Name :Fardeen Alam

rollno. : 1900270130070 sec: IT-2

DSTL LAB PROGRAMS

1. Write a program in C to create two sets and perform the Union operation on sets.

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

int main()

{

   int a[50],b[50],m,n,i,j;

   int c[100],k=0,x=0;

   int ch;

   printf("Enter the number of elements in first set:\n");

   scanf("%d",&m);

   printf("Enter the elements:\n");

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

   {

      scanf("%d",&a[i]);

   }

   printf("\nEnter the number of elements in second set:\n");

   scanf("%d",&n);

   printf("Enter the elements:\n");
```

Name :Fardeen Alam
rollno. : 1900270130070 sec: IT-2

```
for(i=0;i<n;i++)
{
    scanf("%d",&b[i]);
}
for(i=0;i<m;i++)
{
    c[k]=a[i];
    k++;
}
for(i=0;i<n;i++)
{
    x=0;
    for(j=0;j<m;j++)
    {
        if(b[i]==c[j])
        {
            x=1;
            break;
        }
    }
```

```
    if(x==0)
        {
            c[k]=b[i];
            k++;
        }
    }
    printf("\nElement of resultant union set is\n");
    for(i=0;i<k;i++)
    {
        printf("%d\t",c[i]);
    }
}
```

Name :Fardeen Alam

rollno. : 1900270130070 sec: IT-2

```
Enter the number of elements in first set:
5
Enter the elements:
2
3
4
5
6

Enter the number of elements in second set:
5
Enter the elements:
5
7
6
9
8

Element of resultant union set is
2       3       4       5       6       7       9       8
--------------------------------
Process exited after 33.35 seconds with return value 8
Press any key to continue . . .
```

2. Write a program in C to create two sets and perform the Intersection operation on sets.

```
#include <stdio.h>

#include<conio.h>

int main()

{

int a[10], b[10], x = 0, n1, n2, i, j;

printf("Enter array1 size : ");

scanf("%d",&n1);

printf("\nEnter array2 size : ");

scanf("%d",&n2);

printf("\nEnter array1 element : ");

for(i = 0;i < n1;i++)

scanf("%d",&a[i]);

printf("\nEnter array2 element : ");

for(i = 0;i < n2;i++)
```

Name :Fardeen Alam
rollno. : 1900270130070 sec: IT-2

```
scanf("%d",&b[i]);

printf("Intersection: ");

for(i = 0;i < n1;i++)

{

for(j = 0;j < n2;j++)

 {

if(b[i] == a[j])

        {

x = 1;

        }

 }

if(x == 1)

                {

printf("%d ", b[i]);

                }

x = 0;

}

return 0;

}
```

Name :Fardeen Alam

rollno. : 1900270130070 sec: IT-2

```
Enter array1 size : 5

Enter array2 size : 6

Enter array1 element : 1
2
3
4
5

Enter array2 element : 4
5
6
7
8
9
Intersection: 4 5
--------------------------------
Process exited after 31.15 seconds with return value 0
Press any key to continue . . .
```

3. Write a program in C to create two sets and perform the Difference operation on sets.

```
#include<stdio.h>

int a[100],b[100],i,j,k=0,n,m,flag=0;


void difference()

{

        printf("\nA-B\n");

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

         {

                flag=0;
```

```
                for(j=0;j<m;j++)
  {
if(a[i]==b[j])
{
        flag=1;
        break;
}
  }
if(flag==0)
printf("%d ",a[i]);
          }
printf("\n\nB-A\n");
for(i=0;i<m;i++)
{
flag=0;
for(j=0;j<n;j++)
{
if(b[i]==a[j])
{
flag=1;
break;
}
}
if(flag==0)
printf("%d ",b[i]);
}
```

```
}

int main()
{
 printf("Enter the size of array A\n");
 scanf("%d",&n);
 printf("Enter the element of First array A\n");
 for(i=0;i<n;i++)
 {
 scanf("%d",&a[i]);
 }
 printf("Enter the size of array B\n");
 scanf("%d",&m);
 printf("Enter the elements of array B\n");
 for(j=0;j<m;j++)
 {
 scanf("%d",&b[j]);
 }

 printf("difference of set\n");
 difference();

 return 0;
 }
```

Name :Fardeen Alam

rollno. : 1900270130070 sec: IT-2

```
Enter the size of array A
4
Enter the element of First array A
1
2
3
4
Enter the size of array B
6
Enter the elements of array B
3
4
5
6
7
8
difference of set

A-B
1 2

B-A
5 6 7 8
--------------------------------
Process exited after 15.39 seconds with return value 0
Press any key to continue . . .
```

4. Write a program in C to create two sets and perform the Symmetric Difference operation.

