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>              // Structure to represent a day in the calendar
struct Day {
char * dayName;                  // Dynamically allocated string for the day name
int date;
char * activity;                 // Dynamically allocated string for the activity description
};

// Function to create a calendar
void create(struct Day * day)
{                                // Allocate memory for the day name and activity
day -> dayName = (char * ) malloc(sizeof(char) * 20);  // Assuming day names are less than 20 letters
day -> activity = (char * ) malloc(sizeof(char) * 100); // Assuming activity descriptions are less than 100
printf("Enter the day name:");                          // Input the day details
scanf("%s", day -> dayName);
printf("Enter the date:");
scanf("%d", & day -> date);
printf("Enter the activity for the day:");
scanf(" %[^\n]s", day -> activity);                     // Read the entire line, including spaces
}

// Function to read data from the keyboard and create the calendar
void read(struct Day * calendar, int size)
{
for (int i = 0; i < size; i++)
{
printf("Enter details for Day %d:\n", i + 1);
create( & calendar[i]);
}
}

// Function to display the calendar
void display(struct Day * calendar, int size)
```

```
{
printf("\nWeek's Activity Details:\n");
for (int i = 0; i < size; i++) {
printf("Day %d:\n", i + 1);
printf("Day Name: %s\n", calendar[i].dayName);
printf("Date: %d\n", calendar[i].date);
printf("Activity: %s\n", calendar[i].activity);
printf("\n");
}
}
int main()
{
int size;
printf("Enter the number of days in the week:");
scanf("%d", & size);                    // Dynamically allocate memory for the calendar
struct Day * calendar = (struct Day * ) malloc(sizeof(struct Day) * size);
read(calendar, size);                   // Read and display the calendar
display(calendar, size);
return 0;
}
```

OUTPUT:

cc 1.c OR gcc 1.c

./a.out

```
Enter the number of days in the week: 7
Enter details for Day 1:
Enter the day name: Sunday
Enter the date: 1
Enter the activity for the day: Learning
Enter details for Day 2:
Enter the day name: Monday
Enter the date: 2
Enter the activity for the day: Coding
Enter details for Day 3:
Enter the day name: Tuesday
Enter the date: 3
Enter the activity for the day: Testing
Enter details for Day 4:
Enter the day name: Wednesday
Enter the date: 4
Enter the activity for the day: Debugging
```

Enter details for Day 5:
Enter the day name: Thursday
Enter the date: 5
Enter the activity for the day: Publishing
Enter details for Day 6:
Enter the day name: Friday
Enter the date: 6
Enter the activity for the day: Marketing
Enter details for Day 7:
Enter the day name: Saturday
Enter the date: 7
Enter the activity for the day: Earning

Week's Activity Details:
Day 1:
Day Name: Sunday
Date: 1
Activity: Learning
Day 2:
Day Name: Monday
Date: 2
Activity: Coding
Day 3:
Day Name: Tuesday
Date: 3
Activity: Testing
Day 4:
Day Name: Wednesday
Date: 4
Activity: Debugging
Day 5:
Day Name: Thursday
Date: 5
Activity: Publishing
Day 6:
Day Name: Friday
Date: 6
Activity: Marketing
Day 7:
Day Name: Saturday
Date: 7
Activity: Earning

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.

```
#include<stdio.h>
#include<ctype.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");
scanf("%s", &STR);                    //Read the main string
printf("\nEnter a PATTERN string: \n");
scanf("%s", &PAT);                    //Read the pattern string
printf("\nEnter a REPLACE string: \n");
scanf("%s", &REP);                     //Read the replace string
i = m = c = j = 0;
while ( STR[c] != '\0')          //Check till the end of the main string
{
if ( STR[m] == PAT[i] )        // Checking for Match, Cmp main with pattern
{
i++;
m++;
if ( PAT[i] == '\0')
{
flag=1;                              //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];                 //copy non-matched char to ans
j++;
c++;
m = c;
i=0;
}
```

```
}
if(flag==0)
{
printf("Pattern doesn't found!!!\n");
}
else
{
ans[j] = '\0';
printf("\n The RESULTANT string is:%s\n\n",ans);
}
}
```

OUTPUT:

cc 1.c OR gcc 1.c
./a.out

```
Enter the MAIN string:
CSE-ISE
Enter a PATTERN string:
C
Enter a REPLACE string:
M
The RESULTANT string is:MSE-ISE

Enter the MAIN string:
CSE
Enter a PATTERN string:
ISE
Enter a REPLACE string:
MSE
Pattern doesn't found!!!
```

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 Overflow and Underflow situations on Stack
   d. Demonstrate how stack can be used to check Palindrome.
   e. Display the status of Stack
   f. Exit
   Support the program with appropriate functions for each of the above operations

```
#include<stdio.h>
#include<string.h>
#define max_size 5
int stack[max_size],top=-1,flag=1;
int i,temp,item,rev[max_size],num[max_size];
void push();
void pop();
void display();
void pali();
void main()
{
int choice;
printf("\n\n-------- STACK OPERATIONS---------- \n");
printf("1.Push\n");
printf("2.Pop\n");
printf("3.Palindrome\n");
printf("4.Display\n");
printf("5.Exit\n");
printf("---------------------- ");
while(1)
{
printf("\nEnter your choice:\t");
scanf("%d",&choice);
switch(choice)
{
case 1: push();
break;
case 2: pop();
if(flag)
printf("\nThe poped element: %d\t",item);
temp=top;
break;
case 3: palindrome();
```

```
top=temp;
break;
case 4: display();
break;
case 5: exit(0);
break;
default: printf("\nInvalid choice:\n");
break;
}
}
}
void push()
{
if(top==(max_size-1))
{
printf("\nStack Overflow:");
}
else
{
printf("Enter the element to be inserted:\t");
scanf("%d",&item);
top=top+1;
stack[top]=item;
}
temp=top;
}
void pop()
{
if(top==-1)
{
printf("Stack Underflow:");
flag=0;
}
else
{
item=stack[top];
top=top-1;
}
}
void palindrome()
{
i=0;
```

