Lab Experiments

L.1 **In Second year Computer Engineering class of M students, set A of students play cricket and set B of students play badminton. Write C/C++ program to find and display-**

i. **Set of students who play either cricket or badminton or both**

ii. **Set of students who play both cricket and badminton**

iii. **Set of students who play only cricket**

iv. **Set of students who play only badminton**

v. **Number of students who play neither cricket nor badminton**

(Note : While realizing the set duplicate entries are to avoided)

Solution :

```
/*Program for various operations on set.
 Operations covered :
1) Create(), : for creating a new set with initial members
               of the set
2) print()   : diaplays all members of the set
3) Union()  : finds union of two sets, set1[] and set2[]  and
              stores the result in set3[]
4) intersection() : finds intersection of two sets, set1[] and
                    set2[] and stores the result in set3[]
5) difference() :   finds difference of two sets, set1[] and
                    set2[] and stores the result in set3[]
6) membership(): function returns 1 or 0 ,depending
                 onwhether the element x belongs or not to
                 a  set, set1[].
7) insert() :   function inserts a new element x in a given
                set,
                 set1[].
8) Delete() :  function deletes an element x from the given
                set,
                 set1[].

Representation of a set
-----------------------
   A set is representrd using an ordered array of integers.
   It may be declared as:
       int set[30];
   set[0] - gives number of elements in a set.
   set[1] to set[29] are for storing set members.

Example :
   A set,[2,11,3,5 6],when represebted will appear as:
    [5][2][3][5][6][11][ ][ ][ ] <--- array set[]
    0 1 2 3 4 5  6 7 8  <--- index
*/
#define MAX 30
#include<stdio.h>
#include<conio.h>
void create(int set[]);
void print(int set[]);
void Union(int set1[],int set2[],int set3[]);
void intersection(int set1[],int set2[],int set3[]);
void difference(int set1[],int set2[],int set3[]);
int member(int set[],int x);
int subset(int set1[],int set2[]);
void insert(int set[],int x);
void Delete(int set[],int x);
void main()
{ int set1[MAX],set2[MAX],set3[MAX];
 int x,op,M,n;
 clrscr();
 flushall();
 set1[0]=set2[0]=set3[0]=0;
 printf("\nEnter No. of students : ");
 scanf("%d",&M);
 printf("\nCreating Set of students playing
cricket*******");
 create(set1);
 printf("\nCreating Set of students playing
badminton*****");
 create(set2);
 Union(set1,set2,set3);
 printf("\n Students Playing Cricket or Badminton : ");
 print(set3);
 intersection(set1,set2,set3);
```

```
n=M-set1[0]-set2[0]+set3[0];
printf("\n Students Playing both Cricket and Badminton:");
print(set3);
difference(set1,set2,set3);
printf("\nStudents playing only Cricket : ");
print(set3);
difference(set2,set1,set3);
printf("\n Students playing only Badminton : ");
print(set3);
printf("\nNumber of students playing neither cricket nor
badminton : %d",n);
getch();
}
/*creates set[] with initial elements*/
void create(int set[])
 { int n,i,x;
   set[0]=0;/*make it a null set*/
   printf("\n No. of elements in the set:");
   scanf("%d",&n);
   printf("\n enter set elements(integers only) :");
   for(i=1;i<=n;i++)
    { scanf("%d",&x);
     insert(set,x);
     }
}
 /*a new element x is inserted in set[]. Sorted nature of list
 is maintained after insertion*/
 void insert(int set[],int x)
{ int i,n;
     n=set[0]; /* number of elements in the set*/
     for(i=n;i>=1 && set[i]>x ;i--)
       set[i+1]=set[i];
     set[i+1]=x;
     set[0]++;
}
 void print(int set[])
{ int i,n;
  n=set[0];/* number of elements in the set */
  printf("\n-->");
  for(i=1;i<=n;i++)
   printf("%d ",set[i]);
}
/* union of set1[] and set2[] is stored in set3[]*/
void Union(int set1[],int set2[],int set3[])
{ int i,n;
  /* copy set1[] to set3[]*/
  set3[0]=0;/*make set3[] a null set */
  n=set1[0];/* number of elements in the set*/
  for(i=0;i<=n;i++)
    set3[i]=set1[i];
  n=set2[0];
  for(i=1;i<=n;i++)
   if(!member(set3,set2[i]))
    insert(set3,set2[i]);
}
/*function returns 1 or 0 depending on whether x belongs
 to set[] or not */
 int member(int set[],int x)
{ int i,n;
  n=set[0]; /* number of elements in the set*/
  for(i=1;i<=n;i++)
   if(x==set[i])
    return(1);
   return(0);
}
/*intersection of set1[] and set2[] is stored in set3[]*/
void intersection(int set1[],int set2[],int set3[])
{ int i,n;
     set3[0]=0; /* make a NULL set*/
     n=set1[0];/* number of elements in the set*/
     for(i=1;i<=n;i++)
      if(member(set2,set1[i])) /* all common elements are
      inserted in
       set3[]*/
        insert(set3,set1[i]);
}
/*difference of set1[] and set2[] is stored in set3[]*/
void difference(int set1[],int set2[],int set3[])
    { int i,n;
     n=set1[0];/* number of elements in the set*/
     set3[0]=0;/*make it a null set*/
     for(i=1;i<=n;i++)
      if(!member(set2,set1[i]))
       insert(set3,set1[i]);
}
/* function returns 1 or 0 depending on whether set1[] is a
subset of
   set2[] or not */
 int subset(int set1[],int set2[])
{    int i,n;
     n=set1[0];
     for(i=1;i<=n;i++)
      if(!member(set2,set1[i]))
      return(0);
      return(1);
}
void Delete(int set[],int x)
```

```
{ int i,n;
  n=set[0];
  /*locate the element x*/
  i=1;
  while(set[i]!=x)
    i++;
  if(i<=n)
{   /*delete the element*/
    while(i<n)
    {    set[i]=set[i+1];
         i++;
    }
    set[0]--;
    }
}
```

