Bachelors of Science

SEM I

Journal

Roll No.	003
Name	Sarabjeet Singh Chyot
Subject	Computational Logic and Discrete Structures

ANANDIBAI DAMODAR KALE SHAIKSHANIK SANSTHA'S DEGREE ANANDIBAI DAMODAR KALE COLLEGE OF ARTS & COMMERCE

Saibaba Nagar, Borivali (West), Mumbai – 400 092

Tel.: 2807 7126

Email: adkdcollege@yahoo.com / www.adkdcollege.in

ISO 9001: 2008 Certified

CERTIFICATE

This is here to certify that Mr/Ms. **Sarabjeet Singh**, Seat Number **003** of B.Sc. IT, has satisfactorily completed the required number of experiments prescribed by the **ANANDIBAI DAMODAR KALE DEGREE COLLEGE AFFILIATED TO UNIVERSITY OF MUMBAI** during the academic year 2022 – 2023.

Date:

Place: Mumbai

Teacher In-Charge

Head of Department

External Examiner

INDEX

Sr. No.	List of Experiments	Page No	Date	Sign
1	Set Theory **a.** Inclusion Exclusion principle. **b.** Power Sets **c.** Mathematical Induction	01-02		
2	Functions and Algorithms **a.** Recursively defined functions **b.** Cardinality **c.** Polynomial evaluation **d.** Greatest Common Divisor	03-04		
3	Probability Theory 1 **a.** Sample space and events **b.** Finite probability spaces **c.** Equiprobable spaces **d.** Addition Principle.	05-07		

4	Probability Theory 2 **a**. Conditional Probability **b**. Multiplication theorem for conditional probability **c**. Independent events d. Repeated trials with two outcomes	08-11		
5	Counting 1 **a**. Sum rule principle **b**. Product rule principle **c**. Factorial **d**. Binomial coefficients	12-15		
6	Counting 2 **a**. Permutations **b**. Permutations with repetitions **c**. Combinations **d**. Combinations with repetitions	16-18		
7	Graph Theory **a**. Paths and connectivity **b**. Minimum spanning tree **c**. Isomorphism	19		
8	Graph Theory **a**. Paths and connectivity **b**. Minimum spanning tree **c**. Isomorphism	20-23		
9	Directed Graphs **a**. Adjacency matrix **b**. Path matrix	24-25		
10	Recurrence relations **a**. Linear homogeneous recurrence relations with constant coefficients **b**. Solving linear homogeneous recurrence relations with constant coefficients **c**. Solving general homogeneous linear recurrence relations	26-29		

Experiment no – 01

Aim: Set Theory

a. Inclusion Exclusion principle.

b. Power Sets

c. Mathematical Induction

Inclusion Exclusion principle:

CODE :

disp ('To find:number of mathematics students taking atleast one of the lamguages French(F),German(G) and Russian(R)')

F =65; // number o f s t u d e n t s s t u d y i n g French

G =45; // number o f s t u d e n t s s t u d y i n g German

R =42; // number o f s t u d e n t s s t u d y i n g Rus s ian FandG =20; //

number o f s t u d e n t s s t u d y i n g French and German FandR =25; //

number o f s t u d e n t s s t u d y i n g French and Russian GandR =15; //

number o f s t u d e n t s s t u d y i n g German and Russian FandGandR =18; //

number o f s t u d e n t s s t u d y i n g French , German and Russian //By i n c l u s i o n - E x c l u s i o n p r i n c i p l e

ForGorR =F+G+R-FandG -FandR - GandR + FandGandR ; disp (ForGorR , ' the number of students studying atleast one of the languages : ')

Output:

```
"To find:number of mathematics students taking atleast one of the lamguages French(F),German(G) and Russian(R)"

 110.

" the number of students studying atleast one of the languages : "
```

Power Sets:

Code 1:

x =3; // number o f members o f s e t X

P =2^ x // number o f members o f the power s e t o f X

q=P -1; // x i t s e l f i s not the pr ope r s u b s e t . Hence i t i sn ' t count ed disp (q, ' number o f members o f powe r s e t P which a r e pr ope r s u b s e t s o f x a r e : ')

Output:

```
 7.

" number o f members o f powerset P which are proper subsets of x are : "
```

CODE 2:

A=[1 ,2 ,3,4,5]; // e a t a b l e s f o r s a l a d p r e p a r a t i o n 1= onion ,2=tomato ,3= c a r r o t ,4=cabbage ,5=cucumber

p= length (A); // t o t a l number o f e a t a b l e s a v a i l a b l e

n=2^p-1; // no s a l a d can be made wi thout a t l e a s t one o f the e a t a b l e s . Hence n u l l s e t i sn ' t count ed disp (n,'number of different salads that can be prepared using the given eatables ')

Output:

```
--> exec('C:\Users\user\Desktop\clds\lst.sce', -1)

  31.

 "number of different salads that can be prepared using the given eatables"
```

Mathematical Induction:

Code:

U1 =1; // g i v e n

U2 =5; // g i v e n

P =[]; for i =1:2

P(i)=3^i -2^ i;

disp (P(i))

end disp ('P(1)=U(1) and P(2)=U(2) ');

disp ('hence Un=3^n-2^n for all n belonging to N');

Output:

```
-> exec('C:\Users\user\Desktop\clds\lst.sce', -1)

  1.

  5.

 "P(1)==U(1)andP(2)==U(2)"

 "hence Un=3^n-2^n for all n belonging to N"
```

Conclusion: Successfully performed Set Theory.

