DATA STRUCTURES LABORATORY

SEMESTER – III

Course Code	**BCSL305**	**CIE Marks**	50
Number of Contact Hours/Week	0:0:2	**SEE Marks**	50
Total Number of Lab Contact Hours	28	**Exam Hours**	03

Course Learning Objectives:

This laboratory course enables students to get practical experience in design, develop, implement, analyzeand evaluation/testing of

- Dynamic memory management
- Linear data structures and their applications such as stacks, queues and lists
- Non-Linear data structures and their applications such as trees and graphs

Descriptions (if any):

- Implement all the programs in "C " Programming Language and Linux OS.

Programs List

1. Develop a Program in C for the following:
    a. Declare a calendar as an array of 7 elements (A dynamically Created array) to represent 7 days of a week. Each Element of the array is a structure having three fields. The first field is the name of the Day (A dynamically allocated String), The second field is the date of the Day (A integer), the third field is the description of the activity for a particular day (A dynamically allocated String).
    b. Write functions create(), read() and display(); to create the calendar, to read the data from the keyboard and to print weeks activity details report on screen.

2. Develop 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

3. Develop 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

4. Develop 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 alphanumericoperands.

5. Develop 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

6. Develop 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

7. Develop 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

8. Develop 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

9. Develop a Program in C for the following operationson 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

10. Develop 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

11. Develop 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

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. 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.

Laboratory Outcomes: The student should be able to:

- Analyze various linear and non-linear data structures
- Demonstrate the working nature of different types of data structures and their applications
- Use appropriate searching and sorting algorithms for the give scenario.
- Apply the appropriate data structure for solving real world problems

Conduct of Practical Examination:

- Experiment distribution
  - For laboratories having only one part: Students are allowed to pick one experiment from the lot with equal opportunity.
  - For laboratories having PART A and PART B: Students are allowed to pick one experiment from PART A and one experiment from PART B, with equal opportunity.
- Change of experiment is allowed only once and marks allotted for procedure to be made zero of the changed part only.
- Marks Distribution *(Need to change in accordance with university regulations)*

  c) For laboratories having only one part – Procedure + Execution + Viva-Voce: 15+70+15 = 100 Marks

  d) For laboratories having PART A and PART B

  i. Part A – Procedure + Execution + Viva = 6 + 28 + 6 = 40 Marks

  ii. Part B – Procedure + Execution + Viva = 9 + 42 + 9 = 60 Marks

PROGRAM 1

Develop a Program in C for the following:

a. **Declare a calendar as an array of 7 elements (A dynamically Created array) to represent 7 days of a week. Each Element of the array is a structure having three fields. The first field is the name of the Day (A dynamically allocated String), The second field is the date of the Day (A integer), the third field is the description of the activity for a particular day (A dynamically allocated String).**

b. **Write functions create(), read() and display(); to create the calendar, to read the data from the keyboard and to print weeks activity details report on screen.**

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

#define NUM_DAYS_IN_WEEK 7

// Structure to represent a day
typedef struct
{
   char *acDayName;    // Dynamically allocated string for the day name
   int iDate;          // Date of the day
   char *acActivity;   // Dynamically allocated string for the activity description
}DAYTYPE;

void fnFreeCal(DAYTYPE *);
void fnDispCal(DAYTYPE *);
void fnReadCal(DAYTYPE *);
DAYTYPE *fnCreateCal();

int main()
{
   // Create the calendar
   DAYTYPE *weeklyCalendar = fnCreateCal();
   // Read data from the keyboard
   fnReadCal(weeklyCalendar);
   // Display the week's activity details
   fnDispCal(weeklyCalendar);
   // Free allocated memory
   fnFreeCal(weeklyCalendar);
   return 0;
}

DAYTYPE *fnCreateCal()
{
   DAYTYPE *calendar = (DAYTYPE *)malloc(NUM_DAYS_IN_WEEK * sizeof(DAYTYPE));


   for(int i = 0; i < NUM_DAYS_IN_WEEK; i++)
          {
      calendar[i].acDayName = NULL;
      calendar[i].iDate = 0;
      calendar[i].acActivity = NULL;
   }

   return calendar;
}

void fnReadCal(DAYTYPE *calendar)
{
          char cChoice;
   for(int i = 0; i < NUM_DAYS_IN_WEEK; i++)
          {
      printf("Do you want to enter details for day %d [Y/N]: ", i + 1);
      scanf("%c", &cChoice); getchar();

      if(tolower(cChoice) == 'n')
          continue;

      printf("Day Name: ");
      char nameBuffer[50];
      scanf("%s", nameBuffer);
      calendar[i].acDayName = strdup(nameBuffer);  // Dynamically allocate and copy
the string
      printf("Date: ");
      scanf("%d", &calendar[i].iDate);
      printf("Activity: ");
      char activityBuffer[100];
      scanf(" %[^n]", activityBuffer);  // Read the entire line, including spaces
      calendar[i].acActivity = strdup(activityBuffer);
      printf("\n");
```

```
    getchar();                    //remove trailing enter character in input buffer
  }
}
void fnDispCal(DAYTYPE *calendar)
{
  printf("nWeek's Activity Details:n");
  for(int i = 0; i < NUM_DAYS_IN_WEEK; i++)
      {
    printf("Day %d:n", i + 1);
            if(calendar[i].iDate == 0)
            {
                  printf("No Activityn\n");
                  continue;
            }

    printf("  Day Name: %sn", calendar[i].acDayName);
    printf("  Date: %dn", calendar[i].iDate);
    printf("  Activity: %snn", calendar[i].acActivity);
  }
}
void fnFreeCal(DAYTYPE *calendar)
{
  for(int i = 0; i < NUM_DAYS_IN_WEEK; i++)
      {
    free(calendar[i].acDayName);
    free(calendar[i].acActivity);
  }
  free(calendar);
}
```

Output

```
Do you want to enter details for day 1 [Y/N]: N
Do you want to enter details for day 2 [Y/N]: Y
Day Name: Monday
Date: 10
Activity: Meeting with Chairman.

Do you want to enter details for day 3 [Y/N]: N
Do you want to enter details for day 4 [Y/N]: N
Do you want to enter details for day 5 [Y/N]: Y
Day Name: Thursday
Date: 13
```

```
Activity: Product Survey

Do you want to enter details for day 6 [Y/N]: Y
Day Name: Friday
Date: 14
Activity: Budget Breakdown and Planning

Do you want to enter details for day 7 [Y/N]: Y
Day Name: Saturday
Date: 15
Activity: Outing with family