Output

```
Enter No. of students : 5
Creating Set of students playing cricket*******
No. of elements in the set:2

enter set elements(integers only) :1 2

Creating Set of students playing badminton*****
No. of elements in the set:2

enter set elements(integers only) :2 3

Students Playing Cricket or Badminton :
-->1 2 3
Students Playing both Cricket and Badminton :
-->2
Students playing only Cricket :
-->1
Students playing only Badminton :
-->3
Number of students playing neither cricket nor badminton :
2
```

L.2 Write C/C++ program to store marks scored for first test of subject 'Data Structures and Algorithms' for N students. Compute

i. **The average score of class**

ii. **Highest score and lowest score of class**

iii. **Marks scored by most of the students**

iv. **list of students who were absent for the test**

Solution :

```
/*   Program: Student database without using pointers.*/
/* Program details:
Various operations like read,insert,delete,print for student
database */
#include <stdio.h>
#include <conio.h>
#include <string.h>
typedef struct student
{ int rollno;
  char name[20];
  int  marks;
}student;

void print(student st[],int n);
void read(student st[],int n);
void main()
{
  student st[30];
  int n,i,op,position,rollno,high,low,present,sum;
  int highfrequency=0,highgroup;
  float avg;
  int frequency[11]; //marks obtained in each group 0 to
  9,10 to 20 ,..
  clrscr();
  printf("\nEnter No. of student:");
  scanf("%d",&n);
  read(st,n);
  for(i=0;i<=10;i++)
    frequency[i]=0;
// find highest,lowest, and sum of marks, and number of
students present
  high=0;low=9999;present=0;sum=0;
  for(i=0;i<n;i++)
  {
    if(st[i].marks!=0)
     {
        frequency[st[i].marks/10]++;
        if(frequency[st[i].marks/10] > highfrequency)
         {
           highfrequency=frequency[st[i].marks/10];
           highgroup=st[i].marks/10;
         }
        sum=sum+st[i].marks;
        present++;
        if(st[i].marks < low)
          low=st[i].marks;
        if(st[i].marks > high)
```

```
            high=st[i].marks;
        }
    }
  printf("\nAverage score of class :
%8.2f",(float)sum/present);
  printf("\nHighest score = %d\nLowest score=
%d",high,low);
  //students absent
  printf("\n students absent : ");
  print(st,n);
  printf("\nMaximum student in the group %d to
%d",highgroup*10,highgroup*10+9);
}
void print(student st[],int n)
    { int i;
      for(i=0;i<n;i++)
        if(st[i].marks==0)
          printf("\n%20s%5d",st[i].name,st[i].rollno);
    }
void read(student st[],int n)
  { int i;
      printf("\n enter data(name rollno marks(marks as 0 if
absent):");
      for(i=0;i<n;i++)

scanf("%s%d%d",st[i].name,&st[i].rollno,&st[i].marks);
  }
```

Output

```
Enter No. of student:5

 enter data(name rollno marks(marks as 0 if absent):
a1 599 0
a2 654 88
a3 876 75
a5 456 81
a6 567 0

Average score of class :   81.33
Highest score = 88
Lowest score= 75
 students absent :
          a1  599
          a6  567
Maximum student in the group 80 to 89
```

L.3 Set A=(1,3, a, s, t, i} represent alphanumeric characters permitted to be used to set the password of length 4. Write C/C++ program to generate all possible passwords.

Solution :

```
//Program for permutation
#include<stdio.h>
#include<conio.h>
void permutation( char *a,char *x,int k,int m);/*initial
value of k should be 0*/

void main()
{
  char a[7]={'1','3','a','s','t','i','\0'};
  char x[6];int m=4;
  clrscr();
  printf("\nList of Passwords: ");
  permutation(a,x,0,m);
}
void permutation( char *a,char *x,int k,int m)
{ int i;
 if(k==m)
  {x[k]='\0';
  printf("\t%s",x);
  getch();
  }
 else
 for(i=0;a[i]!='\0';i++)
   { x[k]=a[i];
    permutation(a,x,k+1,m);
   }
}
```

Output

```
List of Passwords:   1111  1113  111a  111s  111t
111i  1131
1133  113a  113s  113t  113i  11a1  11a3  11aa
11as  11at
11ai  11s1  11s3  11sa  11ss  11st  11si  11t1
11t3  11ta
11ts  11tt  11ti  11i1  11i3  11ia  11is  11it  11ii
1311
```

L.4 Write C++ program for sparse matrix realization and operations on it-Transpose, Fast Transpose and addition of two matrices