```
if(top ==-1)
{
printf("Push some elements into the stack\n");
}
else
{
while(top!=-1)
{
rev[top]=stack[top];
pop();
}
top=temp;
for(i=0;i<=temp;i++)
{
if(stack[top--]==rev[i])
{
if(i==temp)
{
printf("Palindrome\n");
return;
}
}
}
printf("Not Palindrome\n");
}
}
void display()
{
int i;
top=temp;
if(top==-1)
{
printf("\nStack is Empty:");
}
else
{
printf("\nThe stack elements are:\n" );
for(i=top;i>=0;i--)
{
printf("%d\n",stack[i]);
}
}
}
```

OUTPUT:

cc 1.c OR gcc 1.c
./a.out

```
- STACK OPERATIONS----------
1.Push
2.Pop
3. Palindrome
4. Display
5.Exit
----------------------
Enter your choice: 2
Stack Underflow:
Enter your choice: 3
Stack is Empty:
Enter your choice: 1
Enter the element to be inserted: 25
Enter your choice: 1
Enter the element to be inserted: 45
Enter your choice: 4
The stack elements are:
45
25
Enter your choice: 2
Enter your choice: 4
The stack elements are:
25
Enter your choice: 2
Enter your choice: 2
Stack Underflow
Enter your choice: 1
Enter the element to be inserted: 1
Enter your choice: 1
Enter the element to be inserted: 1
Enter your choice: 4
The stack elements are:
1
1
Enter your choice: 3
Palindrome
Enter your choice: 2
Enter your choice: 1
Enter the element to be inserted: 2
Enter your choice: 1
Enter the element to be inserted: 1
Enter your choice: 2
Enter your choice: 4
```

The stack elements are:
2
1
Enter your choice: 3
Not Palindrome

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.**

```
#include<stdio.h>
#include<string.h>
#include<ctype.h>
int F(char symbol)
{
switch(symbol)
{
case '+':
case '-': return 2;
case '*':
case '/': return 4;
case '^':
case '$': return 5;
case '(': return 0;
case '#': return -1;
default: return 8;
}
}
int G(char symbol)
{
switch(symbol)
{
case '+':
case '-': return 1;
case '*':
case '/': return 3;
case '^':
case '$': return 6;
case '(': return 9;
case ')': return 0;
default: return 7;
}
}
void infix_postfix(char infix[], char postfix[])
{
int top, j, i;
char s[30], symbol;
top = -1;
s[++top] = '#';
j = 0;
for(i=0; i < strlen(infix); i++)
{
symbol = infix[i];
```

```
while(F(s[top]) > G(symbol))
{
postfix[j] = s[top--];
j++;
}
if(F(s[top]) != G(symbol))
s[++top] = symbol;
else
top--;
}
while(s[top] != '#')
{
postfix[j++] = s[top--];
}
postfix[j] = '\0';
}
void main()
{
char infix[20], postfix[20];
printf("\nEnter a valid infix expression\n");
gets(infix);
infix_postfix(infix,postfix);
printf("\nThe infix expression is:\n");
printf ("%s",infix);
printf("\nThe postfix expression is:\n");
printf ("%s",postfix);
}
```

OUTPUT:

cc 1.c –lm OR gcc 1.c –lm
./a.out

```
Enter a valid infix expression
a+b
The infix expression is:
a+b
The postfix expression is:
ab+

Enter a valid infix expression
a+b*c-d
The infix expression is:
a+b*c-d
The postfix expression is:
abc*+d-
```

Enter a valid infix expression
(a+b*c-d)+e/f*h
The infix expression is:
(a+b*c-d)+e/f*h
The postfix expression is:
abc*+d-ef/h*+

Enter a valid infix expression
a+b/d*c^e$g-h+m*c
The infix expression is:
a+b/d*c^e$g-h+m*c
The postfix expression is:
abd/ceg$^*+h-mc*+

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

a. Evaluation of Suffix expression with single digit operands and operators: +, -, *, /, %, ^

```
#include<stdio.h>
#include<math.h>
#include<string.h>
#includde<ctype.h>
float compute(char symbol, float op1,float op2)
{
switch(symbol)
{
case '+': return op1+op2;
case '-': return op1-op2;
case '*': return op1*op2;
case '/': return op1/op2;
case '$':
case '^': return pow(op1,op2);
default: return 0;
}
}
void main()
{
float s[20],res,op1,op2;
int top,i;
char postfix[20],symbol;
printf("\n enter the postfix expression:\n");
scanf("%s", postfix);
top=-1;
for(i=0;i<strlen(postfix);i++)
{
symbol=postfix[i];
if(isdigit(symbol))
{
s[++top]=symbol-'0';
}
else
{
op2=s[top--];
op1=s[top--];
res=compute(symbol,op1,op2);
s[++top]=res;
```

```
}
}
res=s[top--];
printf("\n The result is:%f\n",res);
}
```

OUTPUT:

cc 1.c OR gcc 1.c
./a.out

```
Enter the postfix expression:
67+
The result is:13.000000

Enter the postfix expression:
52-
The result is:3.000000

Enter the postfix expression:
78*
The result is:56.000000

Enter the postfix expression:
98*
The result is:72.000000

Enter the postfix expression:
755/+4-
The result is:4.000000

Enter the postfix expression:
123*+55/-6-
The result is:0.000000

Enter the postfix expression:
925*+55/-
The result is:18.000000
```

b. Solving Tower of Hanoi problem with n disks

```
#include<stdio.h>
int count=0,n;
int tower(int n,char s, char t, char d)
{
if(n==1)
{
printf("\n move disk 1 from %c to %c\n",s,d);
count++;
return 0;
}
tower(n-1,s,d,t);
printf("\n move %d from %c to %c\n",n,s,d);
count++;
tower(n-1,t,s,d);
return 0;
}
void main()
{
printf("\n enter the number of discs:\n");
scanf("%d",&n);
printf("\n...\n");
tower(n,'A','B','C');
printf("\n...\n");
printf("\n total of %d disk takes %d moves\n\n",n,count);
}
```

OUTPUT:

cc 2.c OR gcc 2.c
./a.out

```
enter the number of discs: 2
...
move disk 1 from A to B
move 2 from A to C
move disk 1 from B to C
...
total of 2 disk takes 3 moves


enter the number of discs: 3
...
move disk 1 from A to C
move 2 from A to B
```

move disk 1 from C to B
move 3 from A to C
move disk 1 from B to A
move 2 from B to C
move disk 1 from A to C
...
total of 3 disk takes 7 moves

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

