PRACTICAL No. 1 (A)

CARDINALITY

AIM: Write a program in Scilab to find cardinality of the following sets

(i) A = {1, 3, 5 7, 9}
(ii) B = {x, y, z}
(iii) C = {Apple, Acer, Sony, Dell, Lenovo}

SOURCE CODE:

```
A=[1,3,5,7,9];
a=length(A);
disp(a,'Cardinality of set A is:')
x=1;
y=2;
z=3;
B=[x,y,z];
b=length(B);
disp(b,'Cardinality of set B is:');
Apple=2;
Acer=4;
Sony=6;
Dell=8;
Lenovo=10;
C=[Apple, Acer, Sony, Dell, Lenovo];
c=length(C);
disp(c,'Cardinality of set C is:');
```

OUTPUT:

```
exec('E:\Scilab program files\cardinality.sce');

Cardinality of set A is:

 5.

Cardinality of set B is:

 3.

Cardinality of set C is:

 5.
```

PRACTICAL No.1 (B)

POWER SETS

AIM: Write a program in Scilab to find number of proper subsets of A where

(i) |A| = 5.

(ii) A = {Triangle, Rectangle, Square, Pentagon, Hexagon, Star}.

SOURCE CODE:

(i)

```
n=5;//number of elements of set A
m=2^n-1;//number of elements of power set of A excluding A itself
disp(m,'Number of proper subsets A =');
```

OUTPUT:

```
exec('E:\Scilab program files\power_set1.sce');

Number of proper subsets A =

31.
```

SOURCE CODE:

(ii)

```
Triangle=1;
Rectangle=2;
Square=3;
Pentagon=4;
Hexagon=5;
Star=6;
A=[Triangle, Rectangle, Square, Pentagon, Hexagon, Star];
n=length(A);//number of members in A
m=2^n-1;//number of elements of power set of A excluding A itself
disp(m,'Number of proper subsets of A =');
```

OUTPUT:

```
exec('E:\Scilab program files\power_set1.sce');

Number of proper subsets of A =

63.
```

PRACTICAL No. 1 (C)

INCLUSION EXCLUSION PRINCIPLE

AIM: Write a program in Scilab for the following inclusion exclusion problems

Q.1 Out of 1200 students of a college, 552 took Economics, 627 took Mathematics, 540 took Information Technology, 217 took Economics and Mathematics, 307 took Economics and Information Technology, 240 took Mathematics and Information Technology, 213 took all the three subjects.

(i) How many took at least one of the three subjects?

(ii) How many took none of the three subject?

SOURCE CODE:

```
U=1200;//total number of students
E=552;//number of students taken Economics
M=627;//number of students taken Mathematics
IT=540;//number of students taken Information Technology
EandM=217;//number of students taken Economics and Mathematics
EandIT=307;//number of students taken Economics and Information
Technology
MandIT=240;//number of students taken Mathematics and Information
Technology
EandMandIT=213;//number of students taken all the three subjects
//To find number of students taken at least one of the three subjects
//By InclusionExclusion Principle
EorMorIT=E+M+IT-EandM-EandIT-MandIT+EandMandIT;
disp(EorMorIT,'Number of students taken at least one of the three
subjects is:');
//To find number of students not taken any of the three subjects
None=U-EorMorIT;
disp(None,'Number of students not taken any of the three subjects
is:');
```

OUTPUT:

```
exec('E:\Scilab program files\inc_exc1.sce');
Number of students taken at least one of the three subjects is:
1168.
Number of students not taken any of the three subjects is:
32.
```

Q.2 A survey of 550 T.V. watchers produced the following information: 285 watch football, 195 watch hockey, 115 watch baseball, 45 watch football and baseball, 70 watch football and hockey, 50 watch hockey and baseball, 100 do not watch any of the three games.

(i) How many people in the survey watch all the three games?

(ii) How many people watch exactly one of the three games?

SOURCE CODE:

```
U=550;//total number of people surveyed
F=285;//number of people watch football
H=195;//number of people watch hockey
B=115;//number of people watch baseball
FandB=45;//number of people watch football and baseball
FandH=70;//number of people watch football and hockey
HandB=50;//number of people watch hockey and baseball
None=100;//number of people watch none of the three
//To find number of people watch all the three games
ForHorB=U-None;//number of people watch at least one of the three
//By Inclusion Exclusion Principle
FandHandB=ForHorB-F-H-B+FandH+FandB+HandB;
disp(FandHandB,'Number of people watch all the three games is:');
//To find number of people watch exactly one of the three games
OnlyF=F-FandH-FandB+FandHandB;
OnlyH=H-HandB-FandH+FandHandB;
OnlyB=B-HandB-FandB+FandHandB;
ExactlyOne=OnlyF+OnlyH+OnlyB;//number of people watch exactly
one of the three games
disp(ExactlyOne,'Number of people watch exactly one of the three
games.is:');
```

OUTPUT:

```
exec('E:\Scilab program files\inc_exc2.sce');

Number of people watch all the three games is:

 20.

Number of people watch exactly one of the three games.is:

 325.
```

PRACTICAL No. 1 (D)

MATHEMATICAL INDUCTION

AIM: Write a program in Scilab for the following:

If $U_n = 3U_{n-1} - 2U_{n-2},\ U_1 = 3, U_2 = 5$ then prove that $U_n = 1 + 2^n,\ \forall n \in \mathbb{N}$.

SOURCE CODE:

```
U=[];
P=[];
U(1)=3;
U(2)=5;
k=0;
for i=1:10
    P(i)=1+2^i
    if i>=3 then
        U(i)=3*U(i-1)-2*U(i-2);
    end
    if P(i)==U(i) then
        k=k+1;
    end
end
if k==10 then
    disp('Hence Un=1+2^n for all n belongs to set of natural numbers.');
else
    disp('Statement is not true.');
end
```

OUTPUT:

exec('E:\Scilab program files\induction.sce');

Hence Un=1+2^n for all n belongs to set of natural numbers.

PRACTICAL No. 2 (A)

RECURSIVELY DEFINED FUNCTIONS

AIM: Write a program in Scilab to compute factorial by recursively defined functions.

(non-recursive function)
SOURCE CODE:

```
function fact(a)
    if(a<0|a>100)
        k=0;
        disp('INVALID');
        break;
    elseif(a==0|a==1)
            k=1;
        else
            k=1;
            for i=a:-1:1
                k=k*i;
            end
        end
disp(k, 'factorial value is: ');
endfunction
```

OUTPUT:

```
exec('E:\Scilab program
files\fact.sci');
fact(5);
factorial value is:
120.
```

(recursive function)
SOURCE CODE:

```
function [k]=fact1(a)
    if (a<0|a>170) then
        k=-1;
        disp('INVALID');
        break;
    elseif (a==0|a==1) then
            k=1;
        else
            k=a*fact1(a-1);
    end
endfunction
a=input("Enter value: ");
p=fact1(a);
disp(p,'Factorial value is: ');
```

OUTPUT:

```
exec('E:\Scilab program
files\fact1.sci');
Enter value: 4
Factorial value is:
24.
```

(recursive function)
SOURCE CODE:

```
function [k]=fact2(a)
if(a<0|a>170)
k=-1;
disp("Invalid");
break;
 elseif(a==1|a==0)
k=1;
else
k=a*fact2(a-1);
end
endfunction
```

OUTPUT:

```
exec('E:\Scilab
program
files\fact2.sci', -1)
fact2(5)
 ans  =
120.
fact2(6)
ans  =
720.
```

PRACTICAL No. 2 (B)

POLYNOMIAL EVALUATION

AIM: Write a program in Scilab to evaluate the following polynomials:

(i) $f(x) = x^3 - 2x + 1 \; at \; x = 2$.
(ii) $g(x) = x^2 - 1 \; at \; x = 3$.
(iii) $h(x) = 2x^3 - 7x^2 + 4x - 15 \; at \; x = 5$.