Solution :

```
/*Represent sparse matrix using array and perform matrix
addition, simple transpose and fast transpose */

#include<stdio.h>
#include<conio.h>
#define MAX 20
void printsparse(int[][3]);
```

```
void readsparse(int[][3]);
void transpose(int[][3],int[][3]);  // simple transpose
void Fast_transpose(int B1[MAX][3],int B2[MAX][3]);

void main()
{
    int b1[MAX][3],b2[MAX][3],m,n,b3[MAX][3],op;
    clrscr();
    do
    {
    printf("\n1)Read the First Sparse Matrix");
       printf("\n2)Read the second sparse matrix");
       printf("\n3)Display the first matrix");
       printf("\n4)Display the second matrix");
       printf("\n5)Simple transpose of the first matrix");
       printf("\n6)Fast transpose of the first matrix");
       printf("\n7)Quit");
       printf("\nEnter your choice : ");
       scanf("%d",&op);
       switch(op)
       {
         case 1: readsparse(b1);break;
         case 2: readsparse(b2);break;
         case 3: printsparse(b1);break;
         case 4: printsparse(b2);break;
         case 5: transpose(b1,b3);printsparse(b3);break;
         case 6:
Fast_transpose(b1,b3);printsparse(b3);break;
       }
    }while(op!=7);
}
void readsparse(int b[MAX][3])
{
    int i,t,m,n;

    printf("\n Enter the size of matrix (rows,columns)");
    scanf("%d%d",&m,&n);
    b[0][0]=m;
    b[0][1]=n;
    printf("\nEnter no. of non-zero elements:");
    scanf("%d",&t);
    b[0][2]=t;
    for(i=1;i<=t;i++)
    {
        printf("\n Enter the next triple(row,column,value)
:");
        scanf("%d%d%d",&b[i][0],&b[i][1],&b[i][2]);
    }
}
void printsparse(int b[MAX][3])
{
    int i,n;
    n=b[0][2];  //no of 3-triples
    printf("\nrows = %d\tcolumns = %d",b[0][0],b[0][1]);
    printf("\n");
    for(i=1;i<=n;i++)
        printf("%d\t%d\t%d\n",b[i][0],b[i][1],b[i][2]);
}
void transpose(int b1[][3],int b2[][3])
{
    int i,j,k,n;
    b2[0][0]=b1[0][1];
    b2[0][1]=b1[0][0];
    b2[0][2]=b1[0][2];
    k=1;
    n=b1[0][2];
    for(i=0;i<b1[0][1];i++)
        for(j=1;j<=n;j++)
    /* if a column number of current triple == i
      then insert the current triple in b2 */
        if(i== b1[j][1])
        {
            b2[k][0]=i;
            b2[k][1]=b1[j][0];
            b2[k][2]=b1[j][2];
            k++;
        }
}
void Fast_transpose(int B1[MAX][3],int B2[MAX][3])
{
    int m,n,t,i,col_num,location;
    int total[MAX],index[MAX];
    m=B1[0][0];n=B1[0][1];t=B1[0][2];
    B2[0][0]=n;B2[0][1]=m;B2[0][2]=t;
    for(i=0;i<n;i++)
        total[i]=0;
    for(i=1;i<=t;i++)
    {
        col_num=B1[i][1];
        total[col_num]++;
    }
    index[0]=1;
    for(i=1;i<n;i++)
        index[i]=index[i-1]+total[i-1];
    for(i=1;i<=t;i++)
    {
        col_num=B1[i][1];
```

```
            location=index[col_num];
            index[col_num]++;
            B2[location][0]=B1[i][1];
            B2[location][1]=B1[i][0];
            B2[location][2]=B1[i][2];
        }
    }
```

Output

```
1)Read the First Sparse Matrix
2)Read the second sparse matrix
3)Display the first matrix
4)Display the second matrix
5)Simple transpose of the first matrix
6)Fast transpose of the first matrix
7)Quit
Enter your choice : 1
Enter the size of matrix (rows,columns)2 2
Enter no. of non-zero elements:3
Enter the next triple(row,column,value) :2 3 5
Enter the next triple(row,column,value) :0 2 1
Enter the next triple(row,column,value) :2 5 8

1)Read the First Sparse Matrix
2)Read the second sparse matrix
3)Display the first matrix
4)Display the second matrix
5)Simple transpose of the first matrix
6)Fast transpose of the first matrix
7)Quit
Enter your choice : 3
rows = 2      columns = 2
    2    3    5
    0    2    1
    2    5    8

1)Read the First Sparse Matrix
2)Read the second sparse matrix
3)Display the first matrix
4)Display the second matrix
5)Simple transpose of the first matrix
6)Fast transpose of the first matrix
7)Quit
Enter your choice : 2
Enter the size of matrix (rows,columns)2 2
Enter no. of non-zero elements:3
Enter the next triple(row,column,value) :4 5 7
Enter the next triple(row,column,value) :3 5 9
Enter the next triple(row,column,value) :3 0 3

1)Read the First Sparse Matrix
2)Read the second sparse matrix
3)Display the first matrix
4)Display the second matrix
5)Simple transpose of the first matrix
6)Fast transpose of the first matrix
7)Quit
Enter your choice : 4
rows = 2      columns = 2
    4    5    7
    3    5    9
    3    0    3

1)Read the First Sparse Matrix
2)Read the second sparse matrix
3)Display the first matrix
4)Display the second matrix
5)Simple transpose of the first matrix
6)Fast transpose of the first matrix
7)Quit
Enter your choice : 5
rows = 2      columns = 2
    2    3    5
    0    2    1
    2    5    8
1)Read the First Sparse Matrix
2)Read the second sparse matrix
3)Display the first matrix
4)Display the second matrix
5)Simple transpose of the first matrix
6)Fast transpose of the first matrix
7)Quit

Enter your choice : 6
rows = 2      columns = 2
    2    0    1
    3    2    5
    2    5    8

1)Read the First Sparse Matrix
2)Read the second sparse matrix
3)Display the first matrix
4)Display the second matrix
5)Simple transpose of the first matrix
6)Fast transpose of the first matrix
7)Quit
Enter your choice : 7
```

L.5 Write C++ program for storing binary number using doubly linked lists. Write functions-

