Laboratory Manual

For

DISCRETE MATHEMATICS

(IT 304)

B.Tech (IT)

SEM III

June 2010

Faculty of Technology
Dharmsinh Desai University
Nadiad.
www.ddu.ac.in

LIST OF EXPERIMENTS

1. Write a program to find intersection set of two sets.
2. Write a program to find union set of two sets.
3. Write a program to find Compliment set of the given set.
4. Write a program to find difference set of two sets.
5. Write a program to find symmetric difference set of two sets.
6. Write a program to prove the D'Morgan's Laws of set theory.
7. Write a program to find the power set of the given set.
8. Write a program to find permutation of the set.
9. Write a program to implement Binary Search.
10. Find the cardinality of the set and prove
    $|A \cup B \cup C| = |A| + |B| + |C| - |A \cap B| - |B \cap C| - |A \cap C| + |A \cap B \cap C|$

LABWORK BEYOND CURRICULA

1. Write a program to generate Pascal's Triangle.
2. Implement Bubblesort Algorithm and Largest2 Algorithm.

Sample Experiment

1 AIM: Write a program to implement DMorgan's Law.

2 TOOLS: Turbo c++

3 STANDARD PROCEDURES:

3.1 Analyzing the Problem:

To Implement DMorgan's law we have to verify the equation of DMorgan's Law (AUB)'=A'∩B'.

Step1

Suppose Taking U= {1, 2, 3, 4, 5}

A= {1, 3}

B= {1, 4, 6}

Step2

Union of A and B AUB= {1, 3, 4, 6}

Step3

Complement of AUB is (AUB)' = {2, 5}

Complement of A is A'= {2, 4, 5}

Complement of B is B'= {2, 3, 5}

Step4

Intersection of A' and B' A'∩B'= {2, 5}

3.2 Designing the Solution:

To design the solution of Dmorgan's law we need to implement following three main Functions

1. **Union**

   x=No. of element in set A

   y=No. of element in set B

   O[]=output set

```
int union(int x,int y,int m[],int n[],int o[])
{
            for(int j=0;j<y;j++)
          {
        for(int i=0;i<x;i++)
            {
       if(n[j]==m[i])
       {
            break;
       }
```

```
                  else
                         o[ptr++]=n[j]; /*add element to union set*/
                   /* return union sets*/

         }
    }
```

2. **Intersection**

```
int intersection(int x,int y,int m[],int n[],int o[])
  {
for(int j=0;j<y;j++)
    {
       for(int i=0;i<x;i++)
       {
              if(m[i]==n[j])
              o[ptr++]=n[j]; /* add element to intersection set.*/
          /* return intersection sets*/
       }
    }
}
```

3. **Compliment**

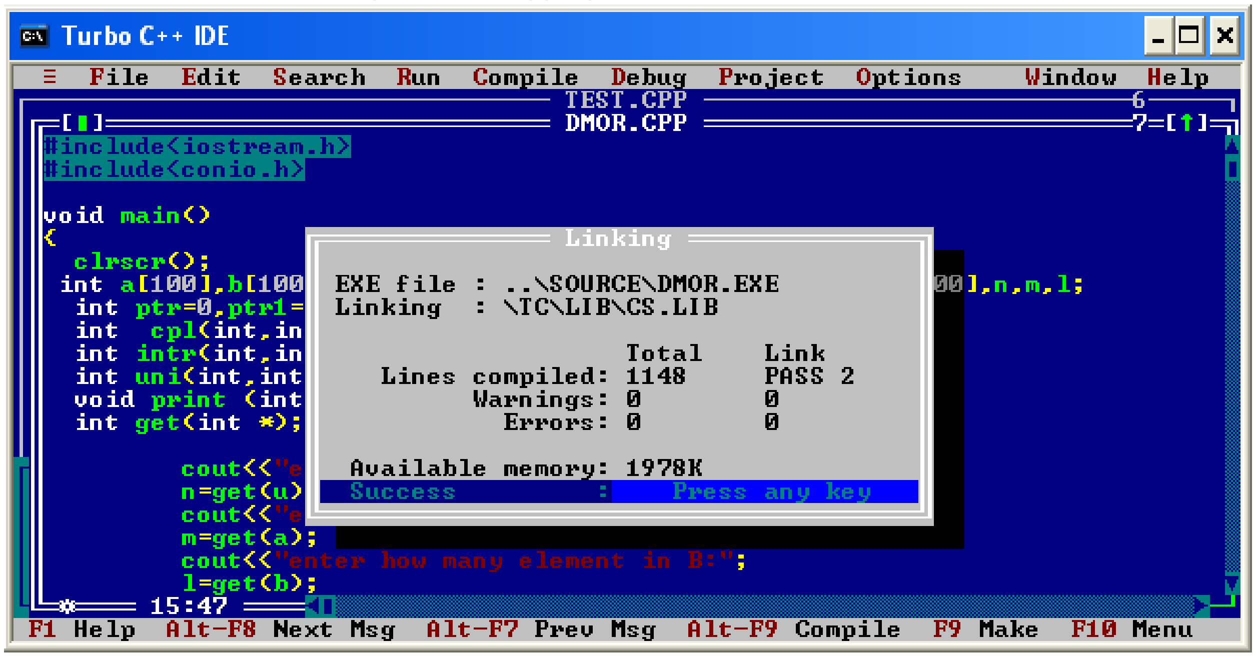
```
int compliment(int x,int y,int m[],int n[],int  o[])
{
                       for(int j=0;j<x;j++)
                   {
               for(int i=0;i<y;i++)
                      {
                    if(n[i]==m[j])
                     {

                          break;
                     }
                     else
                          o[ptr++]=m[j]; /*add element to compliment set*/
                          /* return compliment of set*/
                            }
                        }
             }
```