SOURCE CODE:

```
f=poly([1,-2,0,1],'x','c');
disp(f,'the polynomial f is ');
k=horner(f,2);
disp(k,'value of polynomial f at x = 2 is ');
g=poly([1,-1],'x','r');
disp(g,'the polynomial g is ');
l=horner(g,3);
disp(l,'value of polynomial g at x = 3 is ');
x = poly(0, 'x');
h = 2*x^3-7*x^2+4*x-15;
disp(h,'the polynomial h is')
m=horner(h,5);
disp(m,'value of the polynomial h at x = 5 is ');
```

OUTPUT:

```
Scilab Console

-->exec('E:\Scilab program files\polynomial.sce');
 the polynomial f is

                3
    1 - 2x + x

 value of polynomial f at x = 2 is

    5.

 the polynomial g is

          2
  - 1 + x

 value of polynomial g at x = 3 is

    8.

 the polynomial h is

                  2     3
  - 15 + 4x - 7x + 2x

 value of the polynomial h at x = 5 is

    80.
```

PRACTICAL No. 2 (C)

GREATEST COMMON DIVISOR

AIM: Write a program in Scilab to compute Greatest Common Divisor.

SOURCE CODE:

```
function thegcd(a, b)
   A=int32([a,b]);
ans=gcd(A);
disp(ans,'the gcd of the two numbers is');
endfunction
```

OUTPUT:

```
exec('E:\Scilab program files\thegcd.sci');

thegcd(220,15);

the gcd of the two numbers is

5
```

SOURCE CODE:

```
function thegcd1(a, b, c)
   A=int32([a,b,c]);
ans=gcd(A);
disp(ans,'the gcd of the three numbers is')
endfunction
```

OUTPUT:

```
exec('E:\Scilab program files\thegcd1.sci');

thegcd1(50,25,250);

the gcd of the three numbers is

25
```

PRACTICAL No. 3 (A)

SUM RULE PRINCIPLE

AIM: Write a program in Scilab for the following problems:

Q.1 Suppose a bookcase shelf has 6 Mathematics textbooks, 5 Programming textbooks and 3 Networking textbooks. Find the number of ways a student can choose a textbook.

Q.2 Find number of ways to select an integer between 1 to 20 which are divisible by 4 or prime.

Q.3 Find number of ways to select an integer between 1 to 20 which are divisible by 2 or 5.

SOURCE CODE:

```
M=6;    //number of mathematics textbook
P=5;    //number of programming textbook
N=3;    //number of networking textbook
T=M+P+N;
disp(T,'Number of ways a student can choose a textbook:');
E=[4,8,12,16,20];    //event of choosing numbers divisible by 4
F=[2,3,5,7,11,13,17,19];    //event of choosing prime number
EorF=union(E,F);
disp(length(EorF),'Number of ways of selecting an integer between 1 to
20 which are divisible by 4 or prime:');
P=[2,4,6,8,10,12,14,16,20];//event of choosing numbers divisible by 2
Q=[5,10,15,20];    //event of choosing numbers divisible by 5
R=intersect(P,Q);    //event of choosing numbers divisible by 2 and 5
PorQ=length(P)+length(Q)-length(R)
disp(PorQ,'Number of ways of selecting an integer between 1 to 20
which are divisible by 2 or 5:');
```

OUTPUT:

```
exec('E:\Scilab program files\sum_rule.sce');

 Number of ways a student can choose a textbook:
    14.

 Number of ways of selecting an integer between 1 to 20 which are
divisible by 4 or prime:
    13.

 Number of ways of selecting an integer between 1 to 20 which are
divisible by 2 or 5:
    11.
```

PRACTICAL No. 3 (B)

PRODUCT RULE PRINCIPLE

AIM: Write a program in Scilab for the following problems:

Q.1 Suppose a bookcase shelf has 6 Mathematics textbooks, 5 Programming textbooks and 3 Networking textbooks. Find the number of ways a student can choose one of each type of textbook.

Q.2 How many four digit numbers can be formed with digits 0 – 9 if

(i) repetition of digits not allowed?

(ii) repetition of digits allowed?

SOURCE CODE:

```
M=6;    //number of mathematics textbook
P=5;    //number of programming textbook
N=3;    //number of networking textbook
T=M*P*N;
disp(T,'Number of ways a student can choose a textbook of each
type:');
n=10;
k1=(n-1)*(n-1)*(n-2)*(n-3);
disp(k1,'Number of four digit numbers without repetition of digits:');
k2=(n-1)*n*n*n;
disp(k2,'Number of four digit numbers with repetition of digits:');
```

OUTPUT:

```
exec('E:\Scilab program files\product_rule.sce');

 Number of ways a student can choose a textbook of each type:

   90.

 Number of four digit numbers without repetition of digits:

   4536.

 Number of four digit numbers with repetition of digits:

   9000.
```

PRACTICAL No. 3 (C)

FACTORIAL NOTATION

AIM: Write a program in Scilab for the following

Q.1 Find first six factorial values by using $n! = n \times (n-1)!$ without using inbuilt factorial function.

Q.2 Compute (i) 6! (ii) 6!/4! (iii) 10!/7! , by using inbuilt factorial function.

SOURCE CODE:

```
//finding some initial factorial values without using inbuilt factorial
function
//calculation by using n!=n*(n-1)!
f0=1;
disp(f0,'0!=');
f1=1*f0;
disp(f1,'1!=');
f2=2*f1;
disp(f2,'2!=');
f3=3*f2;
disp(f3,'3!=');
f4=4*f3;
disp(f4,'4!=');
f5=5*f4;
disp(f5,'5!=');
//6! by using inbuilt factorial function
f6=factorial(6);
disp(f6,'6!=');
//finding 6!/4! and 10!/7!
k1=factorial(6)/factorial(4);
disp(k1,'value of 6!/4! is:');
k2=factorial(10)/factorial(7);
disp(k2,'value of 10!/7! is:');
```

OUTPUT:

```
exec('E:\Scilab program files\counting_factorial.sce');

 0!=
    1.

  1!=
    1.

 2!=

    2.

 3!=

    6.

 4!=

    24.

 5!=

    120.

 6!=

    720.

 value of 6!/4! is:

    30.

 value of 10!/7! is:

    720.
```

PRACTICAL No. 3 (D)

BINOMIAL COEFFICIENTS

AIM: Write a program in Scilab to calculate Binomial Coefficient ${}^{n}C_{r}$, where n and r values are given by user (n=10 & r=7)
and to verify ${}^{n}C_{r} = {}^{n}C_{n-r}$ (for n=8 & r=5).