a) to compute 1's and 2's complement

b) add two binary numbers

Solution :

```
/* Program: Addition of two binary numbers represented
using adoubly linked list. */
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
typedef struct dnode
{ int data;
 struct dnode *next;
 struct dnode *prev;
}dnode;
dnode *create();
dnode *add(dnode *head1,dnode *head2);
void print(dnode *head);
void onescomp(dnode *head);
void twoscomp(dnode *head);
void main()
{dnode *head1,*head2,*head3;
 head1=create();
 head2=create();
 head3=add(head1,head2);
 printf("\nFirst  Binary Number:");
 print(head1);
 printf("\nSecond Binary Number:");
 print(head2);
 printf("\nFinal(SUM)  Binary Number:");
 print(head3);
 printf("\n1's complement of first number : ");
 onescomp(head1);
 print(head1);
 printf("\n2's complement of second number : ");
 twoscomp(head2);
 print(head2);
 getch();
}
dnode *create()
{ dnode *p,*head;
 char x;
 head=NULL;
 printf("\nEnter a Binary Number:");
 while((x=getchar())!='\n')
 { if(head==NULL)
   {head=p=(dnode*)malloc(sizeof(dnode));
    p->next=p->prev=NULL;
   }
   else
   { p->next=(dnode*)malloc(sizeof(dnode));
    p->next->prev=p;
    p=p->next;
    p->next=NULL;
   }
   if(x=='1')
   p->data=1;
   else
   p->data=0;
  }
 return(head);
}
dnode *add(dnode *head1,dnode *head2)
{ dnode *head,*p;
 int x,y,z,carry;
 carry=0;/*initial carry*/
 /*go to the least significant digit */
 head=NULL;
 if(head1!=NULL)
  while(head1->next!=NULL)
   head1=head1->next;
 if(head2!=NULL)
  while(head2->next!=NULL)
   head2=head2->next;
 while(head1 !=NULL || head2 !=NULL)
  { x=y=0;
   if(head1!=NULL)
    { x=head1->data;
     head1=head1->prev;
    }
   if(head2!=NULL)
    { y=head2->data;
     head2=head2->prev;
    }
   z=(x+y+carry)%2;
   carry=(x+y+carry)/2;
   if(head==NULL)
    {head=(dnode*)malloc(sizeof(dnode));
    head->next=head->prev=NULL;
    }
   else
    { head->prev=(dnode*)malloc(sizeof(dnode));
    head->prev->next=head;
    head=head->prev;
    head->prev=NULL;
    }
```

```
    head->data=z;
  }
 if(carry==1)
   { head->prev=(dnode*)malloc(sizeof(dnode));
     head->prev->next=head;
     head=head->prev;
     head->prev=NULL;
     head->data=carry;
   }
 return(head);
}
void print(dnode *head)
{ while(head!=NULL)
 {
  printf("%d",head->data);
  head=head->next;
 }
}
void onescomp(dnode *head)
{  dnode *p;
  p=head;
  while(p!= NULL)
   {
    if(p->data==0)
     p->data=1;
    else
     p->data=0;
    p=p->next;
    }
}
void twoscomp(dnode *head)
 {
  dnode *p;
  p=head;
  while(p->next != NULL)
   p=p->next;
  while(p!=NULL && p->data==0)
   p=p->prev;
  if(p!=NULL)
  p=p->prev;
  while(p!=NULL)
  {
   if(p->data == 0)
    p->data = 1;
   else
    p->data = 0;    p=p->prev;
  }
}
```

Output

```
Enter a Binary Number:1011

Enter a Binary Number:0111

First  Binary Number:1011
Second Binary Number:0111
Final(SUM)  Binary Number:10010
1's complement of first number : 0100
2's complement of second number : 1001
```

L.6 **Let $x = (x_1, x_2, \ldots, x_n)$ and $y = (y_1, y_2, \ldots, y_m)$ be two doubly linked lists. Assume that in each linked list, the nodes are in non-decreasing order of their data-field values. Write C/C++ program to merge the two lists to obtain a new linked list z in which the nodes are also in this order. Following the merge, x and y should represent empty lists because each node initially in x or y is now in z. No additional nodes may be used.**

Solution :

```
/*   Merging of Two Sorted, Doubly Linked Lists : */
/*   Experiment 2 : Create two doubly linked list. Sort them
after creation. Merge these two      lists into one list so that
the merged list is in sorted order(No new node should be
     created). */
#include <iostream.h>
#include <conio.h>
/*Structure of a node of a doubly linked list*/
class dnode
 { public:
     int  data;
     dnode *next,*prev;
     dnode()   //default constructor
     { next=prev=NULL;
     }
  /*Whenever a new node is acquired,default constructor
will set the
    'next' field to NULL  */
     dnode(int x)   //parameterised constructor
     {next=prev=NULL;
      data=x;
     }
  /* parameterised constructor will initialize the data field
with x */
  };
/**********Definition of the node ends **********/
/*********Definition of a Doubly linked list *********/
class linkedlist
 {   dnode *head; //to store the address of the starting node
```

```
public:
 linkedlist()
  { head=NULL;
  }
//Default constructor will set the initial value of 'head' to
NULL.
//Initially, a linked list is empty

  void create();
  void print();
  void sort();
  void merge(linkedlist &l1,linkedlist &l2);
};
void linkedlist:: create()
{ int x,n,i;
 dnode *p;
 cout << "\n Enter no of nodes : ";
 cin>>n;
 for(i=0;i<n;i++)
 { cout<<"\n Next Data : ";
  cin >> x;
  if(head==NULL) // inserting the first node
   head=p=new dnode(x);
  else
  { p->next=new dnode(x);
   p->next->prev=p;
   p=p->next;
  }
 }
}
void linkedlist::print()
{ cout<<"\n Data stored in the Doubly linked list : ";
 for(dnode *p=head; p!=NULL ; p=p->next)
 cout<<p->data<<" ";
}
void linkedlist::sort()
{ int i,j,temp;
 int n=0;
 dnode *p;
 //counting number of nodes
 for(p=head;p!=NULL;p=p->next)
  n++;
 //bubble sort is being used for sorting
 for(i=1;i<n;i++)
 {p=head;          //p ponts to jth node
 for(j=0;j<n-i;j++,p=p->next)
 if(p->data > p->next->data)
  { // interchange
   temp=p->data;
   p->data=p->next->data;
   p->next->data=temp;
  }
 }
}
void linkedlist::merge(linkedlist &l1,linkedlist &l2)
{ dnode *p; //p points to the last node of the current linked
list;
 if(l1.head==NULL) // if the first linked list is empty
 {head=l2.head;
  l2.head=NULL;
  return;
 }
 if(l2.head==NULL)  //if the second linked list is empty
 {head=l1.head;
  l1.head=NULL;
  return;
 }
 if(l1.head->data < l2.head->data)
 { head=p=l1.head;
  l1.head=l1.head->next;
 }
 else
 { head=p=l2.head;
  l2.head=l2.head->next;
 }
while(l1.head !=NULL && l2.head != NULL)
 if(l1.head->data < l2.head->data)
  { p->next=l1.head;
   l1.head->prev=p;
   l1.head=l1.head->next;
   p=p->next;
  }
 else
  { p->next=l2.head;
   l2.head->prev=p;
   l2.head=l2.head->next;
   p=p->next;
  }
if(l1.head !=NULL )
 {p->next=l1.head;
 l1.head->prev=p;
 }
if(l2.head != NULL)
 { p->next=l2.head;
  l1.head->prev=p;
 }
```

```
l1.head=l2.head=NULL;
}
/******** end of doubly linked list definition ********/
/* A program showing usage of class linkedlist */
void main()
 { int op,x,loc;
  linkedlist l,l1,l2;
  clrscr();
  cout<<"\n Creating first linked list ****\n";
  l1.create();l1.sort();
  cout<<"\n Creating second linked list ****\n";
  l2.create();l2.sort();
  l.merge(l1,l2);
  cout<<"\n Final Linked List *****\n";
  l.print();
  getch();
 }
```