```
#include<stdio.h>

int a[10],b[10],d[10],i,j,k=0,n,m,flag=0;


void symmetric_diff()

{

 k=0;

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

 {
```

Name :Fardeen Alam
rollno. : 1900270130070 sec: IT-2

```
flag=0;
for(j=0;j<m;j++)
{
if(a[i]==b[j])
{
flag=1;
break;
}
}
if(flag==0)
{
d[k]=a[i];
k++;
}
}
for(i=0;i<m;i++)
{
flag=0;
for(j=0;j<n;j++)
{
if(b[i]==a[j])
{
flag=1;
break;
}
}
if(flag==0)
```

```
 {
 d[k]=b[i];
 k++;
 }
 }

 printf("\nSymmetric Difference (A-B)U(B-A) is \n");
 for(i=0;i<k;i++)
 {
 printf("%d ",d[i]);
 }
}
int main()
{
 printf("Enter the size of array A\n");
 scanf("%d",&n);
 printf("Enter the element of First array A\n");
 for(i=0;i<n;i++)
 {
 scanf("%d",&a[i]);
 }
 printf("Enter the size of array B\n");
 scanf("%d",&m);
 printf("Enter the elements of array B\n");
 for(j=0;j<m;j++)
 {
```

```
scanf("%d",&b[j]);

}

symmetric_diff();

printf("\n");

return 0;

}
```

```
Enter the size of array A
4
Enter the element of First array A
1
2
3
4
Enter the size of array B
6
Enter the elements of array B
3
4
5
6
7
8

Symmetric difference (A-B)U(B-A) is
1 2 5 6 7 8

--------------------------------
Process exited after 19.58 seconds with return value 0
Press any key to continue . . .
```

5. Write a program in C to perform the Power Set operation on a set.

```
#include<stdio.h>

#include<conio.h>

#include<math.h>

#define ROW 8

#define COL 256

void main()

{
```

Name :Fardeen Alam

rollno. : 1900270130070 sec: IT-2

```
int i,j,k,num,temp,flag;
int table[COL][ROW]={0};
char set[ROW][20];
printf("\nEnter the number of set elements: ");
scanf("%d",&num);
printf("\nEnter any %d elements:\n",num);
for(i=0;i<num;i++)
{
 fflush(stdin);
 gets(set[i]);

}
for(i=0;i<pow(2,num);i++)
{
 temp=i;
 for(j=0;j<num;j++)
 {
  table[i][j]=temp%2;
  temp=temp/2;
 }
}
printf("\nThe power set is:\n\n");
for(i=0;i<pow(2,num);i++)
{
 printf("\n\n{");
 for(j=0;j<num;j++)
```

Name :Fardeen Alam
rollno. : 1900270130070 sec: IT-2

```
  {
    flag=0;
    if(table[i][j]==1)
    {
   for(k=j+1;k<num;k++)
    if(table[i][k]==1)
      flag=1;
   if(flag==1)
    printf(" %s ,",set[j]);
   else
    printf(" %s  ",set[j]);
    }
   }
   printf("}");
 }
 getch();

}
```

Name :Fardeen Alam

rollno. : 1900270130070 sec: IT-2

```
Enter the number of set elements: 3

Enter any 3 elements:
1
2
3

The power set is:



{}

{ 1  }

{ 2  }

{ 1 , 2  }

{ 3  }

{ 1 , 3  }

{ 2 , 3  }

{ 1 , 2 , 3  }
--------------------------------
Process exited after 78.17 seconds with return value 61
Press any key to continue . . .
```

6. Write a program in C to Display the Boolean Truth Table for AND, OR, NOT.

Name :Fardeen Alam

rollno. : 1900270130070 sec: IT-2

```
#include<stdio.h>

int OR(int,int);

int AND(int,int);

int NOT(int);

int main()

{

 int a,b,ch,Y;

 printf("\n**** MENU ****\n");

 printf("\n 1. AND\n 2. OR\n 3. NOT\n 4. Quit");

 do

 {

 printf("\n\n Enter the choice: ");

 scanf("%d",&ch);

 switch (ch)