SOURCE CODE:

```
function [k]=bnml(n, r)
    k=factorial(n)/(factorial(r)*factorial(n-r));
endfunction
a=input('Enter value for n:');
b=input('Enter value for r:');
ans=bnml(a,b);
disp(ans,'Value of required binomial coefficient is:');
//to verify nCr=nC(n-r)
n=input('Enter value for n:');
r=input('Enter value for r:');
p=bnml(n,r);
disp(p,'Value of nCr is:');
q=bnml(n,n-r);
disp(q,'Value of nC(n-r) is:');
if p==q then
    disp('Hence nCr=nC(n-r) verified.')
else
    disp('nCr=nC(n-r) not verified.')
end
```

OUTPUT:

```
exec('E:\Scilab program files\bnml.sci');
Enter value for n:10
Enter value for r:7

 Value of required binomial coefficient is:
    120.
Enter value for n:8
Enter value for r:5

 Value of nCr is:
    56.

 Value of nC(n-r) is:
    56.

 Hence nCr=nC(n-r) verified.
```

PRACTICAL No. 3 (E)

PERMUTATIONS

AIM: Write a program in Scilab for the following:

Q.1 A teacher is preparing an examination time table for 5 papers to be held on 5 consecutive days. How many different time table can she make?

Q.2 A manager has 10 persons working under him and he is expected to award 3 prizes to the persons whom he ranks are the top three achievers in the previous year. How many choices does he have?

SOURCE CODE:

```
function [k]=prmtn(n, r)
   k=factorial(n)/factorial(n-r);
endfunction
n=5;//number of days available
r=5;//number of papers to schedule
ans1=prmtn(n,r);
disp(ans1,'Number of different time tables:');
a=10;//total number of people available
b=3;//number people to select
ans2=prmtn(a,b);
disp(ans2,'Number of choices manager has:');
```

OUTPUT:

```
exec('E:\Scilab program files\prmtn.sci');
 Number of different time tables:
    120.
 Number of choices manager has:
    720.
```

PRACTICAL No. 3 (F)

PERMUTATIONS WITH REPETITIONS

AIM: Write a programme in Scilab for the following:

Q.1 Find number of words can be formed by using letters of the word

(i) UNUSUAL

(ii) SOCIOLOGICAL

SOURCE CODE:

```
function [k]=rprmtn(n, r1, r2, r3, r4)
   k=factorial(n)/(factorial(r1)*factorial(r2)*factorial(r3)*factorial(r4));
endfunction
//number of words can be formed by using letters of the word
UNUSUAL
a=7;//total number of letters in the word UNUSUAL
b=3;//letter U is repeated 3 times
ans1=rprmtn(a,b,0,0,0);
disp(ans1,'Number of words can be formed by using letters of the word
UNUSUAL is:');
//number of words can be formed by using letters of the word
SOCIOLOGICAL
n=12;//total number of letters in the word SOCIOLOGICAL
r1=3;//letter O is repeated 3 times
r2=2;//letter C is repeated 2 times
r3=2;//letter I is repeated 2 times
r4=2;//letter L is repeated 2 times
ans2=rprmtn(n,r1,r2,r3,r4);
disp(ans2,'Number of words can be formed by using letters of the word
SOCIOLOGICAL is:');
```

OUTPUT:

```
exec('E:\Scilab program files\rprmtn.sci', -1)

 Number of words can be formed by using letters of the word
UNUSUAL is:

    840.

 Number of words can be formed by using letters of the word
SOCIOLOGICAL is:

    9979200.
```

PRACTICAL No. 3 (G)

COMBINATIONS

AIM: Write a program in Scilab for the following:

Q.1 In how many ways can a committee of 8 people be formed out of a group of 10 men and 5 women?

Q.2 In how many ways can a committee of 6 men and 2 women be formed out of a group of 10 men and 5 women?

SOURCE CODE:

```
function [k]=cmbntn(n, r)
    k=factorial(n)/(factorial(r)*factorial(n-r));
endfunction
m=10;//total number of men
w=5;//total number of women
r=8;//number of people to select
ans1=cmbntn(m+w,r);
disp(ans1,'Number of ways to form a committee of 8 out of a group of
10 men and 5 women is:');
r1=6;//number of men to be selected
r2=2;//number of women to be selected
ans2=cmbntn(m,r1)*cmbntn(w,r2);
disp(ans2,'Number of ways to form a committee of 6 men and 2
women out of a group of 10 men and 5 women is:');
```

OUTPUT:

```
exec('E:\Scilab program files\cmbntn.sci');
```

Number of ways to form a committee of 8 out of a group of 10 men and 5 women is:

6435.

Number of ways to form a committee of 6 men and 2 women out of a group of 10 men and 5 women is:

2100.

PRACTICAL No. 3 (H)

COMBINATION WITH REPETITION

AIM: Write a program in Scilab for the following:

Q.1 A bagel shop sells 5 kinds of bagels. Find the number of ways a customer can buy: (*a*) 8 bagels; (*b*) a dozen bagels.

SOURCE CODE:

```
function [k]=rcmbntn(r, m)
   k=factorial(r+(m-1))/(factorial(r)*factorial(m-1));
endfunction
r1=8;//number of objects to select
m=5;//kinds of objects available
ans1=rcmbntn(r1,m);
disp(ans1,'Number of ways a customer can buy 8 bangles from 5
different kinds of bangles is:');
r2=12;//number of objects to select
ans2=rcmbntn(r2,m);
disp(ans2,'Number of ways a customer can buy 12 bangles from 5
different kinds of bangles is:');
```

OUTPUT:

```
exec('E:\Scilab program files\rcmbntn.sci');

 Number of ways a customer can buy 8 bangles from 5 different kinds
       of bangles is:

   495.

 Number of ways a customer can buy 12 bangles from 5 different
       kinds of bangles is:

   1820.
```

PRACTICAL No. 3 (I)

ORDERED PARTITIONS

AIM: Write a program in Scilab for the following:

Q.1 Let the set S has 8 elements. Find the number of ordered partitions of S into 3 cells with 2, 2 and 3 elements.

SOURCE CODE:

```
n=8;//total number of elements in S
//number of elements in partition cells
n1=2;
n2=2;
n3=3;
//finding number of ordered partitions
ans=factorial(n)/(factorial(n1)*factorial(n2)*factorial(n3));
disp(ans,'Number of ordered partitions is:');
```

OUTPUT:

```
exec('E:\Scilab program files\orderd_partitions.sce');

 Number of ordered partitions is:

    1680.
```

PRACTICAL No. 3 (J)

UNORDERED PARTITIONS

AIM: Write a program in Scilab for the following:

Q.1 Let the set S has 8 elements. Find the number of unordered partitions of S into 3 cells with 2, 2 and 3 elements.

Q.2 Find the number of ways that 10 people can be partitioned into 4 committees, such that 2 committees contain 3 people and the other 2 committees contain 2 people.

