
                scanf("%d",&ele);
                temp=(struct tree *)malloc(sizeof(struct tree));
```

```
                temp->data=ele;
                temp->left=temp->right=NULL;

                if(!root)
                {
                        root=temp;
                        continue;
                }
                prev=cur=root;
                do
                {
                        prev=cur;
                        if (prev->data>ele)
                                cur=prev->left;
                        else
                                cur=prev->right;
                } while(cur);

                if(prev->data>ele)
                        prev->left=temp;
                else
                        prev->right=temp;
        }
    return root;
}

void traverse(struct tree *root)
{
        printf("\nThe Inorder Traversal:\t");
        inorder(root);
        printf("\nThe Preorder Traversal:\t");
        preorder(root);
        printf("\nThe Postorder Traversal:\t");
        postorder(root);
}

void inorder(struct tree *root)
```

```
{
        struct tree *temp=root;
        if(temp)
        {
                inorder(temp->left);
                printf("%d\t",temp->data);
                inorder(temp->right);
        }
}

void preorder(struct tree *root)
{
        struct tree *temp=root;
        if(temp)
        {
                printf("%d\t",temp->data);
                preorder(temp->left);
                preorder(temp->right);
        }
}

void postorder(struct tree *root)
{
        struct tree *temp=root;
        if(temp)
        {
                postorder(temp->left);
                postorder(temp->right);
                printf("%d\t",temp->data);
        }
}

struct tree* search(struct tree *root,int key)
{
        struct tree* temp=root;
        while(temp)
        {
```

```
            if(temp->data==key)
                return temp;
            else if(temp->data>key)
                temp=temp->left;
            else
                temp=temp->right;
        }
        return temp;
}
```

OUTPUT

```
Program to illustrate Binary Search Tree operations
Enter   1=> Create BST of N integers
        2=> Traverse the BST in Inorder, Preorder, and Postorder
        3=> Search the BST for a Key element
        4=> Exit

        1
Enter the number of elements of BST
12
Enter the Elements
6
9
5
2
8
15
24
14
7
8
5
2

Enter   1=> Create BST of N integers
```

2=> Traverse the BST in Inorder, Preorder, and Postorder
3=> Search the BST for a Key element
4=> Exit

2

The Inorder Traversal: 2 2 5 5 6 7 8 8 9 14 15 24

The Preorder Traversal: 6 5 2 2 5 9 8 7 8 15 14 24

The Postorder Traversal: 2 2 5 5 7 8 8 14 24 15 9 6

Enter 1=> Create BST of N integers
2=> Traverse the BST in Inorder, Preorder, and Postorder
3=> Search the BST for a Key element
4=> Exit
4

PROGRAM-11

Design, Develop and Implement a Program in C for the following operations on **Graph (G)** of Cities

a. Create a Graph of **N** cities using Adjacency Matrix.
b. Print all the nodes **reachable** from a given starting node in a digraph using DFS/BFS method.

```
#include<stdio.h>
void dfs(int src, int adj[10][10],int visited[10],int n)
{
        int k;
        visited[src]=1;
        printf("%d",src);
        for(k=0;k<n;k++)
        {
                if(adj[src][k]==1 && visited[k]==0)
                {
                        dfs(k,adj,visited,n);
                }
        }
}

void bfs(int src, int adj[10][10],int n)
{
        int q[20], front=0,rear=-1,v,u,visited[10]={0};
        q[++rear]=src;
        visited[src]=1;
        printf("%d",src);
        while(front<=rear)
        {
                u=q[front++];
                for(v=0;v<n;v++)
                {
                        if(adj[u][v]==1 && visited[v]==0)
                        {
                                q[++rear]=v;
```