```
#include<stdio.h>
#include<stdlib.h>
#define MAXSIZE 3
int choice, item, front, rear, count, q[MAXSIZE];
int front=0, rear=-1, count=0;
void InsertQ()
{
   if(count==MAXSIZE)
   {
      printf("Queue Overflow\n");
      return;
   }
   rear=(rear+1)%MAXSIZE;
   q[rear]=item;
   printf("\n rear=%d front=%d\n",rear, front);
   count++;
}
int DeleteQ()
{
   int item;
   if(count==0)
   return -1;
   item=q[front];
   front=(front+1)%MAXSIZE;
   count-=1;
   return item;
}
void Display()
{
   int i;
   if(count==0)
   {
      printf("Queue is empty\n");
      return;
   }
```

```
    printf("contents of queue is\n");
    for(i=0;i<MAXSIZE;i++)
    printf("%d\n", q[i]);
}
void main()
{
    do
    {
    printf("1. InsertQ\n 2. DeleteQ\n 3. Display\n 4.exit\n");
    printf("Enter the choice:");
    scanf("%d",&choice);
    switch(choice)
    {
        case 1: printf("Enter the item to be inserted:");
                scanf("%d",&item);
                InsertQ();
                break;
        case 2: item=DeleteQ();
                if(count==0)
                {
                   printf("Queue is empty\n");
                }
                printf("Item deleted=%d\n",item);
                printf("rear=%d front=%d\n",rear,front);
                break;
        case 3: Display();
                break;
        case 4: exit(0);
        default: printf("Invalid choice\n");
    }
    }while(choice!=4);
}
```

OUTPUT:

cc 3.c OR gcc 3.c
./a.out

```
1. InsertQ
2. DeleteQ
3. Display
4. Exit
Enter the choice:2
Queue Underflow
```

1. InsertQ
2. DeleteQ
3. Display
4. Exit
Enter the choice:1
Enter the item to be inserted:10
rear=0 front=0

1. InsertQ
2. DeleteQ
3. Display
4. Exit
Enter the choice:1
Enter the item to be inserted:20
rear=1 front=0

1. InsertQ
2. DeleteQ
3. Display
4. Exit
Enter the choice:1
Enter the item to be inserted:30
rear=2 front=0

1. InsertQ
2. DeleteQ
3. Display
4. Exit
Enter the choice:1
Enter the item to be inserted:40
Queue Overflow

1. InsertQ
2. DeleteQ
3. Display
4. Exit
Enter the choice:3
contents of queue is
10
20
30

1. InsertQ
2. DeleteQ
3. Display
4. Exit
Enter the choice: 2
Item deleted=10
rear=2 front=1

1. InsertQ
2. DeleteQ
3. Display
4. Exit
Enter the choice:2
Item deleted=20
rear=2 front=2

1. InsertQ
2. DeleteQ
3. Display
4. Exit
Enter the choice:1
Enter the item to be inserted:40
rear=0 front=2

1. InsertQ
2. DeleteQ
3. Display
4. Exit
Enter the choice:1
Enter the item to be inserted:50
rear=1 front=2

1. InsertQ
2. DeleteQ
3. Display
4. Exit
Enter the choice:3
contents of queue is
40
50
30

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, 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<stdlib.h>
struct node
{
char usn[10];
char name[20];
char branch[10];
int sem;
char pno[10];
struct node *next;              //Self referential structure
};
typedef struct node *NODE;
NODE start=NULL;
void insert_front();
void insert_end();
void delete_front();
void delete_end();
void display();
NODE read();
void main()
{
int choice,n,i,flag=0;
do
{
printf("-----SINGLY LINKED LIST MENU-----\n");
printf("1:CREATE\n2:DISPLAY\n3:INSERT AT END\n4:DELETE FROM
END\n5:INSERT AT FRONT\n6:DELETE FROM FRONT\n7:EXIT\n");
printf("Enter your choice\n");
scanf("%d",&choice);
switch(choice)
{
case 1: if(flag==0)
{
printf("\nEnter number of students:\n ");
scanf("%d",&n);
```

```
for(i=1;i<=n;i++)
{
printf("\nStudent %d details\n",i);
insert_front();
}
flag=1;
}
else
{
printf("\nStudent Details already exists. \n");
}
break;
case 2: display();
break;
case 3: insert_end();
break;
case 4: delete_end();
break;
case 5: insert_front();
break;
case 6: delete_front();
break;
case 7: exit(0);
default: printf("Invalid Choice\n");
}
}while(choice!=7);
}
NODE read()
{
NODE t;
t=(struct node *)malloc(sizeof(struct node));
printf("Enter USN: ");
scanf("%s",t->usn);
printf("Enter Name: ");
scanf("%s",t->name);
printf("Enter branch: ");
scanf("%s",t->branch);
printf("Enter Semester: ");
scanf("%d",&t->sem);
printf("Enter phone number: ");
scanf("%s",t->pno);
return(t);
```

```
}
void insert_front()
{
NODE temp; temp=read();
if(start==NULL)
{
temp->next=NULL;
start=temp;
}
else
{
temp->next=start;
start=temp;
}
}
void insert_end()
{
int num;
NODE q,temp; temp=read();
temp->next=NULL;
if(start==NULL)
{
start=temp;
}
else
{
q=start;
while(q->next!=NULL)
q=q->next;
q->next=temp;
}
}
void delete_front()
{
NODE q;
if(start==NULL)
{
printf("\nThe list is empty. \n");
}
else
{
q=start;
```

```
start=start->next;
printf("\nDeleted Student USN is %s \n\n",q->usn);
free(q);
}
}
void delete_end()
{
NODE q,t;
if(start==NULL)
{
printf("\nThe list is empty. \n");
}
else if(start->next==NULL)
{
printf("\nDeleted Student USN is %s \n\n",start->usn);
start=NULL; free(start);
}
else
{
q=start;
while(q->next->next!=NULL)
q=q->next;
t=q->next;
q->next=NULL;
printf("\nDeleted Student USN is %s \n\n",t->usn);
free(t);
}
}
void display()
{
int count=0;
NODE q;
if(start==NULL)
{
printf("\nList is empty. \n");
}
else
{
q=start;
printf("\nThe Student details are:\n");
printf("\nUSN\tName\tBranch\tSemester\tPhone Number\n");
printf("---------------------------------------------------------------\n");
```

```
while(q!=NULL)
{
printf("\n%s\t%s\t%s\t%d\t%s\n",q->usn,q->name,q->branch,q->sem,q->pno);
q=q->next; count++;
}
}
printf("\nTotal Number of Student Details in the List are: %d\n",count);
}
```