(i) SOURCE CODE:

```
n=8;//total number of elements in S
//number of elements in partition cells
n1=2;
n2=2;
n3=3;
k1=2;//number of cells have 2 elements
k2=1;//number of cells have 3 elements
//finding number of ordered partitions
op=factorial(n)/(factorial(n1)*factorial(n2)*factorial(n3));
//finding unordered partition
ans=op/(factorial(k1)*factorial(k2));
disp(ans,'Number of unordered partitions is:');
```

OUTPUT:

```
exec('E:\Scilab program files\unordered_partitions1.sce');
 Number of unordered partitions is:
   840.
```

(ii) SOURCE CODE:

```
n=10;//total number of people
//2 committees contain 3 people and the other 2 committees contain 2
people
n1=3;
n2=3;
n3=2;
n4=2;
k1=2;//number of cells have 2 elements
k2=2;//number of cells have 3 elements
//finding ordered partions first
op=factorial(n)/(factorial(n1)*factorial(n2)*factorial(n3)*factorial(n4));
//finding unordered partition
ans=op/(factorial(k1)*factorial(k2));
disp(ans,'Number of ways that 10 people can be partitioned into 6
committees, such that 2 committees contain 3 people and the other
2 committees contain 2 people is:');
```

OUTPUT:

```
exec('E:\Scilab program files\unorderd_partitions.sce');

 Number of ways that 10 people can be partitioned into 6 committees,
        such that 2 committees contain 3 people and the other 2
        committees contain 2 people is:

 6300.
```

PRACTICAL No. 4 (A)

SAMPLE SPACE AND EVENTS

AIM: Write a program in Scilab for the following:

Q.1 Suppose an uniform dice is rolled and let A be an event that an even number appears, B be an event that an odd number appears and C be an event that a prime number appears. Then find the events $A \cup C, B \cap C, C^c$. Also verify that the events A and B are mutually exclusive events.

Q.2 Let a fair coin is tossed three times and A be an event that at least two head appears and B be an event that all the tosses are same. Then find the events $A \cup B$ *and* $A \cap B$.

SOURCE CODE:

```
S1=[1,2,3,4,5,6];//sample space for the rolling of a dice
disp(S1,'Sample splace for dice throw experiment is:')
A=[2,4,6];//event that an even number occurs
disp(A,'Event A of getting even number is:');
B=[1,3,5];//event that an odd number occurs
disp(B,'Event B of getting odd number is:');
C=[2,3,5];//event that a prime number occurs
disp(C,'Event C of getting prime number is:');
disp(union(A,C),'The event that an even or a prime number occurs:');
disp(intersect(B,C),'The event that an odd prime number occurs:');
disp(setdiff(S1,C),'The event that a prime number does not occur:');//It
is the complement of the set C.
if intersect(A,B)==[] then
   disp('Events A nad B are mutually exclusive events.');
else
   disp('Events A and B are not mutually exclusive events.');
end
S2=["HHH","HHT","HTH","HTT","THH","THT","TTH","TTT"];//sample
space for the toss of a coin three times
disp(S2,'Sample space for coin toss experiment is:');
P=["HHH","HHT","HTH","THH"];//event that at least two heads appear
disp(P,'Event A of getting at least two heads is:')
Q=["HHH","TTT"];//event that all tosses are the same
disp(Q,'Event B of all tosses result in same is:');
disp(union(P,Q),'Union of A and B event:');
disp(intersect(P,Q),'Intersection of A and B event:');
```

OUTPUT:

```
exec('E:\Scilab program files\events.sce');
 Sample space for dice throw experiment is:
   1.    2.    3.    4.    5.    6.
 Event A of getting even number is:
   2.    4.    6.
 Event B of getting odd number is:
   1.    3.    5.
 Event C of getting prime number is:
   2.    3.    5.
 The event that an even or a prime number occurs:
   2.    3.    4.    5.    6.
 The event that an odd prime number occurs:
   3.    5.
 The event that a prime number does not occur:
   1.    4.    6.
Events A nad B are mutually exclusive events.

 Sample space for coin toss experiment is:
!HHH  HHT  HTH  HTT  THH  THT  TTH  TTT  !
 Event A of getting at least two heads is:
!HHH  HHT  HTH  THH  !
 Event B of all tosses result in same is:
!HHH  TTT  !
 Union of A and B event:
!HHH  HHT  HTH  THH  TTT  !
 Intersection of A and B event:
 HHH
```

PRACTICAL No. 4 (B)

FINITE PROBABILITY SPACES

AIM: Write a program in Scilab for the following:

Q.1 Two uniform dice are rolled. Find the probability of
(i) an event A that sum of the numbers is at most 6.
(ii) an event B that sum of the numbers is at least 10.

SOURCE CODE:

```
disp('Two uniform dice are rolled and sum of the outcomes are
observed:');
S=[2,3,4,5,6,7,8,9,10,11,12];//all possible outcomes for the experiment
disp('The probability space is as follows:');
//Pn = probability of getting n as sum of the outcomes
P2=1/36, P3=2/36, P4=3/36, P5=4/36, P6=5/36, P7=6/36, P8=5/36,
P9=4/36, P10=3/36, P11=2/36, P12=1/36
disp('A is the event that sum of the outcomes is at most 6 and B is the
event that sum of the outcomes is at least 10.');
PA=P2+P3+P4+P5+P6;//probability of event A
disp(PA,'Probability of event A:');
PB=P10+P11+P12;//probability of event B
disp(PB,'Probability of event B:');
```

OUTPUT:

```
exec('E:\Scilab program files\probability_spaces.sce');
 Two uniform dice are rolled and sum of the outcomes are observed:
 The probability space is as follows:
 P2  =
   0.0277778
 P3  =
   0.0555556
 P4  =
   0.0833333
 P5  =
   0.1111111
```

P6 =

0.1388889

P7 =

0.1666667

P8 =

0.1388889

P9 =

0.1111111

P10 =

0.0833333

P11 =

0.0555556

P12 =

0.0277778

A is the event that sum of the outcomes is at most 6 and B is the event that sum of the outcomes is at least 10.

Probability of event A:

0.4166667

Probability of event B:

0.1666667

PRACTICAL No. 4 (C)

EQUIPROBABLE SPACES

Q.1 Suppose an uniform die is rolled and let A be an event that an even number appears, B be an event that an odd number appears and C be an event that a prime number appears. Then find the probability of the events $A, B, C, A \cup C, A \cup B, B \cap C, C^c$.

SOURCE CODE:

```
S=[1,2,3,4,5,6];//sample space for the rolling of a dice
A=[2,4,6];//event that an even number occurs
B=[1,3,5];//event that an odd number occurs
C=[2,3,5];//event that a prime number occurs
AorC=union(A,C);//event that an even or a prime number occurs
AorB=union(A,B);//even that an even or an odd number occurs
BandC=intersect(B,C);//event that an odd prime number occurs
complimentC=setdiff(S,C);//event that a prime number does not occur
PA=length(A)/length(S);
disp(PA,'Probability of occurrence of event A:');
PB=length(B)/length(S);
disp(PB,'Probability of occurrence of event B:');
PC=length(C)/length(S);
disp(PC,'Probability of occurrence of event C:');
PAorC=length(AorC)/length(S);
disp(PAorC,'Probability of the event that an even or a prime number
occurs:');
PAorB=length(AorB)/length(S);
disp(PAorB,'Probability of the event that an even or an odd number
occurs:');
PBandC=length(BandC)/length(S);
disp(PBandC,'Probability of the event that a prime odd number
occurs:');
PcomplimentC=length(complimentC)/length(S);
disp(PcomplimentC,'Probability of the event that prime number does
not occur:');
```

OUTPUT:

```
exec('E:\Scilab program files\equiprobable_spaces.sce');
 Probability of occurrence of event A:
   0.5
 Probability of occurrence of event B:
   0.5
 Probability of occurrence of event C:
   0.5
 Probability of the event that an even or a prime number occurs:
   0.8333333
 Probability of the event that an even or an odd number occurs:
   1.
 Probability of the event that a prime odd number occurs:
   0.3333333
 Probability of the event that prime number does not occur:
   0.5
```

PRACTICAL No. 4 (D)

ADDITION PRINCIPLE

AIM: Write a program in Scilab for the following:

Q.1 Suppose a person is selected at random from a group of 100 people where 45 likes to eat Mango, 35 likes to eat Orange, and 15 likes to eat both Mango and Oranges. Find the probability that the selected person likes Mango or Orange.