```
                        printf("%d",v);
                        visited[v]=1;
                    }
                }
            }
}

void main( )
{
    int choice, i,j,src,flag=0;
    int adj[10][10],visited[10]={0},n;
    clrscr();
    while(1)
    {
        printf("\n\n\t program to perform graph operations\n\n");
        printf("\n\t 1=> Create a graph of  N cities\n\t “);
        printf(“\n\t 2=> To print reachable nodes from source using BFS\n”);
        printf(“\n\t 3=> To check graph connected or not using DFS\n\n");
        printf(“\n\t 4=> Exit \n”);
        printf("\n Enter the choice\n");          scanf("%d",&choice);
        switch(choice)
        {
            case 1: printf("\nenter the number of cities\n");
                    scanf("%d",&n);
                    printf("\nenter the adjacency matrix\n");
                    for(i=0;i<n;i++)
                        for(j=0;j<n;j++)
                            scanf("%d",&adj[i][j]);
                    break;
            case 2: printf("Enter the source vertex to start traversal\n");
                    scanf("%d",&src);
                    printf("vertices visited are\n");
                    bfs(src,adj,n);
                    break;
            case 3: printf("Enter the source vertex to start traversal\n");
                    scanf("%d",&src);
                    for(i=0;i<n;i++)
```

```
                        visited[i]=0;
                    dfs(src,adj,visited,n);
                    for(i=0;i<n;i++)
                        if(visited[i]==0)       flag=1;
                    if(flag==1)
                    printf("the graph is not connected\n");
                        else
                    printf("the graph is connected\n");
                    break;
                case 4: exit();
            }
        }
}
```

OUTPUT

```
[tce123@localhost vb]$ ./a.out
        Program to perform graph operations
        1=> Create a graph of  N cities
        2=> To print reachable nodes from source using BFS
        3=> To check graph connected or not using DFS
        4=> Exit
 Enter the choice
1
enter the number of cities
4
enter the adjacency matrix
0
1
1
1
1
0
0
1
1
0
0
```

```
1
1
1
1
0
        program to perform graph operations
        1=> Create a graph of  N cities
         2=> To print reachable nodes from source using BFS
        3=> To check graph connected or not using DFS
        4=> Exit
 Enter the choice
2
Enter the source vertex to start traversal
0
vertices visited are
0123
        program to perform graph operations
        1=> Create a graph of  N cities
         2=> To print reachable nodes from source using BFS
        3=> To check graph connected or not using DFS
        4=> Exit

 Enter the choice
3
Enter the source vertex to start traversal
0
0132the graph is connected

        program to perform graph operations

        1=> Create a graph of  N cities
         2=> To print reachable nodes from source using BFS
        3=> To check graph connected or not using DFS
        4=> Exit
 Enter the choice
4
[tce123@localhost vb]$
```

PROGRAM-12

Given a File of **N** employee records with a set **K** of Keys (4-digit) which uniquely determine the records in file **F**. Assume that file **F** is maintained in memory by a Hash Table(HT) of **m** memory locations with **L** as the set of memory addresses (2-digit) of locations in HT. Let the keys in **K** and addresses in **L** are Integers. Design and develop a Program in C that uses Hash function **H: K ®L** as H(**K**)=**K** mod **m (remainder** method), and implement hashing technique to map a given key **K** to the address space **L.** Resolve the collision (if any) using **linear probing**.

```
#include <stdio.h>
#define m 10              // Hash table length

//Function declarations
void store(int key, int hk);
void display();
int HT[100];              // Hash Table
int main()
{
    int key;              // Key       (4 digit)
    int hk;               // Hash key (2 digit)
    int ch, i;
    //No elements stored
    for(i = 0; i < m; i++)
        HT[i] = -1;
    for(;;)
    {
        printf("\n1. Insert Key 2.Display 3.Exit\n");
        printf("Choice: ");
        scanf("%d", &ch);
        switch( ch )
        {
            case 1:  //Read key
                     printf("Enter Key: ");
                     scanf("%4d", &key);
                     //generate hash key
```

```
                    hk = key % m;
                    store(key, hk);
                    break;
            case 2: display();
                    break;
            default: printf("\n Thank You ");
                    return;
        }     //end switch
    }         //end for
}             //end main

//Function to store a key in hash table
void store(int key, int hk)
{
    int i;
    int pos = -1;
    int flag = 0;
    if(HT[hk] == -1)
    {
        HT[hk] = key;
        return;
    }
    //linear probing (next location to end of array)
    for(i = hk + 1; i < m; i++)
    {
        if(HT[i] == -1)
        {
            pos = i;
            flag = 1;
            break;
        }
    }