 {

 case 1: printf("\n Truth table for AND\nA B \tY");

 for (a=0; a <= 1; a++)

 {

 for (b=0; b <=1; b++)

 {

 Y =AND(a,b);

 printf("\n%d %d \t%d",a,b,Y);

 }


 }

 break;
```

```
case 2: printf("\n Truth table for OR\nA B \tY");

for (a=0; a <= 1; a++)

{for (b=0; b <=1; b++)

{

Y =OR(a,b);

printf("\n%d %d \t%d",a,b,Y);

}


}

break;

case 3: printf("\n Truth table for NOT\nA \tY");

for (a=0; a <= 1; a++)

{


Y =NOT(a);

printf("\n%d \t%d",a,Y);



}

break;

case 4:

break;


default: printf("\nError");

break;

}
```

```
 } while (ch!=4);


}
int OR(int a,int b)
{
 if (a==0&&b==0)
 {
 return 0;
 }
 else
 {
 return 1;
 }
}
int AND(int a,int b)
{
 if (a==1&&b==1){
 return 1;
 }
 else
 {
 return 0;
 }
}
int NOT(int a)
{
```

Name :Fardeen Alam

rollno. : 1900270130070 sec: IT-2

```
if (a==0)
{
return 1;
}
else
{
return 0;
}
}
```

Name :Fardeen Alam

rollno. : 1900270130070 sec: IT-2

```
**** MENU ****

 1. AND
 2. OR
 3. NOT
 4. Quit

 Enter the choice: 1

 Truth table for AND
A B     Y
0 0     0
0 1     0
1 0     0
1 1     1

 Enter the choice: 2

 Truth table for OR
A B     Y
0 0     0
0 1     1
1 0     1
1 1     1

 Enter the choice: 3

 Truth table for NOT
A       Y
0       1
1       0

 Enter the choice: 4

--------------------------------
Process exited after 23.45 seconds with return value 4
Press any key to continue . . .
```

7. Write a C Program to find Cartesian Product of two sets.

```
#include <stdio.h>

#include <conio.h>

int main()

{

   int a[50], b[50], c[50], i, s1, s2, j, k;

   printf("Enter how many elements in set 1\n");
```

```
    scanf("%d", &s1);
    printf("Enter how many elements in set 2\n");
    scanf("%d", &s2);
    printf("Enter elements of set 1\n");
    for (i = 0; i < s1; i++)
    {
        scanf("%d", &a[i]);
    }
    printf("Enter elements of set 2\n");
    for (i = 0; i < s2; i++)
    {
        scanf("%d", &b[i]);
    }
    printf("cartesian product=");
    printf("{");
    for (i = 0; i < s1; i++)
    {
        for (j = 0; j < s2; j++)
        {
            printf("(%d,%d)", a[i], b[j]);
            printf(",");
        }
    }
    printf("}");
}
```

Name :Fardeen Alam

rollno. : 1900270130070 sec: IT-2

```
Enter how many elements in set 1
3
Enter how many elements in set 2
3
Enter elements of set 1
10
11
12
Enter elements of set 2
14
15
16
cartesian product={(10,14),(10,15),(10,16),(11,14),(11,15),(11,16),(12,14),(12,15),(12,16),}
--------------------------------
Process exited after 32.08 seconds with return value 125
Press any key to continue . . .
```

8. Write a program in C for minimum cost spanning tree.

Name :Fardeen Alam

rollno. : 1900270130070 sec: IT-2

```
#include<stdio.h>

#include<stdlib.h>


#define infinity 9999

#define MAX 20


int G[MAX][MAX],spanning[MAX][MAX],n;


int prims();


int main()

{

        int i,j,total_cost;

        printf("Enter no. of vertices:");

        scanf("%d",&n);


        printf("\nEnter the adjacency matrix:\n");


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

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

                        scanf("%d",&G[i][j]);


        total_cost=prims();

        printf("\nspanning tree matrix:\n");


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

```
        {
                printf("\n");
                for(j=0;j<n;j++)
                        printf("%d\t",spanning[i][j]);
        }

        printf("\n\nTotal cost of spanning tree=%d",total_cost);
        return 0;
}

int prims()
{
        int cost[MAX][MAX];
        int u,v,min_distance,distance[MAX],from[MAX];
        int visited[MAX],no_of_edges,i,min_cost,j;

        //create cost[][] matrix,spanning[][]
        for(i=0;i<n;i++)
                for(j=0;j<n;j++)
                {
                        if(G[i][j]==0)
                                cost[i][j]=infinity;
                        else
                                cost[i][j]=G[i][j];
                                spanning[i][j]=0;
                }
```