3.3 Implementing the Solution

3.3.1 Writing Source Code

```
#include<iostream.h>
#include<conio.h>

void main()
{
  clrscr();
 int a[100],b[100],u[100], c[100],d[100],e[100],g[100],f[100],n,m,l;
  int ptr=0,ptr1=0,ptr2=0,ptr4=0,ptr3=0;
  int  cpl(int,int,int *,int *,int *);
  int intr(int,int,int *,int *,int * );
  int uni(int,int,int *,int *,int * );
  void print (int *,int);
  int get(int *);
 cout<<"enter how many element in U:";
            n=get(u);
            cout<<"enter how many element in A:";
            m=get(a);
            cout<<"enter how many element in B:";
            l=get(b);

   cout<<"\n\n"<<"Union of A & B is: ";
   ptr= uni (m,l,a,b,c);
   print(c,ptr);

   cout<<"\n\n"<<"complement of A union B is: ";
   ptr4= cpl(n,ptr,u,c,g);
   print(g,ptr4);

   cout<<"\n\n"<<"complement of A is: ";
   ptr1=cpl(n,m,u,a,d);
   print(d,ptr1);

   cout<<"\n\n"<<"complement of B is: ";
   ptr2=cpl(n,l,u,b,e);
   print(e,ptr2);
   cout<<"\n\n"<<"intersection of A'& B' is: ";
   ptr3=intr(ptr1,ptr2,d,e,f);
   print(f,ptr3);

            getch();
}

int cpl(int x,int y,int m[],int n[],int  o[]) /* function which perform complement of two sets*/
{
            int flag=1,ptr=0;

            for(int j=0;j<x;j++)
            {
                 for(int i=0;i<y;i++)
                 {
                             if(n[i]==m[j])
                              {
                                    flag=0;
                                    break;
                              }
                              else
                                    flag=1;
                  }

                if(flag==1)
                  o[ptr++]=m[j];

            }
             return ptr;
}

int intr(int x,int y,int m[],int n[],int o[]) /* function which perform intersection of two sets*/
{
              int ptr=0;
             for(int j=0;j<y;j++)
             {
                for(int i=0;i<x;i++)
                {
                        if(m[i]==n[j])
                          o[ptr++]=n[j];
                }
            }
            return ptr;

}

int uni(int x,int y,int m[],int n[],int o[]) /* function which perform union of two sets*/
{
    int flag=1,ptr=0;
    for(int j=0;j<x;j++)
            o[ptr++]=m[j];

   for( j=0;j<y;j++)
           {
                for(int i=0;i<x;i++)
                {
                       if(n[j]==m[i])
                       {
                                   flag=0;
                                   break;
                       }
                       else
                       flag=1;

                }

                if(flag==1)
                       o[ptr++]=n[j];
           }
       return ptr;
}


void print(int o[],int ptr) /* function for printing output*/
{
           cout<<"{ ";
    for(int i=0;i<ptr;i++)
           cout<<o[i]<<" ";
            cout<<"}";
}


int get(int o[])   / * function for taking input*/

{
           int n;

           cin>>n;

           cout<<"enter element:";

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

           cin>>o[i];
           return n;

}
```