Experiment no – 02

Aim: Functions and Algorithms

a. Recursively defined functions

b. Cardinality

c. Polynomial evaluation

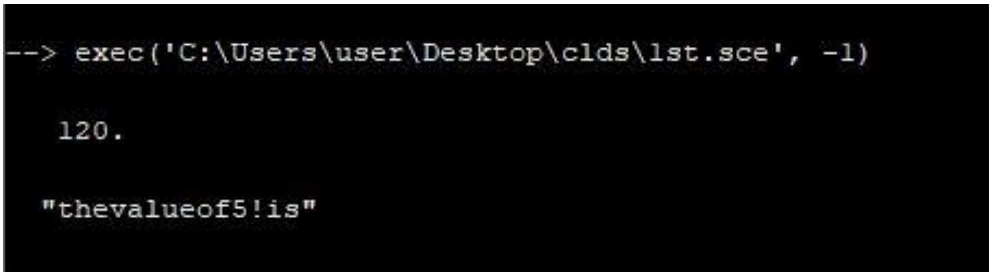
d. Greatest Common Divisor

Recursively defined functions:

Code:

```
function [k]=fact(a)
k= -1;
if(a 200) disp (" I n v a l i d ");
break ; else if(a ==1| a ==0)
k =1; else k=a* fact (a-1);
end end endfunction a =5;
p= fact (a);
disp (p, ' the v a l u e o f 5! i s ' )
```

Output:



```
--> exec('C:\Users\user\Desktop\clds\1st.sce', -1)

  120.

 "thevalueof5!is"
```

Cardinality:

Code:

```
x =1;
y =2;
z =3; A=[x,y,z];
disp ( ' c a r d i n a l i t y o f s e t A i s : ', length (A) )
B=[1 ,3 ,5 ,7 ,9]
disp ( ' c a r d i n a l i t y o f s e t B i s : ', length (B) )
```

Output:

```
--> exec('C:\Users\user\Desktop\clds\1st.sce', -1)

  "cardinalityofsetAis:"

   3.

  "cardinalityofsetBis:"

   5.
```

Polynomial evaluation:

Code:

x = poly(0,'x');

*p = 2*x^3 -7*x ^2+4*x -15;*

disp (p, ' the po l ynomi a l i s ') k= horner (p ,5);

disp (k, ' v a l u e o f the po l ynomi a l at x=5 i s ')

Output:

```
--> exec('C:\Users\user\Desktop\clds\1st.sce', -1)

  -15 +4x -7x² +2x³

  "the polynomial is "

   80.

  "value of the polynomial at x=5 is "
```

Greatest Common Divisor:

V= int32 ([258 ,60]) ;

thegcd = gcd(V);

disp (thegcd , ' the gcd o f the two numbers 258 and 60 i s ')

Output:

```
--> exec('C:\Users\user\Desktop\clds\1st.sce', -1)

  6

  "the gcd of the two numbers 258 and 60 is"
```

Conclusion: Successfully performed Functions and Algorithms.

Experiment no – 03

Aim: Probability Theory 1

a. Sample space and events

b. Finite probability spaces

c. Equiprobable spaces

d. Addition Principle

Sample space and events:

Code:

S=[1,2,3,4,5,6];

A =[2,4,6];

B =[1,3,5];

C =[2,3,5];

disp(union(A,C),'sample space for the event that an even or a prime number occurs')

disp(intersect(B,C),'sample space for the event that an odd prime number occurs')

disp(setdiff(S,C),'sample space for the event that a prime number does not occur')

intersect (A,B)

H=0;

Output:

```
--> exec('C:\Users\user\Desktop\clds\1st.sce', -1)

  2.   3.   4.   5.   6.

 "sample space for the event that an even or a prime number occurs"

  3.   5.

 "sample space for the event that an odd prime number occurs"

  1.   4.   6.

 "sample space for the event that a prime number does not occur"

 "000"

 "sample space for the event in which only heads appear"

 "Experiment: tossing a coin until a head appears and then counting the number of times the coin is tossed"

 "Since every positive integer is an element of S,the sample space is infinite"
```

Finite probability spaces:

Code:

disp ('Experiment: three coins are tossed and the number of heads are observed')

S=[0,1,2,3];

disp (" the probability space is as follows")

```
P0 =1/8;

P1 =3/8;

P2 =3/8;

P3 =1/8;

disp ("A i s the event that atleast one head appears and B is the event that all heads or all tails appear ")

A =[1 ,2 ,3];

B =[0 ,3];

PA=P1+P2+P3;

disp(PA ,'probability of occurrence of event A')

PB=P0+P3;

disp(PB,'probability of occurrence of event B')
```