SOURCE CODE:

```
disp("Experiment: selection of a person out of 100 people ")
T=100;//total no. of people
M=45;//no of people likes to eat Mango
O=35;//no of people likes to eat Orange
MandO=15;//no of people likes both Mango and Orange
PM=M/T;//probability of the selected person likes Mango
disp(PM,'Probability of people like Mango:');
PO=O/T;//probability of the selected person likes Orange
disp(PO,'Probability of people like Orange:');
PMandO=MandO/T;//probability of the selected person like both
Mango nad Orange
disp(PMandO,'Probability of people like both Mango and Orange:');
PMorO=PM+PO-PMandO;//addition rule
disp(PMorO,'Probability of the selected person likes Mango or
Orange:');
```

OUTPUT:

```
exec('E:\Scilab program files\addtion_rule.sce');

 Experiment: selection of a person out of 100 people

 Probability of people like Mango:
   0.45

 Probability of people like Orange:
   0.35

 Probability of people like both Mango and Orange:
   0.15

 Probability of the selected person likes Mango or Orange:
   0.65
```

PRACTICAL No. 4 (E)

CONDITIONAL PROBABILITY

AIM: Write a program in Scilab for the following:

Q.1 A pair of fair dice is thrown. Find the probability that the sum is 10 or greater if:
(*a*) 5 appears on the first die; (*b*) 5 appears on at least one die.

SOURCE CODE:

```
disp('Experiment: Two uniform dice are rolled and sum of the outcomes are observed');
//To find: probability of an event that the sum of the outcomes are 10 or greater if 5 appears on the first die
E1=['(5,1)','(5,2)','(5,3)','(5,4)','(5,5)','(5,6)'];//event that 5 appears on first die
disp(E1,'Event that 5 appears on first die:');
A=['(4,6)','(5,5)','(5,6)','(6,4)','(6,5)','(6,6)'];//event that sum of the outcomes are 10 or greater
disp(A,'Event that sum of the outcomes are 10 or greater:');
AandE1=intersect(A,E1);//event that 5 appears on the first die and sum of the outcomes are 10 or greater
disp(AandE1,'Event that 5 appears on the first die and sum of the outcomes are 10 or greater:');
//P(A/E)=n(AnE)/n(E)
P1=2/6;//probability of event that sum of the outcomes are 10 or greater if 5 appears on the first die
disp(P1,'Probability of event that sum of the outcomes are 10 or greater if 5 appears on the first die:');
//To find: probability of an event that the sum of the outcomes are 10 or greater if 5 appears on at least one die
E2=['(5,1)','(5,2)','(5,3)','(5,4)','(5,5)','(5,6)','(1,5)','(2,5)','(3,5)','(4,5)','(6,5)']
;//event that 5 appears on at least die
disp(E2,'Event that 5 appears on at least one die:');
AandE2=intersect(A,E2);//event that 5 appears on at least one die and sum of the outcomes are 10 or greater
disp(AandE2,'Event that 5 appears on at least one die and sum of the outcomes are 10 or greater:');
P2=3/11;//probability of event that sum of the outcomes are 10 or greater if 5 appears on at least one die
disp(P2,'Probability of event that sum of the outcomes are 10 or greater if 5 appears on at least one die:');
```

OUTPUT:

exec('E:\Scilab program files\conditional_prob.sce');

Experiment: Two uniform dice are rolled and sum of the outcomes are observed

Event that 5 appears on first die:

!(5,1) (5,2) (5,3) (5,4) (5,5) (5,6) !

Event that sum of the outcomes are 10 or greater:

!(4,6) (5,5) (5,6) (6,4) (6,5) (6,6) !

Event that 5 appears on the first die and sum of the outcomes are 10 or greater:

!(5,5) (5,6) !

Probability of event that sum of the outcomes are 10 or greater if 5 appears on the first die:

0.3333333

Event that 5 appears on at least one die:

!(5,1) (5,2) (5,3) (5,4) (5,5) (5,6) (1,5) (2,5) (3,5) (4,5) (6,5) !

Event that 5 appears on at least one die and sum of the outcomes are 10 or greater:

!(5,5) (5,6) (6,5) !

Probability of event that sum of the outcomes are 10 or greater if 5 appears on at least one die:

0.2727273

Q.2 In a certain college town, 35% of the students failed mathematics, 25% failed programming, and 15% failed both mathematics and programming. A student is selected at random.
(*a*) If he failed programming, find the probability that he also failed mathematics.
(*b*) If he failed mathematics, find the probability that he also failed programming.
(*c*) Find the probability that he failed mathematics or programming.
(*d*) Find the probability that he failed neither mathematics nor programming.

SOURCE CODE:

```
PM=0.35;//probability that a student failed in mathematics
disp(PM,'Probability that student failed in mathematics:');
PP=0.25;//probability that a student failed in programming
disp(PP,'Probability that student failed in programming:');
PMnP=0.15;//probability that a student failed in mathematics and
programming
disp(PM,'Probability that student failed in mathematics and
programming:');
//P(A/E)=P(AnE)/P(E)
PMifP=PMnP/PP;//probability that a student failed in mathematics if
he failed in programming
disp(PMifP,'Probability that a student failed in mathematics if he failed
in programming:');
PPifM=PMnP/PM;//probability that a student failed in programming if
he failed in mathematics
disp(PPifM,'Probability that a student failed in programming if he
failed in mathematics:');
//finding probability that a student failed in mathematics or
programming
PMorP=PM+PP-PMnP;//addition rule
disp(PMorP,'Probability that a student failed in mathematics or
programming:');
PNeither=1-PMorP;//probability that a student failed in neither
mathematics nor programming
disp(PNeither,'Probability that a student failed in neither mathematics
nor programming:');
```

OUTPUT:

```
exec('E:\Scilab program files\conditional_prob2.sce');
 Probability that student failed in mathematics:
    0.35
 Probability that student failed in programming:
    0.25
 Probability that student failed in mathematics and programming:
    0.35
 Probability that a student failed in mathematics if he failed in
programming:
    0.6
 Probability that a student failed in programming if he failed in
mathematics:
    0.4285714
 Probability that a student failed in mathematics or programming:
    0.45
 Probability that a student failed in neither mathematics nor
programming:
    0.55
```

PRACTICAL No. 4 (F)

MULTIPLICATION PRINCIPLE FOR CONDITIONAL PROBABILITY

AIM: Write a programme in Scilab for the following:

Q.1 A class has 14 boys and 6 girls. Suppose 4 students are selected at random from the class. Find the probability that they are all boys.