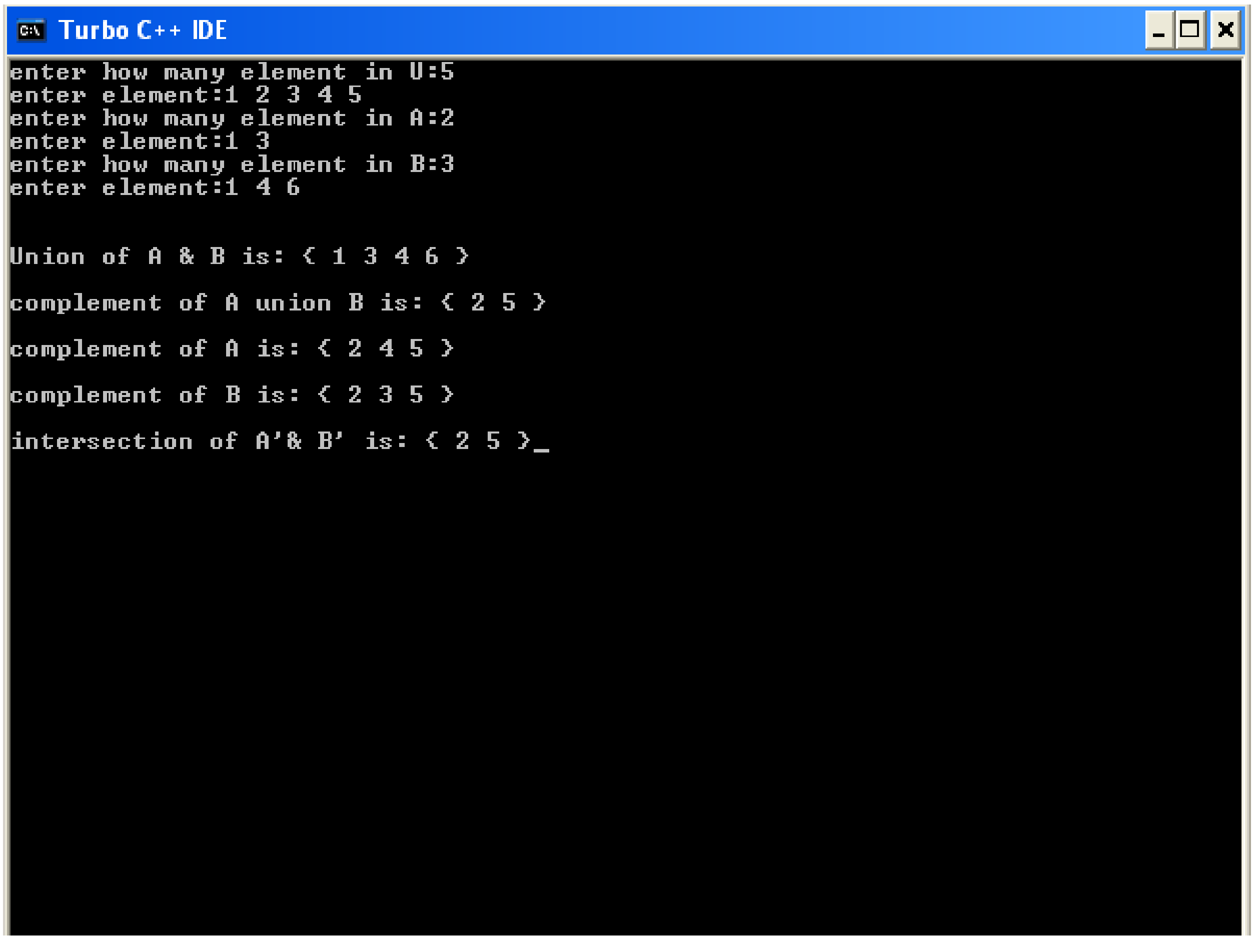
3.3.2 Compilation /Running and Debugging the Solution