Output:

```
--> exec('C:\Users\user\Desktop\clds\1st.sce', -1)

 "Experiment: three coins are tossed and the number of heads are observed"

 " the probability space is as follows"

 "A i s the event that atleast one head appears and B is the event that all heads or all tails appear "

  0.875

 "probability of occurrence of event A"

  0.25

 "probability of occurrence of event B"
```

Equiprobable spaces:

Code:

```
disp (" Experiment : a card is selected from a deck of 52 cards ")

disp ("A is the event of the selected card being a spade ")

disp ("B is the event of the selected card being a face card ")

t =52 ;

s =13;

PA= s/t;

disp (PA , 'probability of selecting a spade ' )

f =12;

PB=f/t;
```

disp (PB , ' probability of selecting a face card ')

sf =3;

Psf =sf/t;

disp (Psf ," probability of selecting a spade face card is: ")

Output:

```
--> exec('C:\Users\user\Desktop\clds\1st.sce', -1)

 " Experiment : a card is selected from a deck of 52 cards "

 "A is the event of the selected card being a spade "

 "B is the event of the selected card being a face card "

  0.25

 "probability of selecting a spade "

  0.2307692

 " probability of selecting a face card "

  0.0576923

 " probability of selecting a spade face card is: "
```

Addition Principle:

Code:

disp("Experiment: selection of a student out of 100 students")

M =30;

C =20;

T =100;

PM = M/T

PC = C/T

MnC =10;

PMnC = MnC/T

PMorC = PM+PC - PMnC ;

disp (PMorC,'probability of the selected student taking mathematics or chemistry')

Output:

```
--> exec('C:\Users\user\Desktop\clds\1st.sce', -1)

 "Experiment: selection of a student out of 100 students"

  0.4

 "probability of the selected student taking mathematics or chemistry"
```

Conclusion: Successfully performed Probability Theory 1.

Experiment no – 04

Aim: Probability Theory 2

a. Conditional Probability

b. Multiplication theorem for conditional probability

c. independent events

d. Repeated trials with two outcomes

Conditional Probability:

Code:

isp("Experiment: A die is tossed and the outcomes are observed");

disp ("To find: probability(PM) of an event that one of the dice is 2 if the sum is 6");

E=["(1,5)","(2,4)","(3,3)","(4,2)","(5,1)"]

A=["(2,1)","(2,2)","(2,3)","(2,4)","(2,5)","(2,6)","(1,2)","(3,2)","(4,2)","(5,2)","(6,2)"]

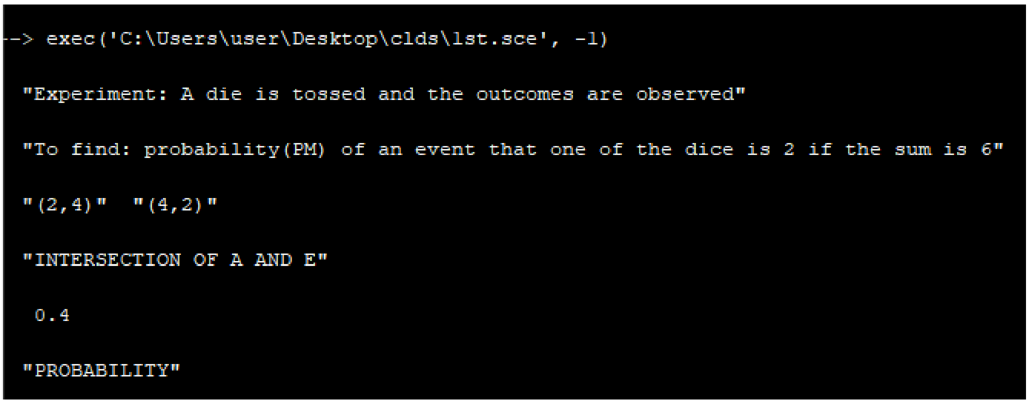
B= intersect (A,E)

PM =2/5

disp(B,'INTERSECTION OF A AND E')

disp(PM,'PROBABILITY')

Output:



```
--> exec('C:\Users\user\Desktop\clds\1st.sce', -1)

 "Experiment: A die is tossed and the outcomes are observed"

 "To find: probability(PM) of an event that one of the dice is 2 if the sum is 6"

 "(2,4)"  "(4,2)"

 "INTERSECTION OF A AND E"

  0.4

 "PROBABILITY"
```

Multiplication theorem for conditional probability:

Code:

disp("A bag contains 12 items of which four are defective.Three items are drawn at random,one after the other");

s =12;

d =4;

Pf =(s-d)/s ;

*Pe=Pf *[(s-d -1) /(s -1)]*[(s-d -2) /(s -2)];*

disp (Pe,'probability that all three items are nondefective')