OUTPUT:

cc 7.c OR gcc 7.c
./a.out

```
-----SINGLY LINKED LIST MENU-----
1:CREATE
2:DISPLAY
3:INSERT AT END
4:DELETE FROM END
5:INSERT AT FRONT
6:DELETE FROM FRONT
7:EXIT
Enter your choice
2
List is empty.
Total Number of Student Details in the List are: 0

-----SINGLY LINKED LIST MENU-----
1:CREATE
2:DISPLAY
3:INSERT AT END
4:DELETE FROM END
5:INSERT AT FRONT
6:DELETE FROM FRONT
7:EXIT
Enter your choice
4
The list is empty.

-----SINGLY LINKED LIST MENU-----
1:CREATE
2:DISPLAY
3:INSERT AT END
4:DELETE FROM END
5:INSERT AT FRONT
6:DELETE FROM FRONT
7:EXIT
Enter your choice
6
The list is empty.
-----SINGLY LINKED LIST MENU-----
```

```
1:CREATE
2:DISPLAY
3:INSERT AT END
4:DELETE FROM END
5:INSERT AT FRONT
6:DELETE FROM FRONT
7:EXIT
Enter your choice
1
Enter number of students:
3
Student 1 details
Enter USN: 1
Enter Name: A
Enter branch: CS
Enter Semester: 1
Enter phone number: 12345
Student 2 details
Enter USN: 2
Enter Name: B
Enter branch: CS
Enter Semester: 3
Enter phone number: 12344
Student 3 details
Enter USN: 3
Enter Name: C
Enter branch: CS
Enter Semester: 5
Enter phone number: 235678

-----SINGLY LINKED LIST MENU-----
1:CREATE
2:DISPLAY
3:INSERT AT END
4:DELETE FROM END
5:INSERT AT FRONT
6:DELETE FROM FRONT
7:EXIT
Enter your choice
2
The Student details are:
USN    Name   Branch    Semester    Phone Number
--------------------------------------------------------------------------------
  3         C         CS           5            235678
  2         B         CS           3            12344
  1         A         CS           1            12345
Total Number of Student Details in the List are: 3

          -----SINGLY LINKED LIST MENU-----
1:CREATE
2:DISPLAY
3:INSERT AT END
4:DELETE FROM END
```

```
5:INSERT AT FRONT
6:DELETE FROM FRONT
7:EXIT
Enter your choice
4
Deleted Student USN is 1

-----SINGLY LINKED LIST MENU-----
1:CREATE
2:DISPLAY
3:INSERT AT END
4:DELETE FROM END
5:INSERT AT FRONT
6:DELETE FROM FRONT
7:EXIT
Enter your choice
2
The Student details are:
USN   Name   Branch   Semester   Phone Number
--------------------------------------------------------------------------------
  3        C       CS          5          235678
  2        B       CS          3          12344
Total Number of Student Details in the List are: 2

-----SINGLY LINKED LIST MENU-----
1:CREATE
2:DISPLAY
3:INSERT AT END
4:DELETE FROM END
5:INSERT AT FRONT
6:DELETE FROM FRONT
7:EXIT
Enter your choice
3
Enter USN: 5
Enter Name: M
Enter branch: CS
Enter Semester: 7
Enter phone number: 12222

-----SINGLY LINKED LIST MENU-----
1:CREATE
2:DISPLAY
3:INSERT AT END
4:DELETE FROM END
5:INSERT AT FRONT
6:DELETE FROM FRONT
7:EXIT
Enter your choice
2


The Student details are:
```

```
USN Name Branch Semester Phone Number
--------------------------------------------------------------------------------
  3      C       CS        5         235678
  2      B       CS        3         12344
  5      M       CS        7         12222
Total Number of Student Details in the List are: 3

-----SINGLY LINKED LIST MENU-----
1:CREATE
2:DISPLAY
3:INSERT AT END
4:DELETE FROM END
5:INSERT AT FRONT
6:DELETE FROM FRONT
7:EXIT
Enter your choice
5
Enter USN: 100
Enter Name: S
Enter branch: CS
Enter Semester: 7
Enter phone number: 122345

-----SINGLY LINKED LIST MENU-----
1:CREATE
2:DISPLAY
3:INSERT AT END
4:DELETE FROM END
5:INSERT AT FRONT
6:DELETE FROM FRONT
7:EXIT
Enter your choice
2
The Student details are:
USN   Name    Branch    Semester    Phone Number
--------------------------------------------------------------------------------
 100     S        CS           7           122345
   3     C        CS           5           235678
   2     B        CS           3           12344
   5     M        CS           7           12222
Total Number of Student Details in the List are: 4

-----SINGLY LINKED LIST MENU-----
1:CREATE
2:DISPLAY
3:INSERT AT END
4:DELETE FROM END
5:INSERT AT FRONT
6:DELETE FROM FRONT
7:EXIT
Enter your choice
7
```

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 node
{
char ssn[20];
char name[20];
char dept[20];
char desig[20];
long int sal;
char pno[12];
struct node *next;
struct node *prev;
};
typedef struct node *NODE;
NODE start=NULL;
                                    //function calls
void insert_front( );
void insert_end( );
void delete_front( );
void delete_end( );
void display( );
NODE read( );
void main( )                        //main method
{
int choice,n,i,flag=0;
do
{
printf("\n-----DOUBLY LINKED LIST MENU-----\n");
printf("\n1:CREATE DLL\n2:DISPLAY DLL\n");
printf("\nDOUBLE ENDED QUEUE OPERATIONS\n");
printf("\n3:INSERT AT END OF DLL\n4:DELETE FROM END OF DLL\n5:INSERT AT
FRONT OF DLL \n6:DELETE FROM FRONT OF DLL\n7:EXIT\n");
printf("\nEnter your choice\n");
```