Name :Fardeen Alam

rollno. : 1900270130070 sec: IT-2

```
//initialise visited[],distance[] and from[]
distance[0]=0;
visited[0]=1;

for(i=1;i<n;i++)
{
        distance[i]=cost[0][i];
        from[i]=0;
        visited[i]=0;
}

min_cost=0;             //cost of spanning tree
no_of_edges=n-1;                //no. of edges to be added

while(no_of_edges>0)
{
        //find the vertex at minimum distance from the tree
        min_distance=infinity;
        for(i=1;i<n;i++)
                if(visited[i]==0&&distance[i]<min_distance)
                {
                        v=i;
                        min_distance=distance[i];
                }
```

```
            u=from[v];

            //insert the edge in spanning tree
            spanning[u][v]=distance[v];
            spanning[v][u]=distance[v];
            no_of_edges--;
            visited[v]=1;

            //updated the distance[] array
            for(i=1;i<n;i++)
                  if(visited[i]==0&&cost[i][v]<distance[i])
                  {
                        distance[i]=cost[i][v];
                        from[i]=v;
                  }

            min_cost=min_cost+cost[u][v];
      }

      return(min_cost);
}
```

Name :Fardeen Alam

rollno. : 1900270130070 sec: IT-2

```
Enter no. of vertices:6

Enter the adjacenct matrix:
0 3 1 6 0 0
3 0 5 0 3 0
1 5 0 5 6 4
6 0 5 0 0 2
0 3 6 0 0 6
0 0 4 2 6 0

spanning tree matrix:

0       3       1       0       0       0
3       0       0       0       3       0
1       0       0       0       0       4
0       0       0       0       0       2
0       3       0       0       0       0
0       0       4       2       0       0

Total cost of spanning tree=13
```

9. Write a program in C for finding shortest path in a Graph.

```
#include<stdio.h>

#define INFINITY 9999

#define MAX 10


void dijkstra(int G[MAX][MAX],int n,int startnode);


int main()

{

  int G[MAX][MAX],i,j,n,u;

  printf("Enter no. of vertices:");

  scanf("%d",&n);

  printf("\nEnter the adjacency matrix:\n");
```

```
 for(i=0;i<n;i++)
   for(j=0;j<n;j++)
     scanf("%d",&G[i][j]);

 printf("\nEnter the starting node:");
 scanf("%d",&u);
 dijkstra(G,n,u);

 return 0;
}

void dijkstra(int G[MAX][MAX],int n,int startnode)
{

 int cost[MAX][MAX],distance[MAX],pred[MAX];
 int visited[MAX],count,mindistance,nextnode,i,j;

 //pred[] stores the predecessor of each node
 //count gives the number of nodes seen so far
 //create the cost matrix
 for(i=0;i<n;i++)
   for(j=0;j<n;j++)
     if(G[i][j]==0)
       cost[i][j]=INFINITY;
```

```
    else
      cost[i][j]=G[i][j];

 //initialize pred[],distance[] and visited[]
 for(i=0;i<n;i++)
 {
  distance[i]=cost[startnode][i];
  pred[i]=startnode;
  visited[i]=0;
 }

 distance[startnode]=0;
 visited[startnode]=1;
 count=1;

 while(count<n-1)
 {
  mindistance=INFINITY;

  //nextnode gives the node at minimum distance
  for(i=0;i<n;i++)
   if(distance[i]<mindistance&&!visited[i])
   {
     mindistance=distance[i];
     nextnode=i;
```

```
    }

    //check if a better path exists through nextnode
    visited[nextnode]=1;
    for(i=0;i<n;i++)
      if(!visited[i])
        if(mindistance+cost[nextnode][i]<distance[i])
        {
          distance[i]=mindistance+cost[nextnode][i];
          pred[i]=nextnode;
        }
  count++;
}

//print the path and distance of each node
for(i=0;i<n;i++)
  if(i!=startnode)
  {
    printf("\nDistance of node%d=%d",i,distance[i]);
    printf("\nPath=%d",i);

    j=i;
    do
    {
      j=pred[j];
```

```
        printf("<-%d",j);

      }while(j!=startnode);

  }

}
```

```
Enter no. of vertices:5

Enter the adjacency matrix:
0 10 0 30 100
10 0 50 0 0
0 50 0 20 10
30 0 20 0 60
100 0 10 60 0

Enter the starting node:0

Distance of node 1=10
 Path=1<-0
Distance of node 2=50
 Path=2<-3<-0
Distance of node 3=30
 Path=3<-0
Distance of node 4=60
 Path=4<-2<-3<-0
Process returned 5 (0x5)   execution time : 47.471 s
Press any key to continue.
```

10.(Computation software).