Week's Activity Details:
Day 1:
No Activity
Day 2:
  Day Name: Monday
  Date: 10
  Activity: Meeting with Chairman.
Day 3:
No Activity
Day 4:
No Activity
Day 5:
  Day Name: Thursday
  Date: 13
  Activity: Product Survey
Day 6:
  Day Name: Friday
  Date: 14
  Activity: Budget Breakdown and Planning
Day 7:
  Day Name: Saturday
  Date: 15
  Activity: Outing with family
```

PROGRAM 2

Develop 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 read();
void replace(char *,char *,char *);
char STR[100],PAT[100],REP[100];
int len=0;
void main()
{
clrscr();
read();
getch();
}
void read ()
{
int i;
printf("\n enter the main string :\n");
gets(STR);
printf("enter the pattern string :\n");
gets(PAT);
      if(STR[0]=='\0'||PAT[0]=='\0')
          printf("string is empty\n");
     else
        {
         for(i=0;PAT[i]!='\0';i++) len++;
           if(find(STR,PAT)==1)
              {
                          printf("enter the replace string:\n");
               gets(REP);
               replace(STR,PAT,REP);
              }
          else
              printf("pattern is not found\n");
```

```
            }
}
int find(char *s,char *p)
{
int i=0,m=0;
while(s[i]!='\0')
        {
            if(s[i]==p[m])
                m++;
            else
               m=0;
            if(m==len) break;
                i++;
         }
if(m==len)
         return 1;
else
        return 0;
}
void replace(char *s,char *p,char *r)
{
int k,i=0,m=0,c=0,j=0;
char temp[100];
while(s[c]!='\0')
        {
            if(s[m]==p[i])
                    {
                      i++;
                      m++;
                           if(p[i]=='\0')
                                       {
                                         for(k=0;r[k]!='\0';k++,j++)
                                          temp[j]=r[k];
                                          i=0;
                                         c=m;
                                       }
                       }
             else
                    {
                     temp[j]=s[c];
                     j++;
                    c++;
                    m=c;
                    i=0;
                  }
         }
temp[j]='\0';
printf("\n the resultant string is %s",temp);
}
```

Output 1:

```
enter the main string :
to be or not to be
enter the pattern string :
to
enter the replace string:
we
 the resultant string is:
 we be or not we be
```

Output 2:

```
enter the main string :
hello world good morning
enter the pattern string :
string is empty
```

Output 3:

```
enter the main string : hello world good morning
enter the pattern string :evening
pattern is not found
```

PROGRAM 3

3. Develop 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<conio.h>
#define MAX 5
int stack[MAX],top;
void main()
{
int ele,choice;
void push(int);
int pop();
void display();
void palin();
clrscr();
top=-1;
for(;;)
    {
     printf("1-push\n2-pop\n3-display\n4-palindrome\n5-exit\n");
     printf("enter the choice\n");
     scanf("%d",&choice);
        switch(choice)
              {
               case 1:push(ele);
                      break;
              case 2:ele=pop();
                        if(ele!=0)
                        printf("The element deleted from stack is %d\n",ele);
                      break;
              case 3:display();
                      break;
              case 4:palin();
                      break;
              case 5:exit(0);
```

```
                 default:printf("invalid choice\n");
                        break;
                 }
      }
}
void push(int element)
{
     if(top==MAX-1)
       printf("the stack is overflow\n");
     else
       {
        top++;
        printf("enter the element to be pushed into the stack\n");
        scanf("%d",&element);
        stack[top]=element;
       }
}
int pop()
{
 char element;
   if(top==-1)
        {
          printf("stack is underflow\n");
          return 0;
        }
   else
       {
         element=stack[top];
          top--;
         return element;
      }
}
void display()
{
int i;
      if(top==-1)
      printf("the stack is empty\n");
      else
       {
         printf("the contents of the stack:\n");
         for(i=top;i>=0;i--)
             printf("%d\n",stack[i]);
       }
```

```
}
void palin()
{
char s[20],st[20];
int i,j,flag=1;
char ele;
top=-1;
printf("\nEnter the string:\n");
fflush(stdin);
gets(st);
for(i=0;st[i]!='\0';i++)
        {
           s[++top]=st[i];
        }
printf("\nThe contents of the stack are:\n");
         for(i=top;i>=0;i--)
                  printf("%c\n",st[i]);
         for(i=0;st[i]!='\0';i++)
                   if(st[i]!=s[top--])
                       {
                         flag=0;
                          break;
                       }
if(flag==0)
         printf("string is not palindrome\n");
else
          printf("string is palindrome\n");
}
```

OUTPUT:

```
1-push
2-pop
3-display
4-palindrome
5-exit
enter the choice
3
the stack is empty
1-push
2-pop
3-display
4-palindrome
```

```
5-exit
enter the choice
2
stack is underflow
1-push
2-pop
3-display
4-palindrome
5-exit
enter the choice
1
enter the element to be pushed into the stack
10
1-push
2-pop
3-display
4-palindrome
5-exit
enter the choice
1
enter the element to be pushed into the stack
20
1-push
2-pop
3-display
4-palindrome
5-exit
enter the choice
1
enter the element to be pushed into the stack
30
1-push
2-pop
3-display
4-palindrome
5-exit
enter the choice
1
enter the element to be pushed into the stack
40
1-push
2-pop
3-display
```

```
4-palindrome
5-exit
enter the choice
1
enter the element to be pushed into the stack
50
1-push
2-pop
3-display
4-palindrome
5-exit
enter the choice
1
the stack is overflow
1-push
2-pop
3-display
4-palindrome
5-exit
enter the choice
3
the contents of the stack:
50
40
30
20
10
1-push
2-pop
3-display
4-palindrome
5-exit
enter the choice
2
The element deleted from stack is 50
1-push
2-pop
3-display
4-palindrome
5-exit
enter the choice
2
The element deleted from stack is 40
```

```
1-push
2-pop
3-display
4-palindrome
5-exit
enter the choice
3
the contents of the stack:
30
20
10
1-push
2-pop
3-display
4-palindrome
5-exit
enter the choice
4
Enter the string:
madam
The contents of the stack are:
m
a
d
a
m
string is palindrome
1-push
2-pop
3-display
4-palindrome
5-exit
enter the choice
4
Enter the string:
hello
The contents of the stack are:
o
l
l
e
h
string is not palindrome
```

```
1-push
2-pop
3-display
4-palindrome
5-exit
enter the choice
5
```

PROGRAM 4

Develop 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.

```
#include<stdio.h>
#include<conio.h>
#include<string.h>
void push(char);
char pop();
int preced(char);
int top=-1;
char infix[20],postfix[20],s[20];
void infix_to_postfix(char infix[ ],char postfix[ ])
{
 int index=0,pos=0,length;
 char sym,temp;
 length=strlen(infix);
 push('#');
 while(index<length)
        {
          sym=infix[index];
              switch(sym)
                    {
                     case '(' : push(sym);
                               break;
                     case ')' :  temp=pop();
                                                   while( temp!= '(' )
                                  {
                                    postfix[pos]=temp;
                                    pos++;
                                    temp=pop();
                                  }
                              break;
                    case '+':
                    case '-':
                                 case '*':
                    case '/ ':
                    case '%':
                    case '^':
                         while(preced(s[top])>=preced(sym))
                                {
```

```
                            temp=pop();
                            postfix[pos]=temp;
                          pos++;
                          }
                          push(sym);
                    break;
                  default:postfix[pos++]=sym;
                        break;
             }
        index++;
      }

 while(top>0)
          {
            temp=pop();
            postfix[pos++]=temp;
          }
           return;
}
      void push(char symb)
      {
          top++;
          s[top]=symb;
       }
       char pop()
       {
          char symb;
          symb=s[top];
          top--;
          return(symb);
  }
        int preced(char symb)
        {
          int p;
             switch(symb)
                {
                 case '^':p=3;
                       break;
                 case '*':
                 case '/':
                 case '%':p=2;
                       break;
                 case '+':
```

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

void main()
   {
               clrscr();
               printf("enter a valid infix expression\n\n");
               scanf("%s",infix);
               infix_to_postfix(infix,postfix);
               printf("\n infix expresion is: %s\n\n",infix);
               printf("******************************************\n");
               printf("postfix expression is: %s\n",postfix);
                getch();
          }
```