Output

```
Creating first linked list ****
Enter no of nodes : 3
Next Data : 11
Next Data : 22
Next Data : 33
Creating second linked list ****
Enter no of nodes : 1 44 10
Next Data :
Final Linked List *****
Data stored in the Doubly linked list : 11 22 33 44
```

L.7 Write C++ program to realize Set using Generalized Linked List (GLL) Ex. A ={ a, b, {c, d,e, {}, {f,g}, h, I, {j,k}, l, m}. Store and print as set notation.

Solution :

```
/*  Generelized Linked List :*/
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
#include <string.h>
#include <ctype.h>
typedef struct node
 { int flag;
  union
   { char val;
     struct node *down;
   }vd;
  struct node *next;
}node;
node * create(char a[],int i,int j);
void display(node *head1);
void main()
 { node *head1=NULL;
  int op,i,j;
  char a[50],b[50];
  clrscr();
  flushall();
  printf("\nEnter lst GLL(Ex.. {a,{b,{c,d},e}} :");
  gets(b);
  //removing commas and white spaces
  i=j=0;
  while(b[i]!='\0')
    {
       if(b[i]!=',' && b[i]!=' ')
         a[j++]=b[i];
       i++;
    }
  a[j]='\0';
  j=strlen(a)-1;
  head1=create(a,1,j-1);
  printf("\n GLL : ");putchar('(');display(head1);
                          putchar(')');

}
node * create(char a[],int i,int j)
    { node *p;int k,count;
     while(a[i]=='}')
      i++;
     if(i>j)
      return(NULL);
     p=(node*)malloc(sizeof(node));
     if(isalnum(a[i]))
      { p->flag=0;
       p->vd.val=a[i];
       p->next=create(a,i+1,j);
       return(p);
      }
     if(a[i]=='{')
     {count=1;k=i;
      while(count!=0)
                  // locate the corresponding right parenthesis
       { k++;
        if(a[k]=='{')
          count++;
        if(a[k]=='}')
          count--;
       }
```

```
        p->flag=1;
        p->vd.down=create(a,i+1,k-1);
        p->next=create(a,k+1,j);
        return(p);
       }
    printf("\nWrong input:");
    return(NULL);
}
void display(node *p)
{if(p!=NULL)
 {if(p->flag==0)
   {printf("%c",p->vd.val);
    display(p->next);
   }
  else
   { printf("(");
     display(p->vd.down);
     printf(")");
     display(p->next);
   }
 }
}
```

Output

```
Enter GLL(Ex.. {a,{b,{c,d},e}} :{a,{b,c},d}

GLL : (a(bc)d)
```

L.8 Implement C++ program for expression conversion as infix to postfix and its evaluation using stack based on given conditions