Mapple:

Maple is a symbolic and numeric computing environment, and is also a multi-paradigm programming language. Developed by Maplesoft, Maple also covers other aspects of technical computing, including visualization, data analysis, matrix computation, and connectivity.

Name :Fardeen Alam

rollno. : 1900270130070 sec: IT-2

Core functionality:

Users can enter mathematics in traditional mathematical notation. Custom user interfaces can also be created. There is support for numeric computations, to arbitrary precision, as well as symbolic computation and visualization. Examples of symbolic computations are given below.

Maple incorporates a dynamically typed imperative-style programming language which resembles Pascal. The language permits variables of lexical scope. There are also interfaces to other languages (C, C#, Fortran, Java, MATLAB, and Visual Basic). There is also an interface to Excel. Maple supports MathML 2.0, a W3C format for representing and interpreting mathematical expressions, including their display in Web pages.

Architecture

Maple is based on a small kernel, written in C, which provides the Maple language. Most functionality is provided by libraries, which come from a variety of sources. Most of the libraries are written in the Maple language; these have viewable source code. Many numerical computations are performed by the NAG Numerical Libraries, ATLAS libraries, or GMP libraries.

Different functionality in Maple requires numerical data in different formats. Symbolic expressions are stored in memory as directed acyclic graphs. The standard interface and calculator interface are written in Java.

11.Write a program in c to print all combinations of a given string.

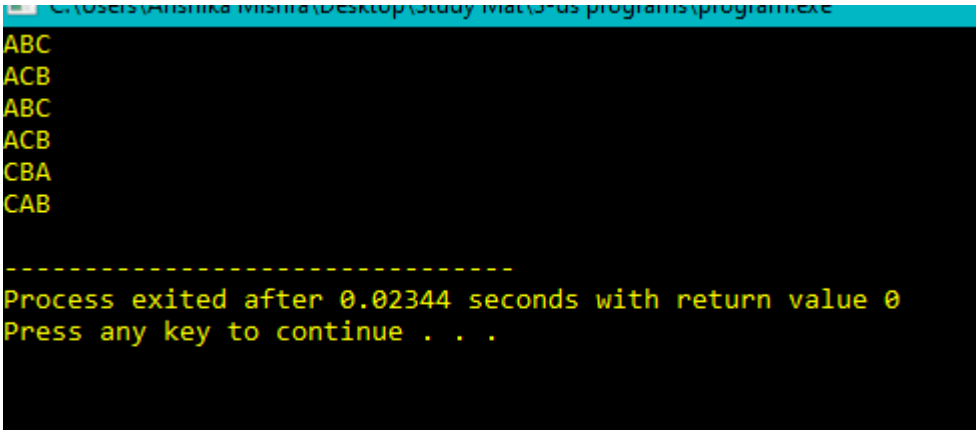
```
#include<stdio.h>
#include<string.h>
```

Name :Fardeen Alam
rollno. : 1900270130070 sec: IT-2

```
void swap(char *x, char *y)
{
   char temp;
   temp=*x;
   *x=*y;
   *y=temp;
}

void permute(char *a,int l,int r)
{
   int i;
   if(l==r)
      printf("%s\n",a);
   else
   {
      for(i=l;i<=r;i++)
      {
         swap((a+l),(a+i));
         permute(a,l+1,r);
         swap((a+1),(a+i));
      }
   }
}

int main()
{
   char str[]="ABC";
   int n=strlen(str);
   permute(str,0,n-1);
   return 0;
}
```