SOURCE CODE:

```
t=20;//total no. of students in class
b=14;//no. of boys in class
P1=b/t;//probability that first selected student is a boy
P2=(b-1)/(t-1);//probability that second selected student is a
boy
P3=(b-2)/(t-2);//probability that third selected student is a
boy
P4=(b-3)/(t-3);//probability that forth selected student is a
boy
P=P1*P2*P3*P4;
disp(P,'Probability that all four selected students are boy:');
```

OUTPUT:

```
exec('E:\Scilab program files\multi_conditional_prob.sce');

 Probability that all four selected students are boy:

   0.2066047
```

PRACTICAL No. 4 (G)

INDEPENDENT EVENTS

AIM: Write a program in Scilab for the following:

Q.1 Consider a family with three children and let A be an event that first child is a boy, B be an event that second child is a boy and C be an event that two successive children are boy. Find which of the two events are independent events.

SOURCE CODE:

```
B=1;//boy child
G=2;//girl child
S=[111,112,121,122,211,212,221,222];//sample space
A=[111,112,121,122];//event that first child is boy
B=[111,112,211,212];//event that second child is boy
C=[111,112,211];//event that two successive children are boy
PA=length(A)/length(S);
disp(PA,'Probability of A is:');
PB=length(B)/length(S);
disp(PB,'Probability of B is:');
PC=length(C)/length(S);
disp(PC,'Probability of C is:');
AandB=intersect(A,B);
AandC=intersect(A,C);
BandC=intersect(B,C);
PAandB=length(AandB)/length(S);
disp(PAandB,'Probability of the event AnB is:');
PAandC=length(AandC)/length(S);
disp(PAandC,'Probability of the event AnC is:');
PBandC=length(BandC)/length(S);
disp(PBandC,'Probability of the event BnC is:');
if((PA*PB)==PAandB)
 disp('A and B are independent.');
else
 disp('A and B are dependent.');
end
if((PA*PC)==PAandC)
  disp('A and C are independent.');
else
 disp('A and C are dependent.');
end
if((PB*PC)==PBandC)
  disp('B and C are independent.');
else
  disp('B and C are dependent.');
end
```

OUTPUT:

```
exec('E:\Scilab program files\independent_events.sce');
 Probability of A is:
    0.5
 Probability of B is:
    0.5
 Probability of C is:
    0.375
 Probability of the event AnB is:
    0.25
 Probability of the event AnC is:
    0.25
 Probability of the event BnC is:
    0.375
 A and B are independent.
 A and C are dependent.
 B and C are dependent.
```

Q.2 The probability that *A* hits a target is 1/3 and the probability that *B* hits a target is 1/5. They both fire at the target. Find the probability that:

(*a*) *A* does not hit the target; (*c*) one of them hits the target;
(*b*) both hit the target; (*d*) neither hits the target.

SOURCE CODE:

```
PA=1/3;//probability that A hits the target
PB=1/5;//probability that B hits the target
PComplimentA=1-PA;//probability that A does not hit the target
disp(PComplimentA,'Probability that A does not hit the target:');
//As both the events are independent P(AnB)=P(A)P(B)
PAandB=PA*PB;//probability that both A nad B hits the target
disp(PAandB,'Probability that both A nad B hits the target:');
//by addition rule
PAorB=PA+PB-PAandB;//probability that one of them hits the target
disp(PAorB,'Probability that one of them hits the target:');
PNeither=1-PAorB;//probability that neither hits the target
disp(PNeither,'Probability that neither hits the target:');
```

OUTPUT:

```
exec('E:\Scilab program files\independent_events2.sce');
 Probability that A does not hit the target:
    0.6666667
 Probability that both A nad B hits the target:
    0.0666667
 Probability that one of them hits the target:
    0.4666667
 Probability that neither hits the target:
    0.5333333
```

PRACTICAL No. 4 (H)
INDEPENDENT REPEATED TRIALS

AIM: Write a program in Scilab for the following:

Q. Whenever three players viz. A, B and C, play together, their respective probability of winning are 0.5, 0.3 and 0.2. They played two matches.

(i) Describe the probability space S_2.

(ii) Find the probability that the same player wins both the matches.

SOURCE CODE:

```
P1=0.5;//probability that player A wins the match
P2=0.3;//probability that player B wins the match
P3=0.2;//probability that player C wins the match
S2=[11,12,13,21,22,23,31,32,33];//sample space S2 where 11 means player A wins both mathches and 12 means player A wins first match and player B wins second match etc
P11=P1*P1;//probability that player A wins both the matches
disp(P11,'Probability that player A wins both the matches:');
P12=P1*P2;//probability that player A wins first match and player B wins second match
disp(P12,'Probability that player A wins first match and player B wins second match:');
P13=P1*P3;//probability that player A wins first match and player C wins second match
disp(P13,'Probability that player A wins first match and player C wins second match:');
P21=P2*P1;//probability that player B wins first match and player A wins second match
disp(P21,'Probability that player B wins first match and player A wins second match:');
P22=P2*P2;//probability that player B wins both the matches
disp(P22,'Probability that player B wins both the matches:');
P23=P2*P3;//probability that player B wins first match and player C wins second match
disp(P23,'Probability that player B wins first match and player C wins second match:');
P31=P3*P1;//probability that player C wins first match and player A wins second match
disp(P31,'Probability that player C wins first match and player A wins second match:');
P32=P3*P2;//probability that player C wins first match and player B wins second match
disp(P32,'Probability that player C wins first match and player B wins second match:');
P33=P3*P3;//probability that player C wins both the matches
disp(P33,'Probability that player C wins both the matches:');
disp(P11+P22+P33,'Probability that same player wins both matches:');
```

OUTPUT:

```
exec('E:\Scilab program files\independent_repeated_trials.sce');
 Probability that player A wins both the matches:
   0.25
 Probability that player A wins first match and player B wins second match:
   0.15
 Probability that player A wins first match and player C wins second match:
   0.1
 Probability that player B wins first match and player A wins second match:
   0.15
 Probability that player B wins both the matches:
   0.09
 Probability that player B wins first match and player C wins second match:
   0.06
 Probability that player C wins first match and player A wins second match:
   0.1
 Probability that player C wins first match and player B wins second match:
   0.06
 Probability that player C wins both the matches:
   0.04
 Probability that same player wins both matches:
   0.38
```

PRACTICAL No. 4 (I)

REPEATED TRIALS WITH TWO OUTCOMES

AIM: Write a program in Scilab for the following:

Q. The probability that Virat hits a target is $p = 1/4$. He fires 6 times. Find the probability that he hits the target: (*a*) exactly two times; (*b*) more than four times; (*c*) at least once.