    //if position not found, search from begining of hash table
    if(flag == 0)
    {
        for(i = 0; i < hk; i++)
```

```
            {
                if(HT[i] == -1)
                {
                    pos = i;
                    flag = 1;
                    break;
                }
            }
        }
        if(flag == 0)
            printf("Hash Table Overflow.\n\n");
        else
            HT[pos] = key;
}     //end store

//Function to display hash table
void display()
{
        int i;
        printf("\nHash Table:\n");
        for(i = 0; i < m; i++)
        {
            printf(" %2d|%4d|\n",i, HT[i]);
            //if(i%5 == 0) printf("\n\n");
        }
}           //end display
```

OUTPUT

```
1. Insert Key 2.Display 3.Exit
Choice: 2
Hash Table:
 0|  -1|
 1|  -1|
 2|  -1|
 3|  -1|
```

4| -1|
5| -1|
6| -1|
7| -1|
8| -1|
9| -1|

1. Insert Key 2.Display 3.Exit
Choice: 1
Enter Key: 1005

1. Insert Key 2.Display 3.Exit
Choice: 2

Hash Table:
0| -1|
1| -1|
2| -1|
3| -1|
4| -1|
5|1005|
6| -1|
7| -1|
8| -1|
9| -1|

1. Insert Key 2.Display 3.Exit
Choice: 1
Enter Key: 2009

1. Insert Key 2.Display 3.Exit
Choice: 1
Enter Key: 4857

1. Insert Key 2.Display 3.Exit
Choice: 2

Hash Table:
0| -1|
1| -1|
2| -1|
3| -1|
4| -1|
5|1005|
6| -1|
7|4857|
8| -1|
9|2009|

1. Insert Key 2.Display 3.Exit
Choice: 1
Enter Key: 2344

1. Insert Key 2.Display 3.Exit
Choice: 1
Enter Key: 7676

1. Insert Key 2.Display 3.Exit
Choice: 2
Hash Table:
0| -1|
1| -1|
2| -1|
3| -1|
4|2344|
5|1005|
6|7676|
7|4857|
8| -1|
9|2009|

1. Insert Key 2.Display 3.Exit
Choice: 1
Enter Key: 2005

1. Insert Key 2.Display 3.Exit
Choice: 2

Hash Table:
0| -1|
1| -1|
2| -1|
3| -1|
4|2344|
5|1005|
6|7676|
7|4857|
8|2005|
9|2009|

1. Insert Key 2.Display 3.Exit
Choice: 1
Enter Key: 2008

1. Insert Key 2.Display 3.Exit
Choice: 2

Hash Table:
0|2008|
1| -1|
2| -1|
3| -1|
4|2344|
5|1005|
6|7676|
7|4857|
8|2005|
9|2009|

1. Insert Key 2.Display 3.Exit
Choice: 1

Enter Key: 4973

1. Insert Key 2.Display 3.Exit
Choice: 2

Hash Table:
0|2008|
1| -1|
2| -1|
3|4973|
4|2344|
5|1005|
6|7676|
7|4857|
8|2005|
9|2009|

1. Insert Key 2.Display 3.Exit
Choice: 1
Enter Key: 5342

1. Insert Key 2.Display 3.Exit
Choice: 2
Hash Table:
0|2008|
1| -1|
2|5342|
3|4973|
4|2344|
5|1005|
6|7676|
7|4857|
8|2005|
9|2009|

1. Insert Key 2.Display 3.Exit
Choice: 1

Enter Key: 6453
1. Insert Key 2.Display 3.Exit
Choice: 2
Hash Table:
0|2008|
1|6453|
2|5342|
3|4973|
4|2344|
5|1005|
6|7676|
7|4857|
8|2005|
9|2009|

1. Insert Key 2.Display 3.Exit
Choice: 1
Enter Key: 2345
Hash Table Overflow.

1. Insert Key 2.Display 3.Exit
Choice: 2
Hash Table:
0|2008|
1|6453|
2|5342|
3|4973|
4|2344|
5|1005|
6|7676|
7|4857|
8|2005|
9|2009|

1. Insert Key 2.Display 3.Exit
Choice: 3
Thank You