Turbo C++ IDE
≡ File Edit Search Run Compile Debug Project Options Window Help
TEST.CPP
DMOR.CPP
15:47
F1 Help Alt-F8 Next Msg Alt-F7 Prev Msg Alt-F9 Compile F9 Make F10 Menu

```
#include<iostream.h>
#include<conio.h>

void main()
{
  clrscr();
 int a[100],b[100                                  00],n,m,l;
  int ptr=0,ptr1=
  int  cpl(int,in
  int intr(int,in
  int uni(int,int
  void print (int
  int get(int *);

        cout<<"e
        n=get(u)
        cout<<"e
        m=get(a);
        cout<<"enter how many element in B:";
        l=get(b);
```

```
                 Linking

 EXE file  : ..\SOURCE\DMOR.EXE
 Linking   : \TC\LIB\CS.LIB

                     Total     Link
    Lines compiled:  1148      PASS 2
          Warnings:  0         0
            Errors:  0         0

  Available memory:  1978K
  Success          :      Press any key
```

3.4 Testing the Solution



```
enter how many element in U:5
enter element:1 2 3 4 5
enter how many element in A:2
enter element:1 3
enter how many element in B:3
enter element:1 4 6


Union of A & B is: { 1 3 4 6 }

complement of A union B is: { 2 5 }

complement of A is: { 2 4 5 }

complement of B is: { 2 3 5 }

intersection of A'& B' is: { 2 5 }_
```

4 Conclusions

So Compliments of union of two sets is equal to compliment of intersection of two sets is a Dmorgan’s law

EXPERIMENT-1

Aim: Write a program to find intersection set of two sets.

Tools / Apparatus:

- O.S.: Microsoft Windows (any) / Linux / DOS
- Packages: Turbo/Borland/GNU - C/C++

Data Structure: Array, Linked lists

Programming Techniques:

- Using Pointers and passing the compared values to particular string.
- Using the concept of recursion, to continuously check the equality of elements in the array.
- By transferring the elements of the one array (set) to the other and comparing them.
- By checking the redundancy in the data by combining them into one array and later transferring them to another array.

Overview:

In discrete mathematical (set theory) terminology intersection means, the intersection set of two sets is the set, which contains the common elements that exist in both sets.

For Example,

A = {1, 2, 3, 4, 5}   B = {3, 4, 5, 6}

Then intersection set of A and B is,

A^B = Common elements of A and B = {3, 4, 5}

EXPERIMENT-2

Aim: Write a program to find union set of two sets.

Tools / Apparatus:

- O.S.: Microsoft Windows (any) / Linux / DOS
- Packages: Turbo/Borland/GNU - C/C++

Data Structure: Array, Linked lists

Programming Techniques:

- Using Pointers and passing the compared values to particular string.
- Using the concept of recursion, to continuously check the equality of elements in the array.
- By transferring the elements of the one array (set) to the other and comparing them.

Overview:

In discrete mathematical (set theory) terminology union means, the union set of two sets is the set, which contains the elements that are in either of set.

For Example,

A = {1, 2, 3, 4, 5}   B = {3, 4, 5, 6}

Then union set of A and B is,

AUB = elements either in A or B = {1, 2, 3, 4, 5, 6}

EXPERIMENT-3

Aim: Write a program to find Complement set of the given set.

Tools / Apparatus:

- O.S.: Microsoft Windows (any) / Linux / DOS
- Packages: Turbo/Borland/GNU - C/C++

Data Structure: Array, Linked lists

Programming Techniques:

- Using Pointers and passing the compared values to particular string.
- Using the concept of recursion, to continuously check the equality of elements in the array.
- By transferring the elements of the one array (set) to the other and comparing them.
- By checking the redundancy in the data by combining them into one array and later transferring them to another array.

Overview:

In discrete mathematical (set theory) terminology ‘compliment’ means, the compliment set of the given set is the set which contains the elements which are in the universal set but not present in the given set.

For Example,

U = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10}   A = {3, 4, 5, 6}

Then compliment set of A,

A’ = elements, which are not in A but present in Universal set of A.