SOURCE CODE:

```
n=6;//number of trials
p=1/4;//probability of success in a trial
q=1-p;//probability of failure
//to find probability that Virat hits the target exactly two times
r2=2;//number of success
//P[X=r]=nCr*p^r*q^(n-r)
P2=(factorial(n)/(factorial(r2)*factorial(n-r2)))*(p^r2)*(q^(n-r2));
disp(P2,'Probability that Virat hits the target exactly two times:');
//to find probability that Virat hits the target more than four times
r5=5, r6=6;
P5=(factorial(n)/(factorial(r5)*factorial(n-r5)))*(p^r5)*(q^(n-r5));
P6=(factorial(n)/(factorial(r6)*factorial(n-r6)))*(p^r6)*(q^(n-r6));
ans2=P5+P6;
disp(ans2,'Probability that Virat hits the target more than four
times:');
//to find probability that Virat hits the target atleast once
//calculation by taking compliment event
r0=0;
P0=(factorial(n)/(factorial(r0)*factorial(n-r0)))*(p^r0)*(q^(n-r0));
ans3=1-P0;
disp(ans3,'Probability that Virat hits the target at least once:');
```

OUTPUT:

```
exec('E:\Scilab program files\repeated_trials_with_two_outcomes.sce');
```

Probability that Virat hits the target exactly two times:

0.2966309

Probability that Virat hits the target more than four times:

0.0046387

Probability that Virat hits the target at least once:

0.8220215

PRACTICAL No. 5 (A)

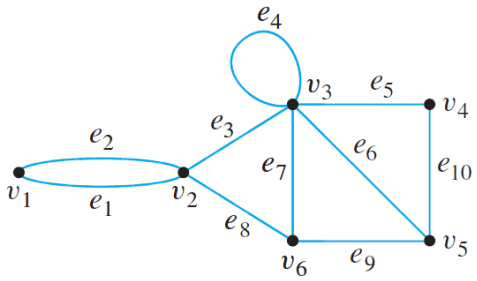
PATHS & CONNECTIVITY

AIM: Write a program Scilab for the following:

Q. In the graph below, determine which of the following walks are trails, paths, circuits, or simple circuits.

a. $v_1e_1v_2e_3v_3e_4v_3e_5v_4$ b. $e_1e_3e_5e_5e_6$ c. $v_2v_3v_4v_5v_3v_6v_2$

d. $v_2v_3v_4v_5v_6v_2$ e. $v_1e_1v_2e_1v_2$ f. v_1



SOURCE CODE:

```
A=[0 1 0 0 0 0;1 0 1 0 0 0;0 1 1 1 0 0;0 0 1 0 0 0;0 0 0 0 0 0;0 0 0 0 0 0];
disp(A,'Adjacency matrix of A is:');
disp('This walk has a repeated vertex but does not have a repeated edge, so it is a trail from v1 to v4 but not a path.');
B=[0 1 0 0 0 0;1 0 0 1 0 0;0 0 0 1 0 0;0 1 1 0 0 0;0 0 1 0 0 0;0 0 0 0 0 0];
disp(B,'Adjacency matrix of B is:');
disp('This is just a walk from v1 to v5. It is not a trail because it has a repeated edge.');
C=[0 0 0 0 0 0;0 0 1 0 0 1;0 1 0 1 1 1;0 0 1 0 1 0;0 0 1 1 0 0;0 1 1 0 0 0];
disp(C,'Adjacency matrix of C is:');
disp('This walk starts and ends at v2, and does not have a repeated edge, so it is a circuit. Since the vertex v3 is repeated in the middle, it is not a simple circuit.');
D=[0 0 0 0 0 0;0 0 1 0 0 1;0 1 0 1 0 0;0 0 1 0 1 0;0 0 0 1 0 1;0 1 0 0 1 0];
disp(D,'Adjacency matrix of D is:');
disp('This walk starts and ends at v2, does not have a repeated edge, and does not have a repeated vertex. Thus it is a simple circuit.');
E=[0 1 0 0 0 0;1 0 0 0 0 0;0 0 0 0 0 0;0 0 0 0 0 0;0 0 0 0 0 0;0 0 0 0 0 0];
disp(E,'Adjacency matrix of E is:');
disp('This is just a closed walk starting and ending at v1. It is not a circuit because edge e1 is repeated.');
F=[0 0 0 0 0 0;0 0 0 0 0 0;0 0 0 0 0 0;0 0 0 0 0 0;0 0 0 0 0 0;0 0 0 0 0 0];
disp(F,'Adjacency matrix of F is:');
disp('The first vertex of this walk is the same as its last vertex, but it does not contain an edge, and so it is not a circuit. It is a closed walk from v1 to v1.');
```

OUTPUT:

exec('E:\Scilab program files\graph_paths.sce');

Adjacency matrix of A is:

0.	1.	0.	0.	0.	0.
1.	0.	1.	0.	0.	0.
0.	1.	1.	1.	0.	0.
0.	0.	1.	0.	0.	0.
0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.

This walk has a repeated vertex but does not have a repeated edge, so it is a trail from v1 to v4 but not a path.

Adjacency matrix of B is:

0.	1.	0.	0.	0.	0.
1.	0.	0.	1.	0.	0.
0.	0.	0.	1.	0.	0.
0.	1.	1.	0.	0.	0.
0.	0.	1.	0.	0.	0.
0.	0.	0.	0.	0.	0.

This is just a walk from v1 to v5. It is not a trail because it has a repeated edge.

Adjacency matrix of C is:

0.	0.	0.	0.	0.	0.
0.	0.	1.	0.	0.	1.
0.	1.	0.	1.	1.	1.
0.	0.	1.	0.	1.	0.
0.	0.	1.	1.	0.	0.
0.	1.	1.	0.	0.	0.

This walk starts and ends at v2, and does not have a repeated edge, so it is a circuit. Since the vertex v3 is repeated in the middle, it is not a simple circuit.

Adjacency matrix of D is:

0.	0.	0.	0.	0.	0.
0.	0.	1.	0.	0.	1.
0.	1.	0.	1.	0.	0.
0.	0.	1.	0.	1.	0.
0.	0.	0.	1.	0.	1.
0.	1.	0.	0.	1.	0.

This walk starts and ends at v2, does not have a repeated edge, and does not have a repeated vertex. Thus it is a simple circuit.

Adjacency matrix of E is:

0.	1.	0.	0.	0.	0.
1.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.

This is just a closed walk starting and ending at v1. It is not a circuit because edge e1 is repeated.

Adjacency matrix of F is:

0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.

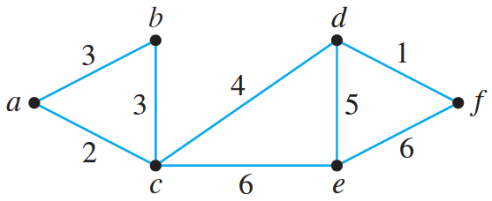
The first vertex of this walk is the same as its last vertex, but it does not contain an edge, and so it is not a circuit. It is a closed walk from v1 to v1.