```
scanf("%d",&choice);
switch(choice)
{
case 1: if(flag==0)
{
printf("Enter number of Employees:\n ");
scanf("%d",&n);
for(i=1;i<=n;i++)
{
printf("Employee %d details\n",i);
insert_end();
}
flag=1;
}
else
{
printf("\nEmployee Details already exists. \n");
}
break;
case 2: display();
break;
case 3: insert_end();
break;
case 4: delete_end();
break;
case 5: insert_front();
break;
case 6: delete_front();
break;
case 7: exit(0);
default: printf("Invalid Choice\n");
}
} while(choice!=7);
}
NODE read()
{
NODE t;
t=(struct node *)malloc(sizeof(struct node));
printf("Enter SSN: ");
scanf("%s",t->ssn);
printf("Enter Name: ");
scanf("%s",t->name);
```

```
printf("Enter Department: ");
scanf("%s",t->dept);
printf("Enter Designation: ");
scanf("%s",t->desig);
printf("Enter Salary: ");
scanf("%ld",&t->sal);
printf("Enter phone number: ");
scanf("%s",t->pno);
return(t);
}
void insert_front()
{
NODE temp;
temp=read();
if(start==NULL)
{
temp->next=NULL;
temp->prev=NULL;
start=temp;
}
else
{
temp->next=start;
temp->prev=NULL;
start->prev=temp;
start=temp;
}
}
void insert_end()
{
NODE q,temp;
temp=read();
temp->next=NULL;
if(start==NULL)
{
start=temp;
temp->prev=NULL;
}
else
{
q=start;
while(q->next!=NULL)
```

```
q=q->next;
temp->prev=q;
q->next=temp;
}
}
void delete_front()
{
NODE q;
if(start==NULL)
{
printf("The DLL list is empty. \n");
}
else if(start->next==NULL)
{
printf("Deleted Employee SSN is %s \n\n",start->ssn);
start=NULL;
free(start);
}
else
{
q=start;
printf("Deleted Employee SSN is %s \n\n",q->ssn);
start=start->next;
free(q);
start->prev=NULL;
}
}
void delete_end()
{
NODE q,t;
if(start==NULL)
{
printf("The DLL list is empty. \n");
}
else if(start->next==NULL)
{
printf("Deleted Employee SSN is %s \n\n",start->ssn);
start=NULL;
free(start);
}
else
{
```

```
q=start;
while(q->next->next!=NULL)
{
q=q->next;
}
t=q->next;
printf("\nDeleted Employee SSN is %s \n\n",t->ssn);
q->next=NULL;
t->prev=NULL;
free(t);
}
}
void display()
{
int count=0;
NODE q;
if(start==NULL)
{
printf(" DLL List is empty. \n");
}
else
{
q=start;
printf("\nThe Employee details are:\n");
printf("\nEmp_SSN\tEmp_Name\tDepartment\tDesignation\tSalary\tPhone_No\n");
printf("\n-------------------------------------------------------------------------------------\n");
while(q!=NULL)
{
printf("\n%s\t\t%s\t\t%s\t\t%s\t%ld\t%s\n",q->ssn,q->name,q->dept,q->desig, q->sal,q->pno);
q=q->next;
count++;
}
}
printf("\nTotal Number of Employee in the DLL List are: %d\n",count);
}
```

OUTPUT:

cc 8.c OR gcc 8.c
./a.out

```
-----DOUBLY LINKED LIST MENU-----
1:CREATE DLL
2:DISPLAY DLL
```

DOUBLE ENDED QUEUE OPERATIONS
3:INSERT AT END OF DLL
4:DELETE FROM END OF DLL
5:INSERT AT FRONT OF DLL
6:DELETE FROM FRONT OF DLL
7:EXIT
Enter your choice
2
DLL List is empty.
Total Number of Employee in the DLL List are: 0

-----DOUBLY LINKED LIST MENU-----
1:CREATE DLL
2:DISPLAY DLL
DOUBLE ENDED QUEUE OPERATIONS
3:INSERT AT END OF DLL
4:DELETE FROM END OF DLL
5:INSERT AT FRONT OF DLL
6:DELETE FROM FRONT OF DLL
7:EXIT
Enter your choice
4
The DLL list is empty.

-----DOUBLY LINKED LIST MENU-----
1:CREATE DLL
2:DISPLAY DLL
DOUBLE ENDED QUEUE OPERATIONS
3:INSERT AT END OF DLL
4:DELETE FROM END OF DLL
5:INSERT AT FRONT OF DLL
6:DELETE FROM FRONT OF DLL
7:EXIT
Enter your choice
6
The DLL list is empty.

-----DOUBLY LINKED LIST MENU-----
1:CREATE DLL
2:DISPLAY DLL
DOUBLE ENDED QUEUE OPERATIONS
3:INSERT AT END OF DLL
4:DELETE FROM END OF DLL
5:INSERT AT FRONT OF DLL
6:DELETE FROM FRONT OF DLL
7:EXIT
Enter your choice
1
Enter number of Employees:
2
Employee 1 details
Enter SSN: 1
Enter Name: A

```
Enter Department: CS
Enter Designation: AF
Enter Salary: 34567
Enter phone number: 1222
Employee 2 details
Enter SSN: 2
Enter Name: B
Enter Department: CS
Enter Designation: JF
Enter Salary: 32145
Enter phone number: 34567

-----DOUBLY LINKED LIST MENU-----
1:CREATE DLL
2:DISPLAY DLL
DOUBLE ENDED QUEUE OPERATIONS
3:INSERT AT END OF DLL
4:DELETE FROM END OF DLL
5:INSERT AT FRONT OF DLL
6:DELETE FROM FRONT OF DLL
7:EXIT
Enter your choice : 2
The Employee details are:
Emp_SSN Emp_Name Department Designation   Salary     Phone_No
-------------------------------------------------------------------------------------------------------
      1          A          CS           AF        34567       1222
      2          B          CS           JF        32145       34567
Total Number of Employee in the DLL List are: 2

-----DOUBLY LINKED LIST MENU-----
1:CREATE DLL
2:DISPLAY DLL
DOUBLE ENDED QUEUE OPERATIONS
3:INSERT AT END OF DLL
4:DELETE FROM END OF DLL
5:INSERT AT FRONT OF DLL
6:DELETE FROM FRONT OF DLL
7:EXIT
Enter your choice : 3
Enter SSN: 3
Enter Name: C
Enter Department: CS
Enter Designation: AP
Enter Salary: 45632
Enter phone number: 23456