i. **Operands and operator, both must be single character.**

ii. **Input Postfix expression must be in a desired format.**

iii. **Only '+', '-', '*' and '/ ' operators are expected.**

Solution :

```
/* program for conversion of:
1. infix to its postfix form
2. infix to its prefix form
3. Evaluation of postfix expression
  operators supported '+,-,*,/,%, ^ ,(,)
  operands supported -- all single character operands
*/
#include<stdio.h>
#include<conio.h>
#include<ctype.h>
#define MAX 50

typedef struct node
 {
     int data;
     struct node *next;
 }node;

int  precedence(char);
void init(node **);
int  empty(node *);
int  pop(node **);
void push(node **,int );
int  top(node *); //value of the top element
void infix_to_prefix(char infix[],char prefix[]);
void infix_to_postfix(char infix[],char postfix[]);
void eval_postfix(char postfix[]);
int  evaluate(char x,int op1,int op2);

void main()
 {
     char infix[30],postfix[30],prefix[30];
     clrscr();
     printf("\nEnter an infix expression : ");
     gets(infix);
     infix_to_postfix(infix,postfix);
     infix_to_prefix(infix,prefix);
     printf("\nPostfix : %s\nprefix: %s  ",postfix,prefix);
     printf("\nPostfix evaluation : ");
     eval_postfix(postfix);
     getch();
 }
void infix_to_prefix(char infix[],char prefix[])
 {
    int i,j;
    char temp,in1[30];
    // reverse the infix expression  and store it in in1[]
    for(i=strlen(infix)-1,j=0;i>=0;i--,j++)
        in1[j]=infix[i];
    in1[j]='\0';
    // reverse the direction of brackets
    for(i=0;in1[i]!='\0';i++)
      {
        if(in1[i]=='(')
            in1[i]=')';
        else
            if(in1[i]==')')
                in1[i]='(';
      }
    // convert from infix to postfix
```

```
    infix_to_postfix(in1,prefix);
    //reverse the final expression
    for(i=0,j=strlen(prefix)-1;i<j;i++,j--)
     {
        temp=prefix[i];
        prefix[i]=prefix[j];
        prefix[j]=temp;
     }
}
void infix_to_postfix(char infix[],char postfix[])
{
    node *head;
    char x;
    int i,j;//i-index for infix[],j-index for postfix
    char token;
    init(&head);
    j=0;
    for(i=0;infix[i]!='\0';i++)
     {
        token=infix[i];
        if(isalnum(token))
            postfix[j++]=token;
        else
            if(token == '(')
                push(&head,'(');
            else
                if(token == ')')
                    while((x=pop(&head))!='(')
                        postfix[j++]=x;
                else
                 {

    while(precedence(token)<=precedence(top(head))
&&                  !empty(head))
                  {
                     x=pop(&head);
                     postfix[j++]=x;
                  }
                  push(&head,token);
                }
    }
    while(!empty(head))
     {
        x=pop(&head);
        postfix[j++]=x;
     }
    postfix[j]='\0';
}
void eval_postfix(char postfix[])
 {
    node *head;
    char x;
    int op1,op2,val,i;
    init(&head);
    for(i=0;postfix[i]!='\0';i++)
     {    x=postfix[i];
        if(isalpha(x))
           { printf("\nEnter the value of %c : ",x);
            scanf("%d",&val);
            push(&head,val);
           }
        else
        {     //pop two operands and evaluate
            op2=pop(&head);
            op1=pop(&head);
            val=evaluate(x,op1,op2);
            push(&head,val);
        }
    }
    val=pop(&head);
    printf("\nvalue of expression = %d",val);
}
int evaluate(char x,int op1,int op2)
{
    if(x=='+')  return(op1+op2);
    if(x=='-')  return(op1-op2);
    if(x=='*')  return(op1*op2);
    if(x=='/')  return(op1/op2);
    if(x=='%')  return(op1%op2);
}
int precedence(char x)
{
    if(x == '(')                    return(0);
    if(x == '+' || x == '-')        return(1);
    if(x == '*' || x == '/' || x == '%') return(2);
    return(3);
}
void init(node **head)
{
    *head=NULL;
}
int empty(node *head)
{
    if(head==NULL)
        return(1);
    return(0);
}
void push(node **head,int x)
{
```

```
    node *p;
    p=(node*)malloc(sizeof(node));
    p->data=x;
    p->next=*head;
    *head=p;
}
int pop(node **head)
{
    int x;
    node *p;
    p=*head;
    *head=p->next;
    x=p->data;
    free(p);
    return(x);
}
int top(node  *head)
{
    return(head->data);
}
```

Output

```
Enter an infix expression : a+b*c
Postfix : abc*+
prefix: +a*bc
Postfix evaluation :
Enter the value of a : 2
Enter the value of b : 3
Enter the value of c : 4
value of expression = 14
```

L.9 Queues are frequently used in computer programming, and a typical example is the creation of a job queue by an operating system. If the operating system does not use priorities, then the jobs are processed in the order they enter the system. Write C++ program for simulating job queue. Write functions to add job and delete job from queue.

Solution :

```
/*  Program: A sample program for queue.
    Show the status of the queue after every operation.
    (a)  Insert 5 elements  (b) Delete 2 elements  */
#include<conio.h>
#include<stdio.h>
#define MAX 10
typedef struct Q
{
    int R,F;
    int data[MAX];
} Q;
void initialise(Q *P);
int empty(Q *P);
int full(Q *P);
void enqueue(Q *P,int x);
int dequeue(Q *P);
void print(Q *P);
void main()
{
    Q q;
    int x,i;
    initialise(&q);
    printf("\nEnter 5 elements :");
    for(i=1;i<=5;i++)
    {
        scanf("%d",&x);
        if(!full(&q))
            enqueue(&q,x);
        else
        {
            printf("\n Queue is full .....exitting");
            exit(0);
        }
    }
    print(&q);
    for(i=1;i<=2;i++)
    {
        if(!empty(&q))
            x=dequeue(&q);
        else
        {
            printf("\ncan not delete...Queue is empty");
            exit(0);
        }
    }
    print(&q);
}
void initialise(Q *P)
{
    P->R=-1;
    P->F=-1;
}
int empty(Q *P)
{
    if(P->R==-1)
        return(1);
    return(0);
}
int full(Q *P)
```

```
{
    if(P->R==MAX-1)
        return(1);
    return(0);
}
void enqueue(Q *P,int x)
{
    if(P->R==-1)
    {
        P->R=P->F=0;
        P->data[P->R]=x;
    }
    else
    {
        P->R=P->R+1;
        P->data[P->R]=x;
    }
}
int dequeue(Q *P)
{
    int x;
    x=P->data[P->F];
    if(P->R==P->F)
    {
        P->R=-1;
        P->F=-1;
    }
    else
        P->F=P->F+1;
    return(x);
}
void print(Q *P)
{
    int i;
    if(!empty(P))
    {
        printf("\n");
        for(i=P->F;i<=P->R;i++)
            printf("%d\t",P->data[i]);
    }
}
```

Output

```
Enter 5 elements :
5   4   3   2   1
5   4   3   2   1
3   2   1
```

L.10 A double-ended queue(deque) is a linear list in which additions and deletions may be made at either end. Obtain a data representation mapping a deque into a one-dimensional array. Write C++ program to simulate deque with functions to add and delete elements from either end of the deque.