Output1:

```
enter a valid infix expression
(a+b)*c
 infix expresion is:(a+b)*c
******************************************
postfix expression is: ab+c*
```

Output2:

```
enter a valid infix expression
a*b/c^d-e+f%g
 infix expresion is: a*b/c^d-e+f%g
******************************************
postfix expression is: ab/cd^*e-fg%+
```

PROGRAM 5

Develop 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<conio.h>
#include<math.h>
#define MAX 100
typedef struct
{
int a[MAX];
int top;
}STACK;
void push(STACK *s,int x)
{
if(s->top==MAX-1)
printf("STACK OVERFLOW\n");
else
s->a[++s->top]=x;
}
int pop(STACK *s)
{
int x;
if(s->top==-1)
printf("STACK UNDERFLOW\n");
else
x=s->a[s->top--];
return x;
}
int operation(int p1,int p2,char op)
{
switch(op)
{
case '+':return p1+p2;
case '*':return p1*p2;
case '-':return p1-p2;
case '/':return p1/p2;
case '%':return p1%p2;
case '^':return pow(p1,p2);
}
return 0;
```

```
}
int evaluate(char pos[])
{
STACK s1;
int p1,p2,result,i;
s1.top=-1;
for(i=0;pos[i]!='\0';i++)
if(isdigit(pos[i]))
push(&s1,pos[i]-'0');
else
{
p2=pop(&s1);
p1=pop(&s1);
result=operation(p1,p2,pos[i]);
push(&s1,result);
}
return pop(&s1);
}
void towers(int n,char Beg,char Aux,char End)
{
int top=-1,Add,temp;
char STN[20],STBEG[20],STEND[20],STAUX[20];
int STADD[20];
level1 :if(n==1)
{
printf("move the disk from %c to %c\n",Beg,End);
goto level5;
}
level2 :top=top+1;
STN[top]=n;
STBEG[top]=Beg;
STAUX[top]=Aux;
STEND[top]=End;
STADD[top]=3;
n=n-1;
temp=Aux;
Aux=End;
End=temp;
goto level1;
level3 :printf("move the disk from %c to %c\n",Beg,End);
level4 :top=top+1;
STN[top]=n;
STBEG[top]=Beg;
```

```
STAUX[top]=Aux;
STEND[top]=End;
STADD[top]=5;
n=n-1;
temp=Beg;
Beg=Aux;
Aux=temp;
goto level1;
level5 :
if(top==-1)
return;
else
{
n=STN[top];
Beg=STBEG[top];
Aux=STAUX[top];
End=STEND[top];
Add=STADD[top];
top=top-1;
if(Add==3)
goto level3;
else
goto level5;
}
}
void main()
{
int n,ch;
char postfix[100];
clrscr();
while(1)
{
printf("\n1.Evaluate postfix expression\n2.Towers of Hanoi\n3.Exit\n");
printf("enter the chioce\n");
scanf("%d",&ch);
switch(ch)
{
case 1:printf("please enter the VALID POSTFIX STRING\n\noperands are SINGLE
DIGIT\n\n");
fflush(stdin);
gets(postfix);
printf("\nthe result is ==>%d",evaluate(postfix));
break;
case 2:
printf("\n-------TOWERS OF HANOI-------\n");
printf("enter the value of number of disks\n");
scanf("%d",&n);
towers(n,'A','B','C');
break;
case 3:exit(0);
}
getch();
}
}
```

Output1:

```
1.Evaluate postfix expression
2.Towers of Hanoi
3.Exit
enter the chioce
1
please enter the VALID POSTFIX STRING
operands are SINGLE DIGIT
123+*321-+*
the result is ==>20
1.Evaluate postfix expression
2.Towers of Hanoi
3.Exit
enter the chioce
1
please enter the VALID POSTFIX STRING
operands are SINGLE DIGIT
23*-
STACK UNDERFLOW
1.Evaluate postfix expression
2.Towers of Hanoi
3.Exit
enter the chioce
2
-------TOWERS OF HANOI-------
enter the value of number of disks
3
move the disk from A to C
move the disk from A to B
move the disk from C to B
move the disk from A to C
move the disk from B to A
move the disk from B to C
move the disk from A to C
1.Evaluate postfix expression
2.Towers of Hanoi
3.Exit
enter the chioce
3
```

PROGRAM 6

Develop 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<stdio.h>
#define MAX 5

int cqueue[MAX];

int f=-1,r=-1;

int queuefull()

{

if((f==r+1)||(f==0&&r==MAX-1))

return 1;

else

return 0;

}

int queueempty()

{

if(f==-1)

return 1;

else

return 0;

}

void insert(int item)

{

if(f==-1)

f=0;

r=(r+1)%MAX;
```

```
cqueue[r]=item;
}
int delete()
{
int element;
if(queueempty())
{
printf("circular queue is underflow\n");
return -1;
}
else
{
element=cqueue[f];
if(f==r)
{
f=-1;r=-1;
}
else
f=(f+1)%MAX;
return element;
}
}
void display()
{
int i;
if(queueempty())
printf("circular queue is empty\n");
else
{
```

```
for(i=f;i!=r;i=(i+1)%MAX)
{
printf("cqueue[%d];",i);
printf("%d\n",cqueue[i]);
}
printf("cqueue[%d]:",i);
printf("%d\n",cqueue[i]);
}
}
void main()
{
int item,ch;
clrscr();
while(1)
{
printf("\n1.insert\n2.delete\n3.display\n4.exit\n");
printf("enter the choice\n");
scanf("%d",&ch);
switch(ch)
{
case 1:if(queuefull())
{
printf("circular queue is overflow\n");
}
else
{
printf("enter the item to be insert:\n");
scanf("%d",&item);
insert(item);
}
break;
case 2:item=delete();
if(item!=-1)
printf("the deleted element from the circlar queue is %d\n",item);
break;
case 3:display();
break;
case 4:exit(0);
default :printf("invalid choice\n");
break;
}   }   }
```

OUTPUT:

```
1.insert
2.delete
3.display
4.exit
enter the choice
2
circular queue is underflow
1.insert
2.delete
3.display
4.exit
enter the choice
3
circular queue is empty
1.insert
2.delete
3.display
4.exit
enter the choice
1
enter the item to be insert:
10
```

```
1.insert
2.delete
3.display
4.exit
enter the choice
1
enter the item to be insert:
20
1.insert
2.delete
3.display
4.exit
enter the choice
1
enter the item to be insert:
30
1.insert
2.delete
3.display
4.exit
enter the choice
1
enter the item to be insert:
40
1.insert
2.delete
3.display
4.exit
enter the choice
3
cqueue[0];10
cqueue[1];20
cqueue[2];30
cqueue[3]:40
1.insert
2.delete
3.display
4.exit
enter the choice
2
the deleted element from the circlar queue is 10
1.insert
2.delete
```

```
3.display
4.exit
enter the choice
3
cqueue[1];20
cqueue[2];30
cqueue[3]:40

1.insert
2.delete
3.display
4.exit
enter the choice
1
enter the item to be insert:
55
1.insert
2.delete
3.display
4.exit
enter the choice
3
cqueue[1];20
cqueue[2];30
cqueue[3];40
cqueue[4]:55
1.insert
2.delete
3.display
4.exit
enter the choice
1
enter the item to be insert:
66
1.insert
2.delete
3.display
4.exit
enter the choice
1
circular queue is overflow
1.insert
2.delete
```

```
3.display
4.exit
enter the choice
3
cqueue[1];20
cqueue[2];30
cqueue[3];40
cqueue[4];55
cqueue[0]:66
1.insert
2.delete
3.display
4.exit
enter the choice
2
the deleted element from the circlar queue is 20
1.insert
2.delete
3.display
4.exit
enter the choice
2
the deleted element from the circular queue is 30
1.insert
2.delete
3.display
4.exit
```