Output:

```
--> exec('C:\Users\user\Desktop\clds\1st.sce', -1)

 "A bag contains 12 items of which four are defective.Three items are drawn at random,one after the other"

  0.2545455

 "probability that all three items are nondefective"
```

Independent events:

Code:

H =1;

T =2;

S =[111,112,121,122,211,212,221,222]

A =[111 ,112 ,121 ,122];

B =[111 ,112 ,211 ,212];

C =[112 ,211];

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')

AnB=intersect(A,B)

AnC=intersect (A,C)

BnC=intersect (B,C)

PAnB=length(AnB)/length(S);

disp(PAnB,'probability of the event AnB')

PAnC=length(AnC)/length(S);

disp(PAnC,'probability of the event AnC')

PBnC = length (BnC)/ length (S);

disp(PBnC,'probability of the event BnC')

*if((PA*PB)==PAnB),*

disp("A and B are independent")

else

disp("A and B a r e dependent ")

end

*if((PA*PC)==PAnC),*

disp("A and C are independent")

else

disp("A and C are dependent")

end

*if ((PB*PC)== PBnC),*

disp("B and C are independent ")

else

disp("B and C are dependent")

end

Output:

```
--> exec('C:\Users\user\Desktop\clds\1st.sce', -1)

  0.5

 "probability of A is"

  0.5

 "probability of B is"

  0.25

 "probability of C is"

  0.25

 "probability of the event AnB"

  0.125

 "probability of the event AnC"

  0.25

 "probability of the event BnC"

 "A and B are independent"

 "A and C are independent"

 "B and C are dependent"
```

Repeated trials with two outcomes:

Code:

disp("Experiment: Three horses race together twice ")

Ph1 =1/2;

Ph2 =1/3;

Ph3 =1/6;

S =[11 ,12 ,13 ,21 ,22 ,23 ,31 ,32 ,33]

*P11 = Ph1 *Ph1*

*P12 = Ph1 *Ph2*

*P13 = Ph1 *Ph3*

*P21 = Ph2 *Ph1*

*P22 = Ph2 *Ph2*

*P23 = Ph2 *Ph3*

*P31 = Ph3 *Ph1*

*P32 = Ph3 *Ph2*

*P33 = Ph3 *Ph3*

disp (P31,'probability of third horse winning the first race and first horse winning the second race is')

Output:

```
--> exec('C:\Users\user\Desktop\clds\1st.sce', -1)

 "Experiment: Three horses race together twice "

  0.0833333

 "probability of third horse winning the first race and first horse winning the second race is"
```

Conclusion: Successfully performed Probability Theory 2.

Experiment no – 05

Aim: Counting 1

a. Sum rule principle

b. Product rule principle

c. Factorial

d. Binomial coefficients

Sum rule principle:

Code:

```
M =8;
F =5;
T=M+F ;
disp(T,'number of ways students choose calculus')
E=[2 ,3 ,5 ,7];
F=[2 ,4 ,6 ,8];
G=intersect(E,F);
H=length(E)+length(F)-length(G);
disp(H,'event of getting even or prime number' )
E =[11 ,13 ,17 ,19];
F =[12 ,14 ,16 ,18];
G=union(E,F);
k=length(G);
disp(k,'no of ways of choosing even and prime no.')
```

Output:

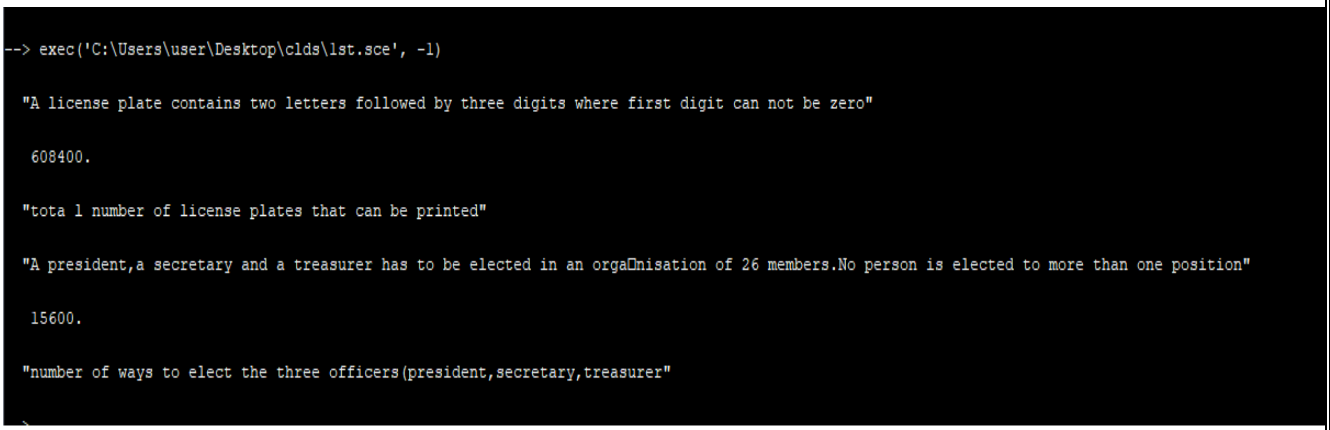
```
--> exec('C:\Users\user\Desktop\clds\1st.sce', -1)

  13.

 "number of ways students choose calculus"

  7.

 "event of getting even or prime number"

  8.

 "no of ways of choosing even and prime no."
```