-----DOUBLY LINKED LIST MENU-----
1:CREATE DLL
2:DISPLAY DLL
DOUBLE ENDED QUEUE OPERATIONS
3:INSERT AT END OF DLL
4:DELETE FROM END OF DLL
5:INSERT AT FRONT OF DLL
```

```
6:DELETE FROM FRONT OF DLL
7:EXIT
Enter your choice: 2
The Employee details are:
Emp_SSN Emp_Name Department Designation   Salary     Phone_No
-------------------------------------------------------------------------------------------------
      1          A          CS          AF        34567       1222
      2          B          CS          JF        32145       34567
      3          C          CS          AP        45632       23456
Total Number of Employee in the DLL List are: 3

-----DOUBLY LINKED LIST MENU-----
1:CREATE DLL
2:DISPLAY DLL
DOUBLE ENDED QUEUE OPERATIONS
3:INSERT AT END OF DLL
4:DELETE FROM END OF DLL
5:INSERT AT FRONT OF DLL
6:DELETE FROM FRONT OF DLL
7:EXIT
Enter your choice :3
Enter SSN: 99
Enter Name: M
Enter Department: CSE
Enter Designation: JRF
Enter Salary: 45643
Enter phone number: 345678

-----DOUBLY LINKED LIST MENU-----
1:CREATE DLL
2:DISPLAY DLL
DOUBLE ENDED QUEUE OPERATIONS
3:INSERT AT END OF DLL
4:DELETE FROM END OF DLL
5:INSERT AT FRONT OF DLL
6:DELETE FROM FRONT OF DLL
7:EXIT
Enter your choice: 2
The Employee details are:
Emp_SSN Emp_Name Department Designation    Salary    Phone_No
-------------------------------------------------------------------------------------------------------
      1          A          CS          AF        34567       1222
      2          B          CS          JF        32145       34567
      3          C          CS          AP        45632       23456
     99          M          CSE         JRF       45643       345678
Total Number of Employee in the DLL List are: 4

-----DOUBLY LINKED LIST MENU-----
1:CREATE DLL
2:DISPLAY DLL
DOUBLE ENDED QUEUE OPERATIONS
3:INSERT AT END OF DLL
4:DELETE FROM END OF DLL
```

```
5:INSERT AT FRONT OF DLL
6:DELETE FROM FRONT OF DLL
7:EXIT
Enter your choice: 6
Deleted Employee SSN is 1

-----DOUBLY LINKED LIST MENU-----
1:CREATE DLL
2:DISPLAY DLL
DOUBLE ENDED QUEUE OPERATIONS
3:INSERT AT END OF DLL
4:DELETE FROM END OF DLL
5:INSERT AT FRONT OF DLL
6:DELETE FROM FRONT OF DLL
7:EXIT
Enter your choice: 2
The Employee details are:
Emp_SSN Emp_Name Department Designation   Salary     Phone_No
-------------------------------------------------------------------------------------------------
      2          B          CS          JF          32145      34567
      3          C          CS          AP          45632      23456
     99          M         CSE         JRF          45643     345678
Total Number of Employee in the DLL List are: 3

-----DOUBLY LINKED LIST MENU-----
1:CREATE DLL
2:DISPLAY DLL
DOUBLE ENDED QUEUE OPERATIONS
3:INSERT AT END OF DLL
4:DELETE FROM END OF DLL
5:INSERT AT FRONT OF DLL
6:DELETE FROM FRONT OF DLL
7:EXIT
Enter your choice: 7
```

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) = 6x2 y 2 z-4yz5 +3x3 yz+2xy5 z-2xyz3

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<math.h>
#include<stdlib.h>
struct node
{
int cf, px, py, pz;
int flag;
struct node *link;
};
typedef struct node NODE;
NODE* getnode()
{
NODE *x;
x=(NODE*)malloc(sizeof(NODE));
if(x==NULL)
{
printf("Insufficient memory\n");
exit(0);
}
return x;
}
void display(NODE *head)
{
NODE *temp;
if(head->link==head)
{
printf("Polynomial does not exist\n");
return;
}
temp=head->link;
printf("\n");
while(temp!=head)
{
printf("%d x^%d y^%d z^%d",temp->cf,temp->px,temp->py,temp->pz);
if(temp->link != head)
printf(" + ");
```

```
temp=temp->link;
}
printf("\n");
}
NODE* insert_rear(int cf,int x,int y,int z,NODE *head)
{
NODE *temp,*cur;
temp=getnode();
temp->cf=cf;
temp->px=x;
temp->py=y;
temp->pz=z;
cur=head->link;
while(cur->link!=head)
{
cur=cur->link;
}
cur->link=temp;
temp->link=head;
return head;
}
NODE* read_poly(NODE *head)
{
int px, py, pz, cf, ch;
printf("\nEnter coeff: ");
scanf("%d",&cf);
printf("\nEnter x, y, z powers(0-indiacate NO term): ");
scanf("%d%d%d", &px, &py, &pz);
head=insert_rear(cf,px,py,pz,head);
printf("\nIf you wish to continue press 1 otherwise 0: ");
scanf("%d", &ch);
while(ch != 0)
{
printf("\nEnter coeff: ");
scanf("%d",&cf);
printf("\nEnter x, y, z powers(0-indiacate NO term): ");
scanf("%d%d%d", &px, &py, &pz);
head=insert_rear(cf,px,py,pz,head);
printf("\nIf you wish to continue press 1 otherwise 0: ");
scanf("%d", &ch);
}
return head;
```

```
}
NODE*  add_poly(NODE *h1,NODE *h2,NODE *h3)
{
NODE *p1,*p2;
int x1,x2,y1,y2,z1,z2,cf1,cf2,cf;
p1=h1->link;
while(p1!=h1)
{
x1=p1->px;
y1=p1->py;
z1=p1->pz;
cf1=p1->cf;
p2=h2->link;
while(p2!=h2)
{
x2=p2->px;
y2=p2->py;
z2=p2->pz;
cf2=p2->cf;
if(x1==x2 && y1==y2 && z1==z2)
break;
p2=p2->link;
}
if(p2!=h2)
{
cf=cf1+cf2;
p2->flag=1;
if(cf!=0)
h3=insert_rear(cf,x1,y1,z1,h3);
}
else
{
h3=insert_rear(cf1,x1,y1,z1,h3);
p1=p1->link;
}
p2=h2->link;
while(p2!=h2)
{
if(p2->flag==0)
h3=insert_rear(p2->cf,p2->px,p2->py,p2->pz,h3);
p2=p2->link;
}
```

```
return h3;
}
}
void evaluate(NODE *h1)
{
NODE *head;
int x, y, z;
float result=0.0;
head=h1;
printf("\nEnter x, y, z, terms to evaluate:\n");
scanf("%d%d%d", &x, &y, &z);
while(h1->link != head)
{
result = result + (h1->cf * pow(x,h1->px) * pow(y,h1->py) *
pow(z,h1->pz));
h1=h1->link;
}
result = result + (h1->cf * pow(x,h1->px) * pow(y,h1->py) *
pow(z,h1->pz));
printf("\nPolynomial result is: %f", result);
}
void main()
{
NODE *h1,*h2,*h3;
int ch;
h1=getnode();
h2=getnode();
h3=getnode();
h1->link=h1;
h2->link=h2;
h3->link=h3;
while(1)
{
printf("\n1.Evaluate polynomial\n2.Add two polynomials\n3.Exit\n");
printf("Enter your choice: ");
scanf("%d", &ch);
switch(ch)
{
case 1: printf("\nEnter polynomial to evaluate:\n");
h1=read_poly(h1);
display(h1);
evaluate(h1);
```

```
break;
case 2: printf("\nEnter the first polynomial:");
h1=read_poly(h1);
printf("\nEnter the second polynomial:");
h2=read_poly(h2);
h3=add_poly(h1,h2,h3);
printf("\nFirst polynomial is: ");
display(h1);
printf("\nSecond polynomial is: ");
display(h2);
printf("\nThe sum of 2 polynomials is: ");
display(h3);
break;
case 3:exit(0);
break;
default: printf("\nInvalid entry");
break;
}
}
}
```