PROGRAM 7

Develop a menu driven Program in C for the following operations on the Singly Linked List (SLL) of Student Data with the fields: USN, Name, Branch, 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<conio.h>
#include<string.h>
#include<malloc.h>
#include<stdlib.h>
struct student
{
char usn[20];
char name[20];
char branch[20];
int sem;
char phno[20];
struct student *next;
};
typedef struct student *NODE;
NODE first;
void insert_front()
{
NODE nw;
char usn[20],branch[20],name[20],num[20];
int x;
printf("\nenter the student usn,name,branch,sem and phno:");
scanf("%s%s%s%d%s",usn,name,branch,&x,num);
nw=(NODE)malloc(sizeof(struct student));
strcpy(nw->usn,usn);
strcpy(nw->name,name);
strcpy(nw->branch,branch);
nw->sem=x;
strcpy(nw->phno,num);
nw->next=first;
first=nw;
```

```
}
void insert_end()
{
NODE nw,p;
char usn[20],branch[20],name[20],num[20];
int x;
printf("\nenter the student usn,name,branch,sem,phno:");
scanf("%s%s%s%d%s",usn,name,branch,&x,num);
nw=(NODE)malloc(sizeof(struct student));
strcpy(nw->usn,usn);
strcpy(nw->name,name);
strcpy(nw->branch,branch);
nw->sem=x;
strcpy(nw->phno,num);
nw->next=NULL;
if(first==NULL)
first=nw;
else
{
p=first;
while(p->next!=NULL)
p=p->next;
p->next=nw;
}
}
void delet_front()
{
NODE temp;
if(first==NULL)
printf("list is empty\n");
else
{
temp=first;
first=temp->next;
printf("\nthe deleted student usn is: %s",temp->usn);
free(temp);
}
}
void delet_end()
{
NODE temp,prev;
temp=first;
if(temp->next==NULL)
```

```
first=NULL;
else
{
while(temp->next!=NULL)
{
prev=temp;
temp=temp->next;
}
prev->next=NULL;
}
printf("\nthe deleted student usn is: %s",temp->usn);
}
void display()
{
NODE temp;
int count=0;
temp=first;
if(first==NULL)
printf("\nempty list\n");
else
{
printf("\nUSN\t\tNAME\t\tBRANCH\t\tSEM\t\tPHNO");
while(temp!=NULL)
{
printf("\n\n%s\t\t%s\t\t%s\t\t%d\t\t%s",temp->usn,temp->name,temp-
>branch,temp->sem,temp->phno);
temp=temp->next;
count++;
}
printf("\n\nthe number of nodes are %d",count);
}
}
void stackdemo()
{
int ch;
while(1)
{
printf("\n___STACK DEMONSTRATION___\n");
printf("\n1.PUSH\n2.POP\n3.DISPLAY\n4.EXIT");
printf("\nenter the choice\n");
scanf("%d",&ch);
switch(ch)
{
```

```
case 1:insert_front();
    break;
case 2:delet_front();
    break;
case 3:display();
    break;
case 4:return;
default :printf("invalid choice\n");
         break;
         }
         }
         }
void main()
{
int sem,ch;
char name[20],usn[20],branch[20],num[20];
clrscr();
while(1)
{
printf("\n____MAIN MENU____\n");
printf("\n1.CREATE(INSERT AT FRONT)\n2.DISPLAY\n3.DELETE AT
FRONT\n4.INSERT AT END\n5.DELETE AT END\n6.STACK
DEMONSTRATION\n7.EXIT");
printf("\nenter the choice\n");
scanf("%d",&ch);
switch(ch)
{
case 1:insert_front();
    break;
case 2:display();
    break;
case 3:delet_front();
    break;
case 4:insert_end();
    break;
case 5:if(first==NULL)
    printf("list is empty\n");
    else
    delet_end();
    break;
case 6:stackdemo();
    break;
case 7:exit(0);
```

```
default :printf("\n invalid selection");
            break;
            }
            }
            }
```

Output:

```
___MAIN MENU___

1.CREATE(INSERT AT FRONT)
2.DISPLAY
3.DELETE AT FRONT
4.INSERT AT END
5.DELETE AT END
6.STACK DEMONSTRATION
7.EXIT
enter the choice
2
empty list
___MAIN MENU___
1.CREATE(INSERT AT FRONT)
2.DISPLAY
3.DELETE AT FRONT
4.INSERT AT END
5.DELETE AT END
6.STACK DEMONSTRATION
7.EXIT
enter the choice
5
list is empty
___MAIN MENU___
1.CREATE(INSERT AT FRONT)
2.DISPLAY
3.DELETE AT FRONT
4.INSERT AT END
5.DELETE AT END
6.STACK DEMONSTRATION
7.EXIT
enter the choice
2
empty list
___MAIN MENU___
1.CREATE(INSERT AT FRONT)
```

```
2.DISPLAY
3.DELETE AT FRONT
4.INSERT AT END
5.DELETE AT END
6.STACK DEMONSTRATION
7.EXIT
enter the choice
1
enter the student usn,name,branch,sem and phno:
4BD13IS100  ABI  ISE  3  9898989898
___MAIN MENU___
1.CREATE(INSERT AT FRONT)
2.DISPLAY
3.DELETE AT FRONT
4.INSERT AT END
5.DELETE AT END
6.STACK DEMONSTRATION
7.EXIT
enter the choice
2
USN          NAME         BRANCH       SEM          PHNO

4BD13IS100          ABI          ISE          3          9898989898

the number of nodes are 1
___MAIN MENU___
1.CREATE(INSERT AT FRONT)
2.DISPLAY
3.DELETE AT FRONT
4.INSERT AT END
5.DELETE AT END
6.STACK DEMONSTRATION
7.EXIT
enter the choice
1
4BD13IS101          ARUN          ISE          3          9797979797
___MAIN MENU___
1.CREATE(INSERT AT FRONT)
2.DISPLAY
3.DELETE AT FRONT
4.INSERT AT END
5.DELETE AT END
6.STACK DEMONSTRATION
```

```
7.EXIT
enter the choice
2
USN          NAME         BRANCH       SEM          PHNO
4BD13IS101          ARUN         ISE          3           9797979797
4BD13IS100          ABI          ISE          3           9898989898

the number of nodes are 2
___MAIN MENU___
1.CREATE(INSERT AT FRONT)
2.DISPLAY
3.DELETE AT FRONT
4.INSERT AT END
5.DELETE AT END
6.STACK DEMONSTRATION
7.EXIT
enter the choice
4
enter the student usn,name,branch,sem,phno:
4BD13IS102  RAJU  ISE  3  8989898989
___MAIN MENU___
1.CREATE(INSERT AT FRONT)
2.DISPLAY
3.DELETE AT FRONT
4.INSERT AT END
5.DELETE AT END
6.STACK DEMONSTRATION
7.EXIT
enter the choice
2
USN          NAME         BRANCH       SEM          PHNO
4BD13IS101          ARUN         ISE          3           9797979797
4BD13IS100          ABI          ISE          3           9898989898
4BD13IS102          RAJU         ISE          3           8989898989
the number of nodes are 3
___MAIN MENU___
1.CREATE(INSERT AT FRONT)
2.DISPLAY
3.DELETE AT FRONT
4.INSERT AT END
5.DELETE AT END
6.STACK DEMONSTRATION
7.EXIT
```