Product rule principle:

Code:

```
disp('A license plate contains two letters followed by three digits where first digit can not be zero')

n =26;

n*n;

p =10;

(p -1) *p*p;

k=n*n*(p -1) *p*p;

disp(k,'tota l number of license plates that can be printed')

disp('A president,a secretary and a treasurer has to be elected in an orga nisation of 26 members.No person is elected to more than one position')

t =26;

j=t*(t -1) *(t -2) ;

disp(j,'number of ways to elect the three officers(president,secretary,treasurer')
```

Output:



```
--> exec('C:\Users\user\Desktop\clds\1st.sce', -1)

 "A license plate contains two letters followed by three digits where first digit can not be zero"

  608400.

 "tota l number of license plates that can be printed"

 "A president,a secretary and a treasurer has to be elected in an orga□nisation of 26 members.No person is elected to more than one position"

  15600.

 "number of ways to elect the three officers(president,secretary,treasurer"
```

Factorial:

Code:

a=factorial(6);

disp(a,'value of 6! is');

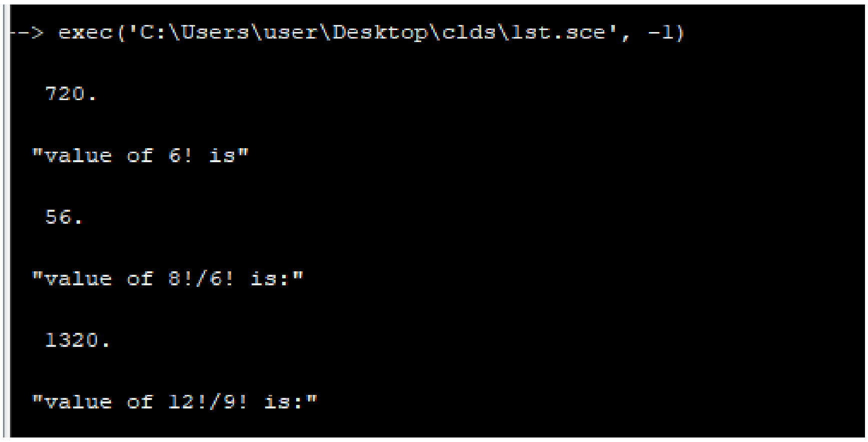
k=factorial(8)/factorial(6);

disp(k,'value of 8!/6! is:')

j=factorial(12)/factorial(9);

disp(j,'value of 12!/9! is:')

Output:



```
--> exec('C:\Users\user\Desktop\clds\1st.sce', -1)

  720.

 "value of 6! is"

  56.

 "value of 8!/6! is:"

  1320.

 "value of 12!/9! is:"
```

Binomial coefficients:

Code:

function[k]=func1(n,r)

*k=factorial(n)/(factorial(r)*factorial(n-r));*

disp(n,'n=')

disp(r,'r=')

disp(k,'k=')

endfunction

func1(8,2)

func1(9 ,4)

func1(12,5)

func1(10,3)

func1(13,1)

*p=factorial(10)/(factorial(10-7)*factorial(7));*

*q=factorial(10)/(factorial(10-3)*factorial(3));*

disp(p,'value of 10C7 is')

disp(q,'value of 10C3 is')

Output:

```
--> exec('C:\Users\user\Desktop\clds\1st.sce', -1)

  8.

 "n="

  2.

 "r="

  28.

 "k="

  9.

 "n="

  4.

 "r="

  126.

 "k="

  12.

 "n="

  5.

 "r="

  792.

 "k="

  10.

 "n="

  3.

 "r="

  120.
```

```
 "k="

  13.

 "n="

  1.

 "r="

  13.

 "k="

  120.

 "value of 10C7 is"

  120.

 "value of 10C3 is"
```

Conclusion: Successfully performed Counting 1.

Experiment no – 06

Aim: Counting 2

a. Permutations

b. Permutations with repetitions

c. Combinations

d. Combinations with repetitions

Permutations:

Code:

disp('finding the number of 3 letter words using only the given 6 letters(A,B,C,D,E,F) without repetition')

n =6;

l1=n;