OUTPUT:

cc 9.c OR gcc 9.c
./a.out

```
1. Evaluate polynomial
2. Add two polynomials
3.Exit
Enter your choice: 1
Enter polynomial to evaluate:
Enter coeff: 3
Enter x, y, z powers(0-indiacate NO term): 3 1 1
If you wish to continue press 1 otherwise 0: 0
3 x^3 y^1 z^1
Enter x, y, z, terms to evaluate:
1
1
1
Polynomial result is: 3.000000

1.Evaluate polynomial
2.Add two polynomials
3.Exit
Enter your choice: 1
Enter polynomial to evaluate:
```

```
Enter coeff: 6
Enter x, y, z powers(0-indiacate NO term): 2
2
1
If you wish to continue press 1 otherwise 0: 0
3 x^3 y^1 z^1 + 6 x^2 y^2 z^1
Enter x, y, z, terms to evaluate:
0 1 5
Polynomial result is: 0.000000

1.Evaluate polynomial
2.Add two polynomials
3.Exit
Enter your choice: 2
Enter the first polynomial:
Enter coeff: 2
Enter x, y, z powers(0-indiacate NO term): 4
5
6
If you wish to continue press 1 otherwise 0: 1
Enter coeff: 4
Enter x, y, z powers(0-indiacate NO term): 2
3
1
If you wish to continue press 1 otherwise 0: 0
Enter the second polynomial:
Enter coeff: 2
Enter x, y, z powers(0-indiacate NO term): 1
1
2
If you wish to continue press 1 otherwise 0: 0
First polynomial is:
3 x^3 y^1 z^1 + 6 x^2 y^2 z^1 + 2 x^4 y^5 z^6 + 4 x^2 y^3 z^1
Second polynomial is:
2 x^1 y^1 z^2
The sum of 2 polynomials is:
3 x^3 y^1 z^1 + 6 x^2 y^2 z^1 + 2 x^4 y^5 z^6 + 4 x^2 y^3 z^1 + 2 x^1 y^1 z^2

1.Evaluate polynomial
2.Add two polynomials
3.Exit
Enter your choice: 3

1.Evaluate polynomial
2.Add two polynomials
3.Exit
Enter your choice: 2
Enter the first polynomial:
```

```
Enter coeff: 4
Enter x, y, z powers(0-indiacate NO term): 2
2
2
If you wish to continue press 1 otherwise 0: 1
Enter coeff: 3
Enter x, y, z powers(0-indiacate NO term): 1
1
2
If you wish to continue press 1 otherwise 0: 0
Enter the second polynomial:
Enter coeff: 6
Enter x, y, z powers(0-indiacate NO term): 2
2
2
If you wish to continue press 1 otherwise 0: 0
First polynomial is:
4 x^2 y^2 z^2 + 3 x^1 y^1 z^2
Second polynomial is:
6 x^2 y^2 z^2
The sum of 2 polynomials is:
10 x^2 y^2 z^2 + 3 x^1 y^1 z^2
1.Evaluate polynomial
2.Add two polynomials
3.Exit
Enter your choice: 3
```