```
enter the choice
3
the deleted student usn is: 4BD13IS101
___MAIN MENU___
1.CREATE(INSERT AT FRONT)
2.DISPLAY
3.DELETE AT FRONT
4.INSERT AT END
5.DELETE AT END
6.STACK DEMONSTRATION
7.EXIT
enter the choice
2
USN         NAME        BRANCH      SEM         PHNO
4BD13IS100          ABI         ISE         3           9898989898
4BD13IS102          RAJU         ISE         3           8989898989
the number of nodes are 2
___MAIN MENU___
1.CREATE(INSERT AT FRONT)
2.DISPLAY
3.DELETE AT FRONT
4.INSERT AT END
5.DELETE AT END
6.STACK DEMONSTRATION
7.EXIT
enter the choice
5
the deleted student usn is: 4BD13IS102
___MAIN MENU___
1.CREATE(INSERT AT FRONT)
2.DISPLAY
3.DELETE AT FRONT
4.INSERT AT END
5.DELETE AT END
6.STACK DEMONSTRATION
7.EXIT
enter the choice
2
USN         NAME        BRANCH      SEM         PHNO
4BD13IS100          ABI         ISE         3           9898989898
the number of nodes are 1
___MAIN MENU___
1.CREATE(INSERT AT FRONT)
```

```
2.DISPLAY
3.DELETE AT FRONT
4.INSERT AT END
5.DELETE AT END
6.STACK DEMONSTRATION
7.EXIT
enter the choice
6
___STACK DEMONSTRATION___
1.PUSH
2.POP
3.DISPLAY
4.EXIT
enter the choice
1
enter the student usn,name,branch,sem and phno:
4BD13IS090  SURYA  ISE  3  8686868686
___STACK DEMONSTRATION___
1.PUSH
2.POP
3.DISPLAY
4.EXIT
enter the choice
3
USN          NAME         BRANCH       SEM          PHNO
4BD13IS090          SURYA        ISE          3            8686868686
4BD13IS100          ABI          ISE          3            9898989898
the number of nodes are 2
___STACK DEMONSTRATION___
1.PUSH
2.POP
3.DISPLAY
4.EXIT
enter the choice
1
enter the student usn,name,branch,sem and phno:
4BD13IS080  KUMAR  ISE  3  9090909090
___STACK DEMONSTRATION___
1.PUSH
2.POP
3.DISPLAY
4.EXIT
enter the choice
```

```
3
USN          NAME         BRANCH       SEM         PHNO
4BD13IS080          KUMAR        ISE         3          90909090903
4BD13IS090          SURYA        ISE         3          8686868686
4BD13IS100          ABI          ISE         3          9898989898
the number of nodes are 3
____STACK DEMONSTRATION____
1.PUSH
2.POP
3.DISPLAY
4.EXIT
enter the choice
2
the deleted student usn is: 4BD13IS080
____STACK DEMONSTRATION____
1.PUSH
2.POP
3.DISPLAY
4.EXIT
enter the choice
3
USN          NAME         BRANCH       SEM         PHNO
4BD13IS090          SURYA        ISE         3          8686868686
4BD13IS100          ABI          ISE         3          9898989898
the number of nodes are 2
____STACK DEMONSTRATION____
1.PUSH
2.POP
3.DISPLAY
4.EXIT
enter the choice
4
____MAIN MENU____
1.CREATE(INSERT AT FRONT)
2.DISPLAY
3.DELETE AT FRONT
4.INSERT AT END
5.DELETE AT END
6.STACK DEMONSTRATION
7.EXIT
enter the choice
7
```

PROGRAM 8

Develop 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>
struct node
{
char ssn[20],name[20],dept[20],desi[20],phno[20];
float salary;
struct node *next;
struct node *prev;
};
typedef struct node *NODE;
NODE first=NULL;
NODE read()
{
NODE temp;
char ssn[20],name[20],dept[20],desi[20],phno[20];
float sal;
printf("Enter the employee ssn,name,dept,designation,sal and phno:\n");
scanf("%s%s%s%s%f%s",ssn,name,dept,desi,&sal,phno);
temp=(NODE)malloc(sizeof(struct node));
strcpy(temp->ssn,ssn);
strcpy(temp->name,name);
strcpy(temp->dept,dept);
strcpy(temp->desi,desi);
temp->salary=sal;
strcpy(temp->phno,phno);
return temp;
}
void addfront()
{
NODE nw;
nw=read();
if(first==NULL)
{
```

```
first=nw;
first->prev=first->next=NULL;
}
else
{
nw->next=first;
first->prev=nw;
nw->prev=NULL;
first=nw;
}
}
void addend()
{
NODE nw,p;
nw=read();
if(first==NULL)
{
first=nw;
first->prev=first->next=NULL;
}
p=first;
while(p->next!=NULL)
p=p->next;
p->next=nw;
nw->prev=p;
nw->next=NULL;
}
void delfront()
{
NODE temp;
temp=first;
if(temp->next==NULL&&temp->prev==NULL)
first=NULL;
else
{
first=temp->next;
first->prev=NULL;
}
printf("Employee information deleted with ssn %s:",temp->ssn);
free(temp);
}
void delend()
{
```

```
NODE temp,p;
temp=first;
if(temp->next==NULL&&temp->prev==NULL)
first=NULL;
else
{
while(temp->next!=NULL)
temp=temp->next;
p=temp->prev;
p->next=NULL;
}
printf("Employee information deleted with ssn %s:",temp->ssn);
free(temp);
}
void display()
{
NODE temp;
temp=first;
printf("\nSSN\tNAME\tDEPT\tDESI\tSALARY\tPHNO\n");
while(temp!=NULL)
{
printf("%s\t%s\t%s\t%s\t%5.2f\t%s\t\n",temp->ssn,temp->name,temp->dept,temp->desi,temp->salary,temp->phno);
temp=temp->next;
}
}
void main()
{
int choice;
clrscr();
printf("\nDoubly Linked List is demonstrated as Double Ended Queue\n");
while(1)
{
printf("\n 1:CREATE(ADDEND)\n 2:DISPLAY\n 3:DELETION AT END\n 4:INSETION AT FRONT\n 5:DELETION AT FRONT\n 6:EXIT\n");
printf("enter the choice\n");
scanf("%d",&choice);
switch(choice)
{
case 1:addend();
     break;
case 2:if(first==NULL)
          printf("List is Empty\n");
```

```
            else
            display();
            break;
case 3:if(first==NULL)
            printf("List is Empty\n");
            else
            delend();
            break;
case 4:addfront();
      break;
case 5:if(first==NULL)
            printf("List is Empty\n");
            else
            delfront();
            break;
case 6:exit(0);
default:printf("invalid choice\n");
break;
}
}
}
```