Solution :

```
#include <stdio.h>
#include <conio.h>
#define MAX 30
typedef struct dequeue
{   int data [MAX];
    int rear, front;
} dequeue;
void initialize (dequeue *p);
int empty (dequeue *p);
int full (dequeue *p);
void enqueueR (dequeue *p, int x);
void enqueueF (dequeue *p, int x);
int dequeueF (dequeue *p);
int dequeueR (dequeue *p);
void print (dequeue *p);
void main( )
{
    int x, op, n,i;
    dequeue q;
    initialize (&q);
    do
    {
    printf("\n1)create\n2)insert(rear)\n3)insert(front)\n4)D
elete(rear)\n5)Delete(front)");
    printf("\n6)print\n7)Quit");
    printf("\n enter your choice :");
    scanf("%d", &op);
    switch (op)
    {
        case 1 : printf("\n enter no. of elements :");
                 scanf("%d", &n);
                 initialize(&q);
                 printf("\n enter the data :");
                 for (i=0; i< n; i++)
        {
            scanf("%d", &x);
            if (full(&q))
            {
                printf("\n queue is full ");
                exit(0);
            }
```

```
            enqueueR(&q, x);
        }
        break;
        case 2 : printf("\n enter element to be inserted:");
                 scanf("%d", &x);
                 if (full (&q))
        {
            printf("\n queue is full ");
            exit(0);
        }
        enqueueR(&q, x);
        break;
        case 3 : printf ("\n enter the element to be
                 inserted :");
                 scanf("%d", &x);
                 if (full (&q))
        {
            printf("\n queue is full ");
            exit(0);
        }
        enqueueF(&q, x);
        break;
        case 4 : if (empty(&q))
        {
            printf("\n queue is empty ");
            exit(0);
        }
        x = dequeueR(&q);
        printf("\n element = %d", x);
            break;
        case 5 : if (empty (&q))
        {
            printf("\n queue is empty ");
            exit(0);
        }
        x = dequeueF(&q);
        printf("\n element=%d" , x);
        break;
        case 6 : print(&q); break;
        default : break;
        }
    }while(op!= 7);
}
void initialize(dequeue *P)
{
    P -> rear = -1;
    P -> front = -1;
}
int empty(dequeue *P)
{
    if(P -> rear == -1)
        return(1);
    return(0);
}
int full(dequeue *P)
{
    if((P->rear + 1)%MAX == P->front)
        return(1);
    return(0);
}
void enqueueR(dequeue *P, int x)
{
    if(empty(P))
    {
        P -> rear = 0;
        P -> front = 0;
        P -> data[0] = x;
    }
    else
    {
        P -> rear =(P->rear + 1) % MAX;
        P -> data[P->rear] = x;
    }
}
void enqueueF(dequeue *P, int x)
{
    if(empty(P))
    {
        P -> rear = 0;
        P -> front = 0;
        P -> data[0] = x;
    }
    else
    {
        P -> front = (P -> front -1 + MAX) % MAX;
        P -> data[P -> front] = x;
    }
}
int dequeueF(dequeue *P)
{
    int x;
    x = P -> data[P -> front];
    if(P -> rear == P -> front)
    /* delete the last element */
        initialize(P);
    else
```

```
        P -> front =(P -> front + 1) % MAX;
    return(x);
}
int dequeueR(dequeue *P)

{
    int x;
    x = P -> data[P -> rear];
    if(P->rear == P -> front)
        initialize(P);
    else
        P->rear =(P->rear - 1 + MAX) % MAX;
    return(x);
}
void print(dequeue *p)
{   int i;
    i = p->front;
    printf("\n");
    while (i != p -> rear)
    {   printf("%d\t", p -> data[i]);
        i = (i + 1)% MAX;
    }
printf("%d\t", p -> data [p -> rear]);
}
```

Output

```
enter your choice :1

enter no. of elements :5

enter the data :1
2
3
4
5

1)create
2)insert(rear)
3)insert(front)
4)Delete(rear)
5)Delete(front)
6)print
7)Quit
enter your choice :6

1   2   3   4   5
1)create
2)insert(rear)
3)insert(front)
4)Delete(rear)
5)Delete(front)
6)print
7)Quit
enter your choice :2

enter element to be inserted:11

1)create
2)insert(rear)
3)insert(front)
4)Delete(rear)
5)Delete(front)
6)print
7)Quit
enter your choice :6

1   2   3   4   5   11
1)create
2)insert(rear)
3)insert(front)
4)Delete(rear)
5)Delete(front)
6)print
7)Quit
enter your choice :7
```

L.11 Write C++ program to store first year percent of students in array. Write function for sor array of floating point numbers in ascen order using

a) Selection Sort

b) Bubble sort and display top five score

Solution :

```
#include<conio.h>
#include<stdio.h>
void selection(float [],int);
void bubble(float [], int);
void main()
{
    int n,i,op;
    float a[30];
    clrscr();
    do
    {
        printf("\n1)Bubble sort\n2)Selection
sort\n3)Quit");
        printf("\nEnter your choice : ");
        scanf("%d",&op);
```

```
        if(op==1)
        {
            printf("\nEnter no of elements :");
            scanf("%d",&n);
            printf("\nEnter array elements :");
            for(i=0;i<n;i++)
                scanf("%f",&a[i]);
            bubble(a,n);
            printf("\nSorted array is :");
            for(i=n-1;i>=n-5;i--)
                printf("%7.2f ",a[i]);
        }
        if(op==2)
        {
            printf("\nEnter no of elements :");
            scanf("%d",&n);
            printf("\nEnter array elements :");
            for(i=0;i<n;i++)
                scanf("%f",&a[i]);
            selection(a,n);
            printf("\nSorted array is :");
            for(i=n-1;i>=n-5;i--)
                printf("%7.2f ",a[i]);
        }
    }while(op!=3);
}
void bubble(float a[],int n)
{
    int i,j;
    float temp;
    for(i=1;i<n;i++)
        for(j=0;j<n-i;j++)
            if(a[j]>a[j+1])
            {
                temp=a[j];
                a[j]=a[j+1];
                a[j+1]=temp;
            }
}
void selection(float a[], int n)
    {   int i, j, k;
        float temp;
        for(i = 0; i < n - 1; i ++)
            {
                k = i ;
            /* ith element is assumed to be the smallest */
                for(j = i + 1; j < n; j ++)
                    if(a[j] < a[k])
                        k = j;
                if(k != i)
                {
                    temp = a[i];
                    a[i] = a[k];
                    a[k] = temp;
                }
            }
    }
```