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

```
#include <stdio.h>
#include <stdlib.h>
struct BST
{
int data;
struct BST *left;
struct BST *right;
};
typedef struct BST NODE;
NODE *node;
NODE* createtree(NODE *node, int data)
{
if (node == NULL)
{
NODE *temp;
temp= (NODE*)malloc(sizeof(NODE));
temp->data = data;
temp->left = temp->right = NULL;
return temp;
}
if (data < (node->data))
{
node->left = createtree(node->left, data);
}
else if (data > node->data)
{
node -> right = createtree(node->right, data);
}
return node;
}
NODE* search(NODE *node, int data)
{
if(node == NULL)
printf("\nElement not found");
else if(data < node->data)
{
```

```
node->left=search(node->left, data);
}
else if(data > node->data)
{
node->right=search(node->right, data);
}
else
printf("\nElement found is: %d", node->data);
return node;
}
void inorder(NODE *node)
{
if(node != NULL)
{
inorder(node->left);
printf("%d\t", node->data);
inorder(node->right);
}
}
void preorder(NODE *node)
{
if(node != NULL)
{
printf("%d\t", node->data);
preorder(node->left);
preorder(node->right);
}
}
void postorder(NODE *node)
{
if(node != NULL)
{
postorder(node->left);
postorder(node->right);
printf("%d\t", node->data);
}
}
void main()
{
int data, ch, i, n;
NODE *root=NULL;
while (1)
```

```
{
printf("\n1.Insertion in Binary Search Tree");
printf("\n2.Search Element in Binary Search Tree");
printf("\n3.Inorder\n 4.Preorder\n 5.Postorder\n 6.Exit");
printf("\nEnter your choice: ");
scanf("%d", &ch);
switch (ch)
{
case 1: printf("\nEnter N value: " );
scanf("%d", &n);
printf("\nEnter-the-values-to-create-BSTn like(6,9,5,2,8,15,24,14,7,8,5,2)\n");
for(i=0; i<n; i++)
{
scanf("%d", &data);
root=createtree(root, data);
}
break;
case 2: printf("\nEnter the element to search: ");
scanf("%d", &data);
root=search(root, data);
break;
case 3: printf("\nInorder Traversal: \n");
inorder(root);
break;
case 4: printf("\nPreorder Traversal: \n");
preorder(root);
break;
case 5: printf("\nPostorder Traversal: \n");
postorder(root);
break;
case 6: exit(0);
default: printf("\nInvalid output");
break;
}
}
}
```

OUTPUT:

cc 3.c OR gcc 3.c
./a.out

```
1.Insertion in Binary Search Tree
2.Search Element in Binary Search Tree
```

3.Inorder
4.Preorder
5.Postorder
6.Exit
Enter your choice: 1
Enter N value: 12
Enter the values to create BST like(6,9,5,2,8,15,24,14,7,8,5,2)
6
9
5
2
8
15
24
14
7
8
5
2

1.Insertion in Binary Search Tree
2.Search Element in Binary Search Tree
3.Inorder
4.Preorder
5.Postorder
6.Exit
Enter your choice: 3
Inorder Traversal:
2 5 6 7 8 9 14 15 24

1.Insertion in Binary Search Tree
2.Search Element in Binary Search Tree
3.Inorder
4.Preorder
5.Postorder
6.Exit
Enter your choice: 4
Preorder Traversal:
6 5 2 9 8 7 15 14 24

1.Insertion in Binary Search Tree
2.Search Element in Binary Search Tree
3.Inorder
4.Preorder
5.Postorder
6.Exit
Enter your choice: 5
Postorder Traversal:

2 5 7 8 14 24 15 9 6

1.Insertion in Binary Search Tree
2.Search Element in Binary Search Tree
3.Inorder
4.Preorder
5.Postorder
6.Exit
Enter your choice: 2
Enter the element to search: 7
Element found is: 7

1.Insertion in Binary Search Tree
2.Search Element in Binary Search Tree
3.Inorder
4.Preorder
5.Postorder
6.Exit
Enter your choice: 2
Enter the element to search: 100
Element not found

1.Insertion in Binary Search Tree
2.Inorder
3.Preorder
4.Postorder
5.Exit
Enter your choice: 6

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 diagraph using DFS/BFS method.

```
#include<stdio.h>
#include<conio.h>
int a[10][10], n, m, i, j, source, s[10], b[10];
int visited[10];
void create()
{
printf("\nEnter the number of vertices of the digraph: ");
scanf("%d", &n);
printf("\nEnter the adjacency matrix of the graph:\n");
for(i=1; i<=n; i++)
for(j=1; j<=n; j++)
scanf("%d", &a[i][j]);
}
void bfs()
{
int q[10], u, front=0, rear=-1;
printf("\nEnter the source vertex to find other nodes reachable or not: ");
scanf("%d", &source);
q[++rear] = source;
visited[source] = 1;
printf("\nThe reachable vertices are: ");
while(front<=rear)
{
u = q[front++];
for(i=1; i<=n; i++)
{
if(a[u][i] == 1 && visited[i] == 0)
{
q[++rear] = i;
visited[i] = 1;
printf("\n%d", i);
}
}
}
}
void dfs(int source)
{
int v, top = -1;
s[++top] = 1;
```

```
b[source] = 1;
for(v=1; v<=n; v++)
{
if(a[source][v] == 1 && b[v] == 0)
{
printf("\n%d -> %d", source, v); dfs(v);
}
}
}
void main()
{
int ch;
while(1)
{
printf("\n1.Create Graph\n2.BFS\n3.Check graph connected or not(DFS)\n4.Exit");
printf("\nEnter your choice: ");
scanf("%d", &ch);
switch(ch)
{
case 1: create();
break;
case 2: bfs();
for(i=1;i<=n;i++)
if(visited[i]==0)
printf("\nThe vertex that is not rechable %d" ,i);
break;
case 3: printf("\nEnter the source vertex to find the connectivity: ");
scanf("%d",&source);
m=1;
dfs(source);
for(i=1;i<=n;i++)
{
if(b[i]==0)
m=0;
}
if(m==1)
printf("\nGraph is Connected");
else
printf("\nGraph is not Connected");
break;
default: exit(0);
}
}
}
```

OUTPUT:

cc 3.c OR gcc 3.c
./a.out

```
1. Create Graph
2.BFS
3.Check graph connected or not (DFS)
4.Exit
Enter your choice: 1
Enter the number of vertices of the digraph: 4
Enter the adjacency matrix of the graph:
0       0       1       1
0       0       0       0
0       1       0       0
0       1       0       0

1.Create Graph
2.BFS
3.Check graph connected or not (DFS)
4.Exit
Enter your choice: 2
Enter the source vertex to find other nodes reachable or not: 1
The reachable vertices are:
3
4
2

1. Create Graph
2.BFS
3.Check graph connected or not (DFS)
4.Exit
Enter your choice: 3
Enter the source vertex to find the connectivity: 1
1 -> 3
3 -> 2
1 -> 4
Graph is Connected

1. Create Graph
2.BFS
3.Check graph connected or not (DFS)
4.Exit
Enter your choice: 3
Enter the source vertex to find the connectivity: 2
Graph is not Connected

1. Create Graph
2.BFS
3.Check graph connected or not (DFS)
4.Exit
Enter your choice: 4
```