= {1, 2, 7, 8, 9, 10}

EXPERIMENT-4

Aim: Write a program to find difference set of two sets.

Tools / Apparatus:

- O.S.: Microsoft Windows (any) / Linux / DOS
- Packages: Turbo/Borland/GNU - C/C++

Data Structure: Array, Linked lists

Programming Techniques:

- Using Pointers and passing the compared values to particular string.
- Using the concept of recursion, to continuously check the equality of elements in the array.
- By transferring the elements of the one array (set) to the other and comparing them.
- By checking the redundancy in the data by combining them into one array and later transferring them to another array.

Overview:

In discrete mathematical (set theory) terminology difference means, the difference set of two sets (say A and B) is the set which contains the elements which are in the set A but not in A^B.

For Example,

A = {1, 2, 3, 4, 5} B = {3, 4, 5, 6}

Then difference set of A and B is,

A - B = elements in A but not in A^B = {1, 2}

EXPERIMENT-5

Aim: Write a program to find symmetric difference set of two sets.

Tools / Apparatus:

- O.S.: Microsoft Windows (any) / Linux / DOS
- Packages: Turbo/Borland/GNU - C/C++

Data Structure: Array, Linked lists

Programming Techniques:

- Using Pointers and passing the compared values to particular string.
- Using the concept of recursion, to continuously check the equality of elements in the array.
- By transferring the elements of the one array (set) to the other and comparing them.
- By checking the redundancy in the data by combining them into one array and later transferring them to another array.

Overview:

In discrete mathematical (set theory) terminology symmetric difference means, the symmetric difference set of two sets (say A and B) is the set which contains the elements which are in the set A and in set B but not in A^B.

For Example,

A = {1, 2, 3, 4, 5} B = {3, 4, 5, 6}

Then symmetric difference set of A and B is,

A Δ B = elements in A and elements in B but not in A^B = {1, 2, 6}

EXPERIMENT-6

Aim: Write a program to prove the D'Morgan's Laws of set theory.

Tools / Apparatus:

- O.S.: Microsoft Windows (any) / Linux / DOS
- Packages: Turbo/Borland/GNU - C/C++

Data Structure: Array, Linked lists

Programming Techniques:

- Use the prepared functions those must have been made in earlier experiments and should use them accordingly when needed to prove the theorems. (i.e. union(…), intersection(…), compliment(…))
- Storing the value/outcome of LHS in one string and comparing that with another string, which contains the value/outcome of RHS.
- Using Pointers and passing the compared values to particular string.
- Using the concept of recursion, to continuously check the equality of elements in the array.
- By transferring the elements of the one array (set) to the other and comparing them.
- By checking the redundancy in the data by combining them into one array and later transferring them to another array.

Overview:

In discrete mathematical (set theory) terminology the D'Morgan's Laws are given below (for set A and B),

(A^B)' = A' U B' ----------------------- (1)

(AUB)' = A' ^ B' ----------------------- (2)

Equations (1) and (2) describe the D'Morgan's Laws for the set theory.

The programmer should check the validity of these two equations by using the programming techniques given above.

For Example,

A = {1, 2, 3, 4, 5} B = {3, 4, 5, 6, 7} U = {1, 2, 3, 4, 5, 6, 7, 8, 9}

A^B = {3, 4, 5} (A^B)' = {1, 2, 6, 7, 8, 9} -------------------------------------- (1)

A' = {6, 7, 8, 9} B' = {1, 2, 8, 9} A'UB' = {1, 2, 6, 7, 8, 9} – (2)

From (1) and (2) the first law is proved.

EXPERIMENT-7

Aim: Write a program to find the power set of the given set.

Tools / Apparatus:

- O.S.: Microsoft Windows (any) / Linux / DOS
- Packages: Turbo/Borland/GNU - C/C++

Data Structure: Array, Linked lists

Programming Techniques:

- Using Pointers and passing the compared values to particular string.
- Combining each value with the all other value respective of the concept of the Power set.
- Make one feature to tackle the power set of the 'Empty set'.
- Using the concept of recursion, to continuously check the equality of elements in the array.
- By transferring the elements of the one array (set) to the other and comparing them.
- By checking the redundancy in the data by combining them into one array and later transferring them to another array.
- Print the power set as a single set whose elements are also sets.