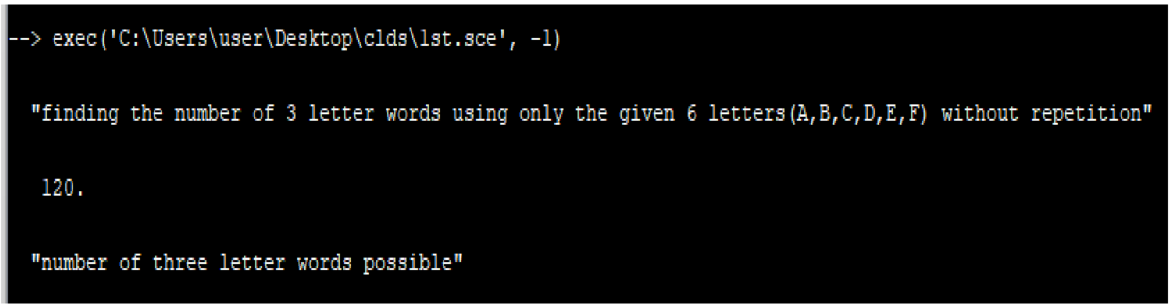
l2=n-1;

l3=n-2;

*k=l1*l2*l3;*

disp(k,'number of three letter words possible')

Output:



```
--> exec('C:\Users\user\Desktop\clds\1st.sce', -1)

  "finding the number of 3 letter words using only the given 6 letters(A,B,C,D,E,F) without repetition"

   120.

  "number of three letter words possible"
```

Permutations with repetitions:

Code:

function [k]=funct1(n, p, q)

k= factorial (n)/(factorial (p) factorial (q));*

endfunction

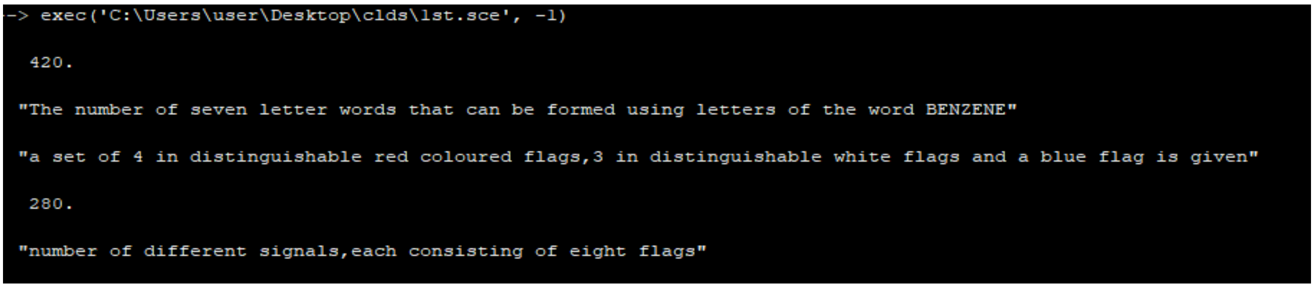
k=funct1(7,3,2)

disp(k,'The number of seven letter words that can be formed using letters of the word BENZENE') *disp('a set of 4 in distinguishable red coloured flags,3 in distinguishable white flags and a blue flag is given')*

j=funct1(8,4,3);

disp(j,'number of different signals,each consisting of eight flags')

Output:



```
-> exec('C:\Users\user\Desktop\clds\1st.sce', -1)

  420.

 "The number of seven letter words that can be formed using letters of the word BENZENE"

 "a set of 4 in distinguishable red coloured flags,3 in distinguishable white flags and a blue flag is given"

  280.

 "number of different signals,each consisting of eight flags"
```

Combinations:

Code:

function [k]=myfunc(n, r)

k= factorial (n)/(factorial (n-r) factorial (r));*

endfunction

a= myfunc (8 ,3);

disp (a, ' no. of committees of 3 that can be formed out of 8 people is ')

cows = myfunc (6 ,3)

disp(cows, 'No of cows=')

bulls = myfunc (5 ,2)

disp(bulls, 'No of bulls=')

hens = myfunc (8 ,4)

disp(hens, 'No of hens=')

*p= cows * bulls * hens ;*

disp(p,'total number of ways that a farmer can choose all these animals')

Output:

```
--> exec('C:\Users\user\Desktop\clds\1st.sce', -1)

  56.

 " no. of committees of 3 that can be formed out of 8 people is "

  20.

 "No of cows="

  10.

 "No of bulls="

  70.

 "No of hens="

  14000.

 "total number of ways that a farmer can choose all these animals"
```

Combinations with repetitions:

Code:

r =5;

M =2;

*m= factorial (r+(M -1))/(factorial (r+(M -1) -(M -1))*factorial (M -1));*

disp (m, ' no. of non negative integer solutions of the given equation x+y+z=18 ')

Output:

```
--> exec('C:\Users\user\Desktop\clds\1st.sce', -1)

  6.

 " no. of non negative integer solutions of the given equation x+y+z=18 "
```

Conclusion: Successfully performed Counting 2.