Name :Fardeen Alam

rollno. : 1900270130070 sec: IT-2



Name :Fardeen Alam

rollno. : 1900270130070 sec: IT-2

12.Write a program to Implement recursion and induction.

```
#include <stdio.h>
int sum (int a);
int main()
{
   int num, result;

   printf("Enter the number: ");
   scanf("%d", &num);
   result = sum(num);
   printf("Sum of digits in %d is %d\n", num, result);
   return 0;
}

int sum (int num)
{
   if (num != 0)
   {
      return (num % 10 + sum (num / 10));
   }
   else
   {
     return 0;
   }
}
```

```
Enter the number: 819
Sum of digits in 819 is 18

--------------------------------
Process exited after 3.816 seconds with return value 0
Press any key to continue . . .
```

Name :Fardeen Alam

rollno. : 1900270130070 sec: IT-2

13. Permutations and combinations.

Permutation is an arrangement of some elements in which order matters. In other words a Permutation is an ordered Combination of elements.

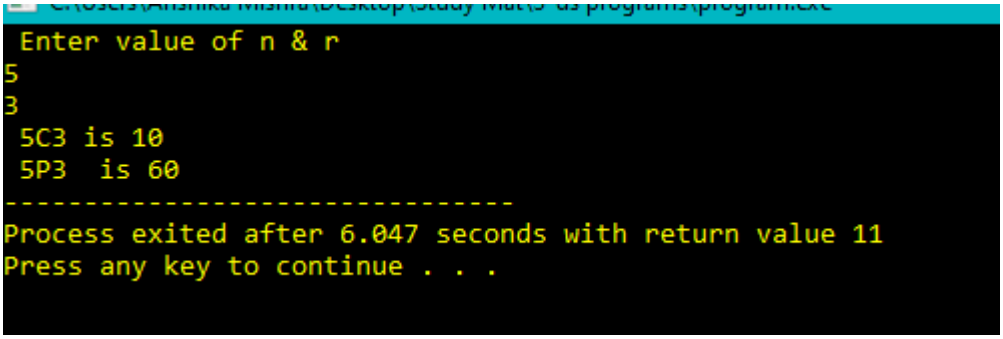
Combination is selection of some given elements in which order does not matter.

```
#include <stdio.h>

#include <conio.h>

main() {

        int n , r, ncr( int , int);

        long npr( int , int);

        long double fact( int);

        printf(" Enter value of n & r \n");

        scanf("%d %d",&n , &r);

        if( n>= r) {

                printf( " %dC%d is %d\n", n,r,ncr( n , r));

                printf(" %dP%d  is %ld", n,r,npr( n, r));

        } else {

                printf("WRONG INPUT?? enter the correct input");

        }

}

long double fact( int p) {

        long double facts = 1;
```

Name :Fardeen Alam

rollno. : 1900270130070 sec: IT-2

```
        int i;

        for ( i = 1; i<= p; i++)

          facts = facts * i;

        return( facts);

}

int ncr ( int n, int r) {

        return( fact( n) / (fact( r) * fact(n- r) ) ) ;

}

long npr( int n , int r) {

        return( fact( n) / fact( n- r));

    }
```

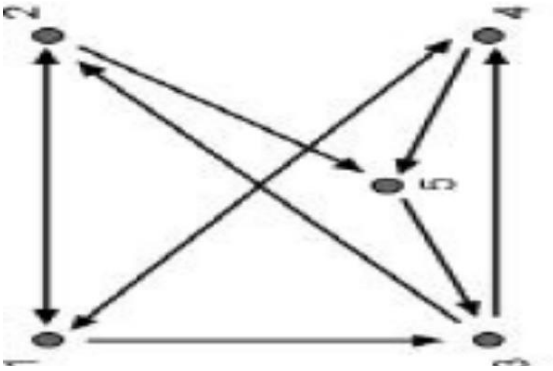


Name :Fardeen Alam

rollno. : 1900270130070 sec: IT-2

14. Binary Relations: Influence

As part of its campaign to step up a recycling project, a group wants to entertain the most influential members of the city council. After some investigation, the firm's perception of pairwise influence among council members is as shown below. (An arc goes from member x to member y if x appears to influence y.)



Open the Maple file relations.mws, and select Execute Worksheet from the Edit menu. Save your worksheet using a different file name so that you do not write over this beginning file. Define the above relation using the following command:

> R := [[[1,2],[1,3],[1,4],[2,1],[2,5],[3,2],[3,4],[4,1], [4,5],[5,3]], 5];

Compare the way that Maple draws the arrow diagram with the picture above.

> drawGraph(R);

Write the adjacency matrix M corresponding to this graph.