Output

```
Doubly Linked List is demonstrated as Double Ended Queue
 1:CREATE(ADDEND)
 2:DISPLAY
 3:DELETION AT END
 4:INSETION AT FRONT
 5:DELETION AT FRONT
 6:EXIT
enter the choice
2
List is Empty
 1:CREATE(ADDEND)
 2:DISPLAY
 3:DELETION AT END
 4:INSETION AT FRONT
 5:DELETION AT FRONT
 6:EXIT
enter the choice
3
List is Empty
 1:CREATE(ADDEND)
 2:DISPLAY
 3:DELETION AT END
 4:INSETION AT FRONT
 5:DELETION AT FRONT
 6:EXIT
enter the choice
5
List is Empty
 1:CREATE(ADDEND)
 2:DISPLAY
 3:DELETION AT END
 4:INSETION AT FRONT
 5:DELETION AT FRONT
 6:EXIT
enter the choice
1
Enter the employee ssn,name,dept,designation,sal and phno:
111 RAHUL ACCOUNT MANAGER 29000 9898098900
 1:CREATE(ADDEND)
 2:DISPLAY
 3:DELETION AT END
 4:INSETION AT FRONT
 5:DELETION AT FRONT
 6:EXIT
enter the choice
2
SSN     NAME    DEPT    DESI    SALARY PHNO
111     RAHUL   ACCOUNT MANAGER 29000.00       9898098900

 1:CREATE(ADDEND)
 2:DISPLAY
 3:DELETION AT END
 4:INSETION AT FRONT
 5:DELETION AT FRONT
 6:EXIT
enter the choice
1
Enter the employee ssn, name, dept, designation, sal and phno:
112 AMAR SALES EXECUTIVE 15000 9090909090

 1:CREATE(ADDEND)
 2:DISPLAY
 3:DELETION AT END
 4:INSETION AT FRONT
 5:DELETION AT FRONT
 6:EXIT
enter the choice
4
Enter the employee ssn, name, dept, designation, sal and phno:
200 RAKESH PRODUCTION SUPERVISIOR 20000 8908908900

 1:CREATE(ADDEND)
 2:DISPLAY
 3:DELETION AT END
 4:INSETION AT FRONT
 5:DELETION AT FRONT
 6:EXIT
enter the choice
2

SSN    NAME    DEPT    DESI    SALARY PHNO
200    RAKESH PRODUCTION     SUPERVISIOR     20000.00      8908908900
111    RAHUL   ACCOUNT MANAGER 29000.00      9898098900
112    AMAR    SALES   EXECUTIVE      15000.00      9090909090

 1:CREATE(ADDEND)
 2:DISPLAY
 3:DELETION AT END
 4:INSETION AT FRONT
 5:DELETION AT FRONT
 6:EXIT
enter the choice
5
Employee information deleted with ssn 200:
 1:CREATE(ADDEND)
 2:DISPLAY
 3:DELETION AT END
 4:INSETION AT FRONT
 5:DELETION AT FRONT
 6:EXIT
enter the choice
2
SSN    NAME    DEPT    DESI    SALARY  PHNO
111    RAHUL   ACCOUNT MANAGER 29000.00      9898098900
112    AMAR    SALES   EXECUTIVE      15000.00      9090909090
```

```
 1:CREATE(ADDEND)
 2:DISPLAY
 3:DELETION AT END
 4:INSETION AT FRONT
 5:DELETION AT FRONT
 6:EXIT
enter the choice
3
Employee information deleted with ssn 112:
 1:CREATE(ADDEND)
 2:DISPLAY
 3:DELETION AT END
 4:INSETION AT FRONT
 5:DELETION AT FRONT
 6:EXIT
enter the choice
2
SSN     NAME    DEPT    DESI    SALARY PHNO
111     RAHUL   ACCOUNT MANAGER 29000.00        9898098900
 1:CREATE(ADDEND)
 2:DISPLAY
 3:DELETION AT END
 4:INSETION AT FRONT
 5:DELETION AT FRONT
 6:EXIT
enter the choice
6
```

PROGRAM 9

Develop 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<conio.h>
#include<math.h>
struct node
{
int coef,xexp,yexp,zexp;
struct node *link;
};
typedef struct node *NODE;

void display(NODE head)
{
NODE temp;
temp=head->link;
while(temp!= head)
{
   printf("%dx^%d y^%d z^%d  ",temp->coef,temp->xexp,temp->yexp,temp->zexp);
   temp=temp->link;
}
printf("\n");
}

NODE attach(int coef,int xexpon,int yexpon,int zexpon,NODE head)
{
NODE temp,cur;
temp=(NODE)malloc(sizeof(struct node));
temp->coef=coef;
temp->xexp=xexpon;
temp->yexp=yexpon;
temp->zexp=zexpon;
cur=head->link;
while(cur->link!=head)
 cur=cur->link;
cur->link=temp;
```

```
temp->link=head;
return head;
}
NODE readpoly(NODE head)
{
int i=1,coef,xexpon,yexpon,zexpon;
printf("enter the coefficient as -99 to end the polynomial\n");
while(1)
{
printf("enter the %d term of the polynomial\n",i++);
printf("enter the coefficient:\n");
scanf("%d",&coef);
if(coef==-99)
break;
printf("enter the xexponent:\n");
scanf("%d",&xexpon);
printf("enter the yexponent:\n");
scanf("%d",&yexpon);
printf("enter the zexponent:\n");
scanf("%d",&zexpon);
head=attach(coef,xexpon,yexpon,zexpon,head);
}
display(head);
return head;
}
int evaluate(NODE head)
{
NODE temp;
int x,y,z,value=0;
temp=head->link;
printf("Enter the value of x,y and z\n");
scanf("%d%d%d",&x,&y,&z);
while(temp!=head)
{
value+=temp->coef*(pow(x,temp->xexp)*pow(y,temp->yexp)*pow(z,temp->zexp));
temp=temp->link;
}
return value;
}
NODE addpoly(NODE head1,NODE head2,NODE head3)
{
 NODE a,b;
 int sum;
```

```
a=head1->link;
b=head2->link;
while(a!=head1 && b!=head2)
{
if(a->xexp==b->xexp)
{
        if(a->yexp==b->yexp)
        {
                if(a->zexp==b->zexp)
                {
                sum=a->coef+b->coef;
                attach(sum,a->xexp,a->yexp,a->zexp,head3);
                a=a->link;
                b=b->link;
                }
                else if(a->zexp>b->zexp)
                {
                attach(a->coef,a->xexp,a->yexp,a->zexp,head3);
                a=a->link;
                }
                else
                {
                attach(b->coef,b->xexp,b->yexp,b->zexp,head3);
                b=b->link;
                }
        }
        else if(a->yexp>b->yexp)
        {
        attach(a->coef,a->xexp,a->yexp,a->zexp,head3);
        a=a->link;
        }
        else
        {
        attach(b->coef,b->xexp,b->yexp,b->zexp,head3);
        b=b->link;
        }
}
else if(a->xexp>b->xexp)
{
attach(a->coef,a->xexp,a->yexp,a->zexp,head3);
a=a->link;
}
else
```

```
 {
 attach(b->coef,b->xexp,b->yexp,b->zexp,head3);
 b=b->link;
 }
}
while(a!=head1)
{
head3=attach(a->coef,a->xexp,a->yexp,a->zexp,head3);
a=a->link;
}
while(b!=head2)
{
head3=attach(b->coef,b->xexp,b->yexp,b->zexp,head3);
b=b->link;
}
return head3;
}

void main()
{
int ch;
NODE head,head1,head2,head3;
head=(NODE)malloc(sizeof(struct node));
head1=(NODE)malloc(sizeof(struct node));
head2=(NODE)malloc(sizeof(struct node));
head3=(NODE)malloc(sizeof(struct node));
clrscr();
while(1)
{
printf("\n1.Evaluate Polynomial\n2.Add two polynomials\n3.Exit\n");
printf("\nEnter the choice\n");
scanf("%d",&ch);
switch(ch)
{
case 1: head->link=head;
        printf("Enter the polynomial\n");
        readpoly(head);
        printf("The result of polynomial is:%d",evaluate(head));
        break;
case 2: head1->link=head1;
        head2->link=head2;
        head3->link=head3;
        printf("Enter the first polynomial\n");
```