PRACTICAL No. 5 (B)

MINIMUM SPANNING TREE

AIM: Write a program Scilab for the following:

Q. Find minimum spanning tree for the following graph by using Kruskal's algorithm.



SOURCE CODE:

```
//M=[0 3 2 0 0 0;3 0 3 0 0 0;2 3 0 4 6 0;0 0 4 0 5 1;0 0 6 5 0 6;0 0 0 1
6 0];
//disp(M,'Adjacency matrix of given weigthed graph is:');
disp('Edges of the graph are:');
AB=3
AC=2
BC=3
CD=4
CE=6
DE=5
DF=1
EF=6
W=[AB,AC,BC,CD,CE,DE,DF,EF];
V=int32(M);
L=gsort(V);    //sorting edges in decreasing order of their weights
N=[DF,AC,AB,CD,DE];    //edges in minimum spanning tree
disp(N,'Weights of edges in minimum spanning tree:');
k=sum(N);
disp(k,'Weight of the minimum spanning tree is:');
disp('Finding minimum spanning tree by another method:');
H=gsort(V,'g','i');    //sorting edges in decreasing order of their weights
N2=[DE,CD,BC,AC,DF];
disp(N2,'Weights of edges in minimum spanning tree:');
k2=sum(N2);
disp(k2,'Weight of the minimum spanning tree is:');
```

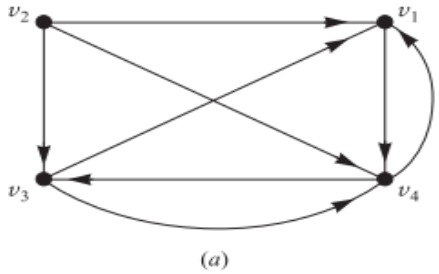
OUTPUT:

```
exec('E:\Scilab program files\spanning_tree.sce');

 Edges of the graph are:
AB  =
   3.
AC  =
   2.
BC  =
   3.
CD  =
   4.
CE  =
   6.
DE  =
   5.
DF  =
   1.
EF  =
   6.

Weights of edges in minimum spanning tree:

   1.    2.    3.    4.    5.

 Weight of the minimum spanning tree is:

   15.

 Finding minimum spanning tree by another method:

 Weights of edges in minimum spanning tree:

   5.    4.    3.    2.    1.

 Weight of the minimum spanning tree is:

   15.
```

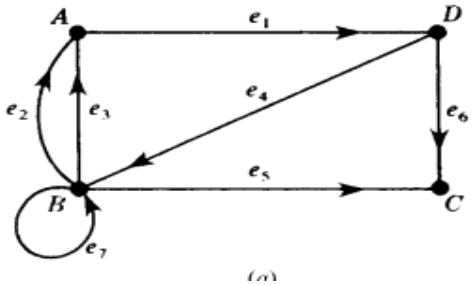
PRACTICAL No. 6 (A)

ADJACENCY MATRIX

AIM: Write a program in Scilab to display adjacency matrix of the following digraph and number of edges.



(a)



(a)

SOURCE CODE:

```
A=[0 0 0 1;1 0 1 1;1 0 0 1;1 0 1 0];
disp(A,'Adjacency matrix of first digraph is:');
k=sum(A);
disp(k,'Number of edges in first digraph is:');
B=[0 0 0 1;2 1 1 0;0 0 0 0;0 1 1 0];
disp(B,'Adjacency matrix of second digraph is:');
n=sum(B);
disp(n,'Number of edges in second digraph is:');
```

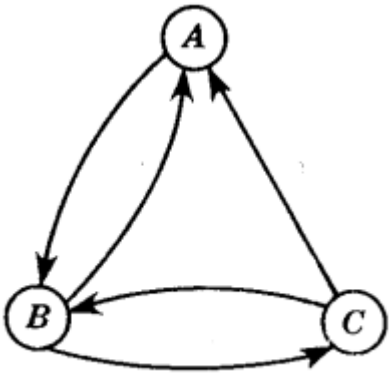
OUTPUT:

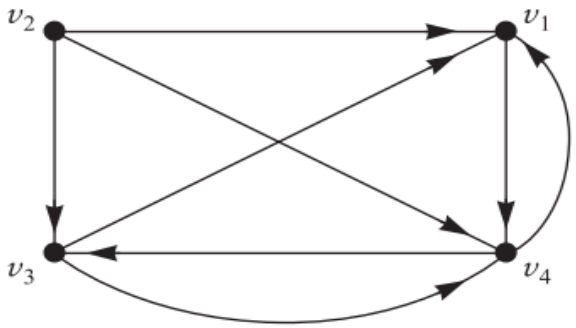
```
exec('E:\Scilab program files\adjacency.sce');
Adjacency matrix of first digraph is:
   0.    0.    0.    1.
   1.    0.    1.    1.
   1.    0.    0.    1.
   1.    0.    1.    0.
 Number of edges in first digraph is:
   8.
  Adjacency matrix of second digraph is:
   0.    0.    0.    1.
   2.    1.    1.    0.
   0.    0.    0.    0.
   0.    1.    1.    0.
 Number of edges in second digraph is:
   7.
```

PRACTICAL No. 6 (B)

PATH MATRIX

AIM: Write a program in Scilab to compute and display path (reachability) matrix and to check whether the following digraphs are strongly connected:





(i) SOURCE CODE:

```
A=[0 1 0 ;1 0 1 ;1 1 0];
disp(A,'Adjacency matrix of first digraph G is:');
B3=A+A^2+A^3;
disp(B3,'B3 matrix for the given digraph is:');
k=0;
for i=1:9
    if B3(i)~=0 then
        B3(i)=1;
    end
end
disp(B3,'Path (reachability) matrix P for given digraph is:');
for i=1:9
    if B3(i)==0 then
        k=1;
    end
end
if k==1 then
    disp('There are zero entries in path matrix P, therefore the
digraph is not strongly connected.');
else
    disp('There is no zero entry in path matrix P, therefore the
digraph is strongly connected.');
end
```

OUTPUT:

exec('E:\Scilab program files\path_matrix2.sce');

Adjacency matrix of first digraph is:

0. 1. 0.

1. 0. 1.

1. 1. 0.

B3 matrix for the given digraph is:

2. 3. 1.

4. 3. 3.

4. 4. 2.

Path (reachability) matrix P for given digraph is:

1. 1. 1.

1. 1. 1.

1. 1. 1.

There is no zero entry in path matrix P, therefore the digraph is strongly connected.

(ii) SOURCE CODE:

```
A=[0 0 0 1;1 0 1 1;1 0 0 1;1 0 1 0];
disp(A,'Adjacency matrix of second digraph is:');
B4=A+A^2+A^3+A^4;
disp(B4,'B4 matrix for the given digraph is:');
k=0;
for i=1:16
   if B4(i)~=0 then
      B4(i)=1;
   end
end
disp(B4,'Path (reachability) matrix P for given digraph is:');
for i=1:16
   if B4(i)==0 then
      k=1;
   end
end
if k==1 then
```

```
    disp('There are zero entries in path matrix P, therefore the
digraph is not strongly connected.');
else
    disp('There is no zero entry in path matrix P, therefore the
digraph is strongly connected.');
end
```

OUTPUT:

exec('E:\Scilab program files\path_matrix.sce');

Adjacency matrix of second digraph is:

0. 0. 0. 1.

1. 0. 1. 1.

1. 0. 0. 1.

1. 0. 1. 0.

B4 matrix for the given digraph is:

4. 0. 3. 4.

11. 0. 7. 11.

7. 0. 4. 7.

7. 0. 4. 7.

Path (reachability) matrix P for given digraph is:

1. 0. 1. 1.

1. 0. 1. 1.

1. 0. 1. 1.

1. 0. 1. 1.

There are zero entries in path matrix P, therefore the digraph is not strongly connected.