> M := listToMatrix(R);

Suppose we want the matrix that shows indirect influence. Person x has indirect influence over person z if person x has direct influence over person y and person y has direct influence over person z. Use Maple to evaluate the matrix M2, and explain what this matrix has to do with
indirect influence.

Name :Fardeen Alam
rollno. : 1900270130070 sec: IT-2

> evalm(M^2);

Now consider that matrix sum M + M2. Compute this sum and explain how it relates to influence relation in this problem.

> evalm(M + M^2);

15. Poker Hands problem

In one version of Poker, each player gets a five-card hand of cards from a standard deck. Here are a few types of poker hands:
Full House – A hand consisting of 3 cards of the same denomination and 2 cards of the same denomination (i.e. 3 of a kind and a pair). Pair– A hand consisting of exactly one pair of cards with the same denomination.
What is the probability of obtaining each of the above poker hands?
Again, we'll use Maple to simulate.
Procedure:
The appropriate command this time is

```
>with(combinat,randcomb);
randset := combinat[randcomb];
cardDeck := {Ac,Ah,As,Ad, Kc,Kh,Ks,Kd,
Qc,Qh,Qs,Qd, Jc,Jh,Js,Jd, seq(i.c, i=2..10),
seq(i.h, i=2..10), seq(i.s, i=2..10), seq(i.d,i=2..10)};
randset(cardDeck,5);
```

This returns the following:
{Ah,Qd,4 d,3 s,9 c}
where Ah stands for ace of hearts, Qd stands for Queen of diamonds, and so forth.
Simulate each game 20 times and keep track of your results in the following tables.
Full House
Trial 1 2 3 4 5 6 7 8 9 10
Full House?
Trial 11 12 13 14 15 16 17 18 19 20

Name :Fardeen Alam

rollno. : 1900270130070 sec: IT-2

16. Baseball best-of-5 series: Experimental probabilities

The Baltimore Orioles and the Boston Red Sox are playing a best of 5 series. This means that whichever team is the first to win 3 games is the winner of the series. Let's first suppose that both teams are evenly matched. Before doing any calculations, use your intuition to answer the following questions: What is the probability that the Orioles will sweep the Red Sox?

Is the series most likely to end in 3, 4, or 5 games? On average, how long do you expect the series to last?

We'll use Maple to help us simulate this best-of-5 series. The script is as follows, where n = 2 and k = 5.

```
> randseries:=proc(n,k)
local i;
randomize();
RETURN(map(rand(1..n),[seq(i,i=1..k)]))
end proc;
```

Let Orioles := 1 and Red Sox :=2. Note that the series doesn't necessarily go to 5 games; if Boston sweeps the Orioles (i.e. any sequence that begins with 222) then the series is over in 3 games. Execute the procedure 20 times and record your results in the following table.

# of games needed	3	4	5
Orioles win the series			
Boston wins the series			

Name :Fardeen Alam

rollno. : 1900270130070 sec: IT-2

17. Expected Value Problems

Suppose five cards including an Ace are drawn from a standard deck. (If the five cards do not include an Ace, they are replaced and the deck is reshuffled.) What is the expected number of aces among the five cards?

We'll approach this problem using simulated data first. Begin by using Maple to randomly select 5 cards from a fair deck of 52.
The following script generates 50 5-card hands from a fair deck of 52.

Procedure

```
>with(combinat, randcomb);
>carddeck:={Ac, Ac, As, Ad, Kc, Kh, Ks, Kd,
Qc, Qh, Qs, Qd, Jc,Jh,Js,Jd,seq(i.c,i=2..10),
seq(i.h,i=2..10),seq(i.s,i=2..10),
seq(i.d,i=2..10)};
>aces:={Ac,Ah,As,Ad}
>for i from 1 to 50 do
sort(randcomb(carddeck,5),lexorder);
end do;
```

Remember to eliminate hands that do not to contain at least one ace.