```
          readpoly(head1);
          printf("\nEnter the second polynomial\n");
          readpoly(head2);
          printf("\nthe first polynomial is:");
          display(head1);
          printf("\nthe second polynomial is:");
          display(head2);
          addpoly(head1,head2,head3);
          printf("\nthe result of adding two polynomials is:");
          display(head3);
          break;
case 3:exit(0);
}
getch();
}
}
```

Output:

```
1.Evaluate Polynomial

2.Add two polynomials

3.Exit

Enter the choice

1

Enter the polynomial

enter the coefficient as -99 to end the polynomial

enter the 1 term of the polynomial

enter the coefficient:

3

enter the xexponent:

3

enter the yexponent:

1

enter the zexponent:

1
```

```
enter the 2 term of the polynomial

enter the coefficient:

6

enter the xexponent:

2

enter the yexponent:

2

enter the zexponent:

1

enter the 3 term of the polynomial

enter the coefficient:

2

enter the xexponent:

1

enter the yexponent:

5

enter the zexponent:

1

enter the 4 term of the polynomial

enter the coefficient:

-2

enter the xexponent:

1

enter the yexponent:

1

enter the zexponent:

3
```

```
enter the 5 term of the polynomial

enter the coefficient:

-4

enter the xexponent:

0

enter the yexponent:

1

enter the zexponent:

5

enter the 6 term of the polynomial

enter the coefficient:

-99

3x^3 y^1 z^1  6x^2 y^2 z^1  2x^1 y^5 z^1  -2x^1 y^1 z^3  -4x^0 y^1 z^5

Enter the value of x,y and z

1

1

1

The result of polynomial is:5

1.Evaluate Polynomial

2.Add two polynomials

3.Exit

Enter the choice

2

Enter the first polynomial

enter the coefficient as -99 to end the polynomial

enter the 1 term of the polynomial

enter the coefficient:
```

```
3
enter the xexponent:
3
enter the yexponent:
3
enter the zexponent:
1
enter the 2 term of the polynomial
enter the coefficient:
6
enter the xexponent:
2
enter the yexponent:
1
enter the zexponent:
1
enter the 3 term of the polynomial
enter the coefficient:
4
enter the xexponent:
1
enter the yexponent:
1
enter the zexponent:
1
enter the 4 term of the polynomial
enter the coefficient:
```

```
-99
3x^3 y^3 z^1  6x^2 y^1 z^1  4x^1 y^1 z^1


Enter the second polynomial
enter the coefficient as -99 to end the polynomial
enter the 1 term of the polynomial
enter the coefficient:
4
enter the xexponent:
5
enter the yexponent:
2
enter the zexponent:
4
enter the 2 term of the polynomial
enter the coefficient:
5
enter the xexponent:
3
enter the yexponent:
3
enter the zexponent:
1
enter the 3 term of the polynomial
enter the coefficient:
10
enter the xexponent:
```

```
1

enter the yexponent:

1

enter the zexponent:

1

enter the 4 term of the polynomial

enter the coefficient:

-99

4x^5 y^2 z^4  5x^3 y^3 z^1  10x^1 y^1 z^1

the first polynomial is:3x^3 y^3 z^1  6x^2 y^1 z^1  4x^1 y^1 z^1

the second polynomial is:4x^5 y^2 z^4  5x^3 y^3 z^1  10x^1 y^1 z^1

the result of adding two polynomials is:4x^5 y^2 z^4  8x^3 y^3 z^1  6x^2 y^1 z^1
14x^1 y^1 z^1

1.Evaluate Polynomial

2.Add two polynomials

3.Exit

Enter the choice 3
```

PROGRAM 10

Develop 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.**

```
#include<stdio.h>
struct node
{
int info;
struct node *lt,*rt;
};
typedef struct node *NODE;
NODE root=NULL;
void inorder(NODE temp)
{
if(temp!=NULL)
{
inorder(temp->lt);
printf("%d\t",temp->info);
inorder(temp->rt);
}
}
void preorder(NODE temp)
{
if(temp!=NULL)
{
printf("%d\t",temp->info);
preorder(temp->lt);
preorder(temp->rt);
}
}
void postorder(NODE temp)
{
if(temp!=NULL)
{
postorder(temp->lt);
postorder(temp->rt);
printf("%d\t",temp->info);
}
```

```
}
void createbst()
{
NODE temp,cur,prev;
int ele;
temp=(NODE)malloc(sizeof(struct node));
temp->lt=temp->rt=NULL;
printf("\nEnter the element\n");
scanf("%d",&ele);
temp->info=ele;
if(root==NULL)
{
root=temp;
return;
}
prev=NULL;
cur=root;
while(cur!=NULL)
{
prev=cur;
if(ele==cur->info)
{
printf("duplicate");
free(temp);
return;
}
if(ele<cur->info)
cur=cur->lt;
else
cur=cur->rt;
}
if(ele<prev->info)
prev->lt=temp;
else
prev->rt=temp;
}
int search()
{
int key;
NODE temp;
printf("Enter the key element to be search\n");
scanf("%d",&key);
temp=root;
```

```
while(temp!=NULL)
{
if(temp->info==key)
return 1;
else if(key>temp->info)
temp=temp->rt;
else
temp=temp->lt;
}
return 0;
}

void main()
{
int ch;
clrscr();
while(1)
{
printf("\n\n1.create BST\n2.Traversals\n3.search\n4.exit\n");
printf("Enter the choice\n");
scanf("%d",&ch);
switch(ch)
{
case 1:createbst();
        break;
case 2:if(root==NULL)
        printf("\nBST is Empty\n");
        else
        {
        printf("\n\nINORDER:");
        inorder(root);
        printf("\n\nPREORDER:");
        preorder(root);
        printf("\n\nPOSTORDER:");
        postorder(root);
        }
        break;
case 3:if(root==NULL)
        printf("\nBST is Empty\n");
        else
        {
        if(search()==0)
        printf("The key element is not present\n");
        else
        printf("The key element is present in BST\n");
        }
        break;
case 4:exit(0);
default:printf("\ninvalid choice\n");
        break;
}
}
}
```

Output:

```
1.create BST
2.Traversals
3.search
4.exit
Enter the choice
2
BST is Empty
1.create BST
2.Traversals
3.search
4.exit
Enter the choice
3
BST is Empty
1.create BST
2.Traversals
3.search
4.exit
Enter the choice
1
Enter the element
6
1.create BST
2.Traversals
3.search
4.exit
Enter the choice
1
Enter the element
9
1.create BST
```

```
2.Traversals
3.search
4.exit
Enter the choice
1
Enter the element
5
1.create BST
2.Traversals
3.search
4.exit
Enter the choice
1
1.create BST
2.Traversals
3.search
4.exit
Enter the choice
1
Enter the element
2
1.create BST
2.Traversals
3.search
4.exit
Enter the choice
1
Enter the element
8
1.create BST
2.Traversals
3.search
4.exit
Enter the choice
1
Enter the element
15
1.create BST
2.Traversals
3.search
4.exit
Enter the choice
1
```