Output

```
1)Bubble sort
2)Selection sort
3)Quit
Enter your choice : 1

Enter no of elements :7

Enter array elements :5.6 7.9 21 2 55.67 1.91 23.89

Sorted array is :  55.67   23.89   21.00    7.90    5.60
1)Bubble sort
2)Selection sort
3)Quit
Enter your choice : 2

Enter no of elements :7

Enter array elements :5.6 7.9 21 2 55.67 1.91 23.89

Sorted array is :  55.67   23.89   21.00    7.90    5.60
1)Bubble sort
2)Selection sort
3)Quit
Enter your choice :
```

L.12 Write C++ program to store second year percentage of students in array. Write function for sorting array of floating point numbers in ascending order using

a) Insertion sort

b) Shell Sort and display top five scores.

Solution :

```
#include<conio.h>
#include<stdio.h>
void insertion(float [],int);
void shell(float [], int);
void main()
{
    int n,i,op;
```

```
    float a[30];
    clrscr();
    do
    {
        printf("\n1)Insertion sort\n2)Shell sort\n3)Quit");
        printf("\nEnter your choice : ");
        scanf("%d",&op);
        if(op==1)
        {
            printf("\nEnter no of elements :");
            scanf("%d",&n);
            printf("\nEnter array elements :");
            for(i=0;i<n;i++)
                scanf("%f",&a[i]);
            insertion(a,n);
            printf("\nSorted array is :");
            for(i=n-1;i>=n-5;i--)
                printf("%7.2f ",a[i]);
        }
        if(op==2)
        {
            printf("\nEnter no of elements :");
            scanf("%d",&n);
            printf("\nEnter array elements :");
            for(i=0;i<n;i++)
                scanf("%f",&a[i]);
            shell(a,n);
            printf("\nSorted array is :");
            for(i=n-1;i>=n-5;i--)
                printf("%7.2f ",a[i]);
        }
    }while(op!=3);
}
void shell(float a[ ], int n)
{
    int i, j, step;
    float temp;
    for(step = n/2; step > 0; step = step/2)
    for(i = step; i < n; i++)
    {
        temp = a[i];
        for(j = i; j >= step; j = j - step)
            if(temp < a[j - step])
                a[j] = a[j - step];
            else
                break;
            a[j] = temp;
        }
}
    void insertion(float a[],int n)
{
    int i,j;
    float temp;
    for(i=1;i<n;i++)
    {
        temp=a[i];
        for( j=i-1;j>=0 && a[ j]>temp;j--)
            a[ j+1]=a[ j];
        a[ j+1]=temp;
    }
}
```

Output

```
1)Insertion sort
2)Shell sort
3)Quit
Enter your choice : 1

Enter no of elements :7

Enter array elements :34.6 11.2 6.7 45.78 23.99 87.91
11.54

Sorted array is :  87.91   45.78   34.60   23.99   11.54
1)Insertion sort
2)Shell sort
3)Quit
Enter your choice : 2

Enter no of elements :7

Enter array elements :34.6 11.2 6.7 45.78 23.99 87.91
11.54

Sorted array is :  87.91   45.78   34.60   23.99   11.54
1)Insertion sort
2)Shell sort
3)Quit
Enter your choice :
```

L.13 Write C++ program to store first year percentage of students in array. Sort array of floating point numbers in ascending order using quick sort and display top five scores.

Solution :

```
#include<conio.h>
#include<stdio.h>
void quick_sort(float [],int,int);
int partition(float [],int,int);
void main()
```

```
{
  int n,i,op;
  float a[30];
  clrscr();
  do
  {
      printf("\n1)Quick Sort\n2)Quit");
      printf("\nEnter your choice : ");
      scanf("%d",&op);
      if(op==1)
      {
          printf("\nEnter no of elements :");
          scanf("%d",&n);
          printf("\nEnter array elements :");
          for(i=0;i<n;i++)
              scanf("%f",&a[i]);
          quick_sort(a,0,n-1);
          printf("\nSorted array is :");
          for(i=n-1;i>=n-5;i--)
              printf("%7.2f ",a[i]);
      }
  }while(op!=2);
}
void quick_sort(float a[],int l,int u){
  int j;
  if(l<u)
  {
      j=partition(a,l,u);
      quick_sort(a,l,j-1);
      quick_sort(a,j+1,u);
  }
}
int partition(float a[],int l,int u)
{
  int i,j;
  float v,temp;
  v=a[l];
  i=l;
  j=u+1;
  do
  {
    do
    {   i++;
    }while(a[i]<v && i<=u);
    do
    {   j--;
    }while(a[j]>v);
    if(i<j)
    {
        temp=a[i];
        a[i]=a[j];
        a[j]=temp;
    }
  }while(i<j);
  a[l]=a[j];
  a[j]=v;
  //Intermediate result with pivot
  printf("\nPartition Point=%d",j);
  printf("  List : ");
  for(i=l;i<=u;i++)
    printf("%7.2f ",a[i]);
  return(j);
}
```

Output

```
1)Quick Sort

Enter your choice : 1

Enter no of elements :8

Enter array elements :1.5 99.67 12.7 6.54 23.78 11.87
87.0 5

Partition Point=0  List : 1.50  99.67  12.70  6.54  23.78
11.87  87.00  5.00
Partition Point=7  List :  5.00  12.70  6.54  23.78
11.87  87.00  99.67
Partition Point=1  List :  5.00  12.70  6.54  23.78
11.87  87.00
Partition Point=4  List :  11.87  6.54  12.70  23.78
87.00
Partition Point=3  List :  6.54  11.87
Partition Point=5  List :  23.78  87.00
Sorted array is :  99.67  87.00  23.78  12.70  11.87
1)Quick Sort
2)Quit
Enter your choice :2
```