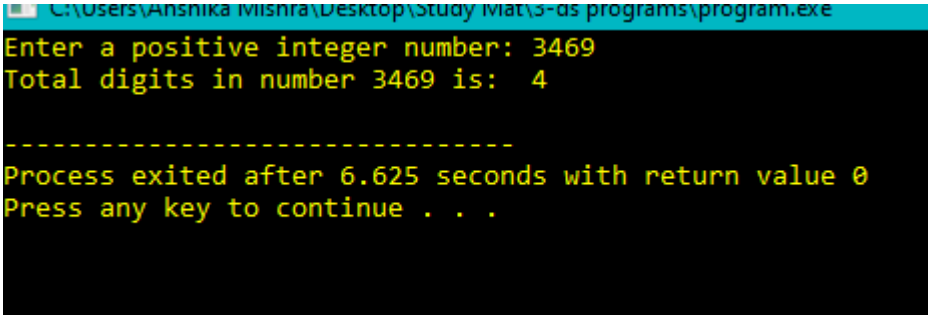
Name :Fardeen Alam

rollno. : 1900270130070 sec: IT-2

18. Implementation of a recursive counting technique

```
#include <stdio.h>

//function to count digits
int countDigits(int num)
{
   static int count=0;

   if(num>0)
   {
      count++;
      countDigits(num/10);
   }
   else
   {
      return count;
   }
}
int main()
{
```

```
    int number;
    int count=0;

    printf("Enter a positive integer number: ");
    scanf("%d",&number);

    count=countDigits(number);

    printf("Total digits in number %d is:  %d\n",number,count);

    return 0;
}
```



19. Combinatorial equivalence

```
#include <stdio.h>
#define MAX 10
int linearsearch(int numbers[], int key)
{
  int i;
  for (i = 0; i < MAX; i++)
  {
     if (key == numbers[i])
        return i;
  }
  return -1;
}
int binarysearch(int numbers[], int key)
{
  int l = 0, u = MAX - 1;
  int c, mid;
   while (l <= u){
```

```
      mid = (l + u) / 2;
      if (key == numbers[mid]){
         return mid;
      }
      else if (key < numbers[mid]){
         u = mid - 1;
      }
      else
         l = mid + 1;
  }
  return -1;
}
int main() {
  int numbers[MAX];
  int i;
  int index, key;
  printf("Enter %d numbers : ", MAX);
  for (i = 0; i < MAX; i++)
  {
      scanf("%d", &numbers[i]);
  }
  printf("\nEnter a key to find using linear search: ");
  scanf("%d", &key);
  index = linearsearch(numbers, key);
   if ( index >= 0)
     printf("\n%d found at index %d using linear search.", key, index);
  else
    printf("\nNot found!!");
  printf("\nEnter %d numbers in increasing order: ", MAX);
  for (i = 0 ; i < MAX; i++)
     scanf("%d", &numbers[i]);
  printf("\nEnter a key to find using binary search: ");
  scanf("%d", &key);
  index = binarysearch(numbers, key);
  if (index >= 0 )
      printf("Found at index %d", index);
  else
     printf("\nNot found!!!");
```

```
  return 0;
 }
```

```
Enter 10 numbers : 2
2
3
5
6
1
8
3
9
5

Enter a key to find using linear search: 6

6 found at index 4 using linear search.
Enter 10 numbers in increasing order: 2
3
10
12
16
19
23
29
35
69

Enter a key to find using binary search: 35
Found at index 8
--------------------------------
Process exited after 128 seconds with return value 0
Press any key to continue . . .
```

20. Counting

```
#include <stdio.h>
#define MAX 7

void pigeonhole_sort(int, int, int *);
void main()
{
 int a[MAX], i, min, max;
  printf("enter the values into the matrix :");
```

```
  for (i = 0; i < MAX; i++)
  {
     scanf("%d", &a[i]);
  }
  min = a[0];
  max = a[0];
  for (i = 1; i < MAX; i++)
  {
     if (a[i] < min)
     {
        min = a[i];
     }
     if (a[i] > max)
     {
        max = a[i];
     }
  }
  pigeonhole_sort(min, max, a);
  printf("Sorted order is :\n");
  for (i = 0; i < MAX; i++)
  {
     printf("%d", a[i]);
  }
}

/* sorts the array using pigeonhole algorithm */
void pigeonhole_sort(int mi, int ma, int * a)
{

  int size, count = 0, i;
  int *current;
  current = a;
  size = ma - mi + 1;
  int holes[size];
   for (i = 0; i < size; i++)
 {
      holes[i] = 0;
   }
   for (i = 0; i < size; i++, current++)
```

Name :Fardeen Alam

rollno. : 1900270130070 sec: IT-2

```
  {
     holes[*current-mi] += 1;
  }
  for (count = 0, current = &a[0]; count < size; count++)
  {
     while (holes[count]--> 0)
    {
       *current++ = count + mi;
    }
  }
}
```