```
Enter the element
24
1.create BST
2.Traversals
3.search
4.exit
Enter the choice
1
Enter the element
7
1.create BST
2.Traversals
3.search
4.exit
Enter the choice
1
Enter the element
8
duplicate
1.create BST
2.Traversals
3.search
4.exit
1
Enter the element
5
duplicate

1.create BST
2.Traversals
3.search
4.exit
Enter the choice
1
Enter the element
2
duplicate
1.create BST
2.Traversals
3.search
4.exit
Enter the choice
2
```

```
INORDER: 2      5      6      7      8      9      15     24
PREORDER: 6     5      2      9      8      7      15     24
POSTORDER: 2    5      7      8      24     15     9      6
1.create BST
2.Traversals
3.search
4.exit
Enter the choice
3
Enter the key element to be search
15
The key element is present in BST
1.create BST
2.Traversals
3.search
4.exit
Enter the choice
3
Enter the key element to be search
12
The key element is not present
1.create BST
2.Traversals
3.search
4.exit
Enter the choice
4
```

PROGRAM 11

Develop 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>
int a[10][10]={0};

void creategraph(int n)

{

int ans;

int i,j,k;

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

{

printf("Enter the vertex adjacent to the vertex %d\n",i);

for(j=0;j<n-1;j++)

{

scanf("%d",&k);

if(k==-1)break;

else if(k<n)

a[i][k]=1;

}

}

}

void dfs(int a[][10],int n,int start)

{

int stack[10],visited[10]={0};

int top=-1,i;

printf("%d-",start);

visited[start]=1;

stack[++top]=start;
```

```
while(top!=-1)
{
start=stack[top];
for(i=0;i<n;i++)
{
if(a[start][i]&&visited[i]==0)
{
stack[++top]=i;
printf("%d-",i);
visited[i]=1;
break;
}
}
if(i==n)
top--;
}
}

void bfs(int a[][10],int n,int start)
{
int visited[10]={0},queue[10],rear=-1,front=-1,i;
queue[++rear]=start;
visited[start]=1;
while(rear!=front)
{
start=queue[++front];
printf("%d-",start);
for(i=0;i<n;i++)
{
```

```
if(a[start][i]==1 && visited[i]==0)
{
queue[++rear]=i;
visited[i]=1;
}
}
}
}
void main()
{
int n,i,j,src,ch;
clrscr();
printf("Enter the number of cities\n");
scanf("%d",&n);
creategraph(n);
printf("\nThe adjacency matrix is\n");
for(i=0;i<n;i++)
{
for(j=0;j<n;j++)
{
printf("%d\t",a[i][j]);
}
printf("\n");
}
do
{
printf("\nEnter the source vertex\n");
scanf("%d",&src);
printf("\n------DFS--------\n");
```

```
dfs(a,n,src);

printf("\n-------BFS-------\n");

bfs(a,n,src);

printf("\nWant to continue for another source vertex press0/1:");

scanf("%d",&ch);

}while(ch!=0);

}
```

Output:

```
Enter the number of cities
4
Enter the vertex adjacent to the vertex 0
1 3 -1
Enter the vertex adjacent to the vertex 1
0 2 -1
Enter the vertex adjacent to the vertex 2
1 3 -1
Enter the vertex adjacent to the vertex 3
0 2 -1

The adjacency matrix is
0	1	0	1
1	0	1	0
0	1	0	1
1	0	1	0

Enter the source vertex
3
------DFS--------
3-0-1-2-
-------BFS-------
3-0-2-1-
Want to continue for another source vertex press0/1:1

Enter the source vertex
1
------DFS--------
1-0-3-2-
```

```
-------BFS-------
1-0-2-3-
Want to continue for another source vertex press0/1:
0
Enter the number of cities
4
Enter the vertex adjacent to the vertex 0
1 3 -1
Enter the vertex adjacent to the vertex 1
2 -1
Enter the vertex adjacent to the vertex 2
0 -1
Enter the vertex adjacent to the vertex 3
2 -1

The adjacency matrix is
0       1       0       1
0       0       1       0
1       0       0       0
0       0       1       0

Enter the source vertex
0

------DFS--------
0-1-2-3-
-------BFS-------
0-1-3-2-
Want to continue for another source vertex press0/1:1

Enter the source vertex
2
------DFS--------
2-0-1-3-
-------BFS-------
2-0-1-3-
Want to continue for another source vertex press0/1:
0
```

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. 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>
#include<conio.h>
#include<stdlib.h>
#define MAX 100
int create(int);
void linear_prob(int[],int,int);
void display(int[]);
void main()
{
 int a[MAX],num,key,i;
 int ans=1,count=0;
 clrscr();
 printf("Collision handling by linear probing:\n");
 for(i=0;i<MAX;i++)
 {
  a[i]=-1;
 }
 do
 {
  printf("\nEnter the data");
  scanf("%4d",&num);
  count++;
  key=create(num);
  linear_prob(a,key,num);
  if(count==MAX)
  {
  printf("Hash Table is Full\n");
  break;
  }
  printf("\nDo you wish to continue?(1/0)");
  scanf("%d",&ans);
 }
```

```
  while(ans);
  display(a);
  getch();
 }
int create(int num)
{
 int key;
 key=num%100;
 return key;
}
void linear_prob(int a[MAX],int key,int num)
{
 int flag,i;
 flag=0;
 if(a[key]==-1)
 {
  a[key]=num;
 }
 else
 {
  printf("\nCollision Detected...!!!\n");
  printf("Collision avoided successfully using LINEAR PROBING\n");
  for(i=key+1;i<MAX;i++)
  if(a[i]==-1)
  {
   a[i]=num;
   flag=1;
   break;
  }
  i=0;
  while((i<key)&&(flag==0))
  {
   if(a[i]==-1)
   {
    a[i]=num;
    flag=1;
    break;
   }
   i++;
  }
 }
}
void display(int a[MAX])
```

```
{
 int i,choice,ch;
 do
 {
 printf("\n1.Display ALL \n2.Filtered Display\n");
 scanf("%d",&choice);
 if(choice==1)
 {
  printf("\nthe hash table is\n");
  for(i=0;i<MAX;i++)
  printf("\n [%d] %d",i,a[i]);
 }
 else
 {
  printf("\nthe hash table is\n");
  for(i=0;i<MAX;i++)
  if(a[i]!=-1)
    printf("\n [%d] %d",i,a[i]);
  }
   printf("\nwant to continue\n");
   scanf("%d",&ch);
  }while(ch!=0);
 }
```

Output:

```
Collision handling by linear probing:
Enter the data
1234
Do you wish to continue?(1/0)1
Enter the data2526
Do you wish to continue?(1/0)1
Enter the data5698
Do you wish to continue?(1/0)1
Enter the data1202
Do you wish to continue?(1/0)1
Enter the data3298

Collision Detected...!!!
Collision avoided successfully using LINEAR PROBING
Do you wish to continue?(1/0)1
Enter the data3526
Collision Detected...!!!
Collision avoided successfully using LINEAR PROBING
```

```
Do you wish to continue?(1/0)1
Enter the data9998
Collision Detected...!!!
Collision avoided successfully using LINEAR PROBING
Do you wish to continue?(1/0)0
1.Display ALL
2.Filtered Display
2
the hash table is
 [0]    9998
 [2]    1202
 [26]   2526
 [27]   3526
 [34]   1234
 [98]   5698
 [99]  3298
want to continue
1
1.Display ALL
2.Filtered Display
0
```