Overview:

In discrete mathematical (set theory) terminology the 'Power set' means, the set of all subsets of a set (say A) is called subset (of A).

For Example,

A = {1, 2, 3, 4, 5}

Power set of A = P(A) = { { }, {1}, {2}, {3}, {4}, {5}, {1,2}, {2,3}, ……..}

EXPERIMENT-8

Aim: Write a program to find permutation of the set.

Tools / Apparatus:

- O.S.: Microsoft Windows (any) / Linux / DOS
- Packages: Turbo/Borland/GNU - C/C++

Data Structure: Array, Linked lists

Programming Techniques:

- Use the concept of subset and use them according to different combination and also by considering the avoidance of duplicate data in terms of its subsets.
- Also by using the concept of factorial for sets.
- Using Pointers and passing the compared values to particular string.
- Using the concept of recursion, to continuously check the equality of elements in the array.
- By transferring the elements of the one array (set) to the other and comparing them.
- By checking the redundancy in the data by combining them into one array and later transferring them to another array.

Overview:

In discrete mathematical (set theory) terminology the permutation means, the power set of a particular given set contains the collection of sets of different permuted subsets and their combinations.

EXPERIMENT-9

Aim: Write a program to implement Binary Search.

Tools / Apparatus:

- O.S.: Microsoft Windows (any) / Linux / DOS
- Packages: Turbo/Borland/GNU - C/C++

Data Structure: Array, Linked lists

Programming Techniques:

- By using the robust algorithm of binary search and then implement it for the sets.
- Check whether the data is in sorted order or not and if not then sort those data.
- Use the concepts of low, high and mid design your algorithm and them implement it with any of technique described below.
- Using Pointers and passing the compared values to particular string.
- Using the concept of recursion, to continuously check the equality of elements in the array.

Overview:

In discrete mathematical (set theory) terminology the permutation means, the power set of a particular given set contains the collection of sets of different permuted subsets and their combinations.

EXPERIMENT-10

Aim: Find the cardinality of the set and prove
$|A \cup B \cup C| = |A| + |B| + |C| - |A \cap B| - |B \cap C| - |A \cap C| + |A \cap B \cap C|$

Tools / Apparatus:

- O.S.: Microsoft Windows (any) / Linux / DOS
- Packages: Turbo/Borland/GNU - C/C++

Data Structure: Array, Linked lists

Programming Techniques:

- Using Pointers and passing the compared values to particular string.
- Using the concept of recursion, to continuously check the equality of elements in the array.
- By checking the redundancy in the data by combining them into one array and later transferring them to another array.

Overview:

In discrete mathematical (set theory) terminology union means, the union set of two sets is the set, which contains the elements that are in either of set.

For Example,

A = {1, 2, 3, 4, 5} B = {3, 4, 5, 6} C= {4, 7, 8};

Then union set of A and B is,

|AUBUC| = elements either in A or B = {1, 2, 4, 6,7,8}

EXPERIMENT-11

Aim: Write a program to generate Pascal's Triangle.

Tools / Apparatus:

- O.S.: Microsoft Windows (any) / Linux / DOS
- Packages: Turbo/Borland/GNU - C/C++

Data Structure: Array, Linked lists

Programming Techniques:

- Using Pointers and passing the compared values to particular string.
- Using the concept of recursion, to continuously check the equality of elements in the array.
- By checking the redundancy in the data by combining them into one array and later transferring them to another array.

Overview:

For given elements limit, draw the Pascal's triangle as

```
            1

        1       1
        1   2   1
    1   3   3   1
```

EXPERIMENT-12

Aim: Implement Bubble sort Algorithm and Largest2 Algorithm.

Tools / Apparatus:
- O.S.: Microsoft Windows (any) / Linux / DOS
- Packages: Turbo/Borland/GNU - C/C++

Data Structure: Array, Linked lists

Programming Techniques:

- Using Pointers and passing the compared values to particular string.
- Using the concept of recursion, to continuously check the equality of elements in the array.
- By checking the redundancy in the data by combining them into one array and later transferring them to another array.

Overview:

- Do the following for i=0, 1, 2 …n-1..Compare the numbers in registers xi and xi+1. Place the larger of the two numbers in register xi+1 and the smaller in register xi.

- Finally, the number in register xn is the largest of the n numbers.