Experiment no – 07

Aim: Counting 3

a. Ordered partitions

b. Unordered partitions

Ordered partitions:

Code:

c1 =3;

c2 =2;

c3 =2;

c4 =2;

m= factorial (9) /(factorial (3) factorial (2)* factorial(2) * factorial (2));*

disp(m,'number of ways nine toys can be divide dbetween four children with the youngest son getting 3 toys and others getting 2 each')

Output:

```
--> exec('C:\Users\user\Desktop\clds\1st.sce', -1)

  7560.

 "number of ways nine toys can be divide dbetween four children with the youngest son getting 3 toys and others getting 2 each"
```

Unordered partitions:

Code:

p =12;

t =3;

disp('each partition of the students can be arranged in 3 ! ways as an ordered partition')

r= factorial (12) /(factorial (4) factorial (4) * factorial(4))*

m=r/ factorial (t);

disp (m, ' number o f ways that 12 students can be partitioned into three teams so that each team consists of 4 students')

Output:

```
--> exec('C:\Users\user\Desktop\clds\1st.sce', -1)

 "each partition of the students can be arranged in 3 ! ways as an ordered partition"

  5775.

 " number o f ways that 12 students can be partitioned into three teams so that each team consists of 4 students"
```

Conclusion: Successfully performed Counting 3.

Experiment no – 08

Aim: Graph Theory

a. Paths and connectivity

b. Minimum spanning tree

c. Isomorphism

Paths and connectivity:

Code:

disp('given a graph with 6 node sviz . node1 , node2. . . . node6 ')

A =[0 1 0 1 1 0;1 0 1 0 1 0;0 1 0 0 0 1;1 0 0 0 0 0;1 1 0 0 0 0;0 0 1 0 0 0]; disp(A,'The adjacency matrix for A is') disp('sequence A i s a path from node4 to node6 ; but it is not a trail since the edge from node1 to node2 is used twice')

B =[0 0 0 1 1 0;0 0 0 0 1 1;0 0 0 0 0 0;1 0 0 0 0 0;1 1 0 0 0 0;0 1 0 0 0 0]; disp(B,'The adjacency matrix for B is') disp('sequence B is not a path since there is no edge from node2 to node6 is used twice')

C =[0 0 0 1 1 0;0 0 1 0 1 0;0 1 0 0 1 0;1 0 0 0 0 0;1 1 1 0 0 1;0 0 0 0 1 0]; disp(C,'The adjacency matrix for C is') disp('sequence C is a trail since is no edge is used twice')

D =[0 0 0 1 1 0;0 0 0 0 0 0;0 0 0 0 1 1;1 0 0 0 0 0;1 0 1 0 0 0;0 0 1 0 0 0]; disp(D,'The a djacency matrix for D is') disp('sequence D is a simple path from node4 to node6')

Output:

```
--> exec('C:\Users\user\Desktop\clds\1st.sce', -1)

 "given a graph with 6 node sviz . node1 , node2. . . . node6 "

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

 "The adjacency matrix for A is"

 "sequence A i s a path from node4 to node6 ; but it is not a trail since the edge from node1 to node2 is used twice"

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

 "The adjacency matrix for B is"

 "sequence B is not a path since there is no edge from node2 to node6 is used twice"

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

 "The adjacency matrix for C is"

 "sequence C is a trail since is no edge is used twice"

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

 "The a djacency matrix for D is"

 "sequence D is a simple path from node4 to node6"
```

Minimum spanning tree:

Code:

```
disp('to find: minimal spanning tree' )
disp('the adjacency matrix for the weighted graph (nodeA , nodeB . . . nodeF )of 6 nodes is :')
K =[0 0 7 0 4 7;0 0 8 3 7 5;7 8 0 0 6 0;0 3 0 0 0 4;4 7 6 0 0 0;7 5 0 4 0 0]
disp ('edges of the graph')
AC =7;
AE =4;
AF =7;
BC =8;
BD =3;
BE =7;
BF =5;
CE =6;
DF =4;
M=[AC ,AE ,AF ,BC ,BD ,BE ,BF ,CE ,DF ];
V= int32 (M);
L= gsort (V)
disp('deleting edges without disconnecting the graph until 5 edges remain' )
N=[BE ,CE ,AE ,DF ,BD ];
Sum=sum (N);
disp(Sum,'weight of the minimal spanning tree is ')
disp ( ' another method of finding a minimal spanning tree is : ')
K= gsort (V, 'g','i' )
N2 =[BD ,AE ,DF ,CE ,AF ];
Sum2=sum(N2);
disp(Sum2 ,' weight of the minimal spanning tree is' )
```

Output:

```
-> exec('C:\Users\user\Desktop\clds\1st.sce', -1)

 "to find: minimal spanning tree"

 "the adjacency matrix for the weighted graph (nodeA , nodeB . . . nodeF )of 6 nodes is :"

 "edges of the graph"

 "deleting edges without disconnecting the graph until 5 edges remain"

  24.

 "weight of the minimal spanning tree is "

 " another method of finding a minimal spanning tree is : "

  24.

 " weight of the minimal spanning tree is"
```

Isomorphism:

Code:

```
A_V=5;
R_V=5;
A_E=5;
R_E=5;
A=[0 1 1 0 0;1 0 1 1 0; 1 1 0 0 1;0 1 0 0 0; 0 0 1 0 0];
R=[0 1 1 0 0; 1 0 1 1 0; 1 1 0 0 1; 0 1 0 0 0; 0 0 0 0 1]
disp(A, 'adjacency matrix for graph A')
disp(R,'adjacnecy matrix for graph R');
k=0;
if (A==R) then
 for i=1:25
 if (A(i)==R(i)) then
 k=k+1;
 else
 break;
 end
 end
end
```

if (k==25) then

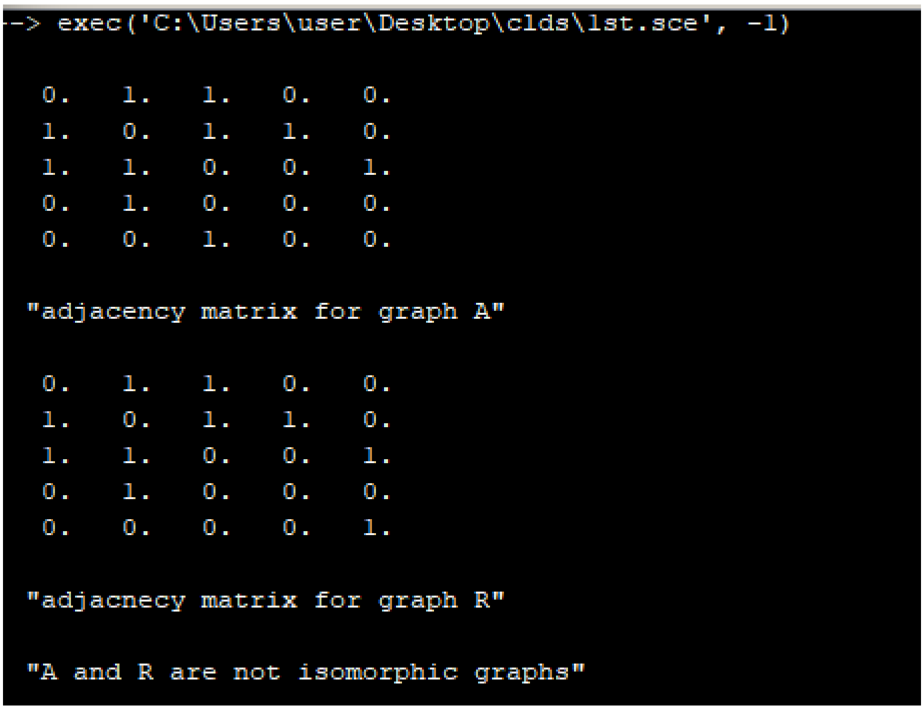
disp('A and R are isomorphic graphs')

else

disp('A and R are not isomorphic graphs')

end

Output:



```
-> exec('C:\Users\user\Desktop\clds\1st.sce', -1)

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

 "adjacency matrix for graph A"

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

 "adjacnecy matrix for graph R"

 "A and R are not isomorphic graphs"
```

Conclusion: Successfully performed Graph Theory.

Experiment no – 09

Aim: Directed Graphs

a. Adjacency matrix

b. Path matrix

Adjacency matrix:

Code:

A=[0 0 0 1;1 0 1 1;1 0 0 1;1 0 1 0];

disp (A,'adjacency matrix of graph G is')

A2=A^2

A3=A^3

disp(A2,'the number of ones in A is equal to the no. of edges in the graph ie.8')

Output:

```
-> exec('C:\Users\user\Desktop\clds\1st.sce', -1)

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

 "adjacency matrix of graph G is"

  1.   0.   1.   0.
  2.   0.   1.   2.
  1.   0.   1.   1.
  1.   0.   0.   2.

 "the number of ones in A is equal to the no. of edges in the graph ie.8"
```

Path matrix:

Code:

A =[0 0 0 1;1 0 1 1;1 0 0 1;1 0 1 0];

disp (A, ' a d j a c e n c y mat r ix o f graph G i s ')

A4=A^4;

A3=A^3;

A2=A^2;

B4=A+A2+A3+A4;

B4 =[4 11 7 7 0 0 0 0 3 7 4 4 4 11 7 7];

for i =1:16

if(B4(i) ~=0) then

B4(i) =1;

end

end

disp (B4,'Replacing non zero entries of B4 with 1 ,we get path (reach ability)matrix P is:')

disp('there are zero entries in P,therefore the graph is not strongly connected')

Output:

```
-> exec('C:\Users\user\Desktop\clds\1st.sce', -1)

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

 " a d j a c e n c y mat r ix o f graph G i s "

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

 "Replacing non zero entries of B4 with 1 ,we get path (reach ability)matrix P is:"

 "there are zero entries in P,therefore the graph is not strongly connected"
```

Conclusion: Successfully performed Directed Graphs.

Experiment no – 10

Aim: Recurrence relations

a. Linear homogeneous recurrence relations with constant coefficients

b. Solving linear homogeneous recurrence relations with constant coefficients

c. Solving general homogeneous linear recurrence relations

Linear homogeneous recurrence relations with constant coefficients:

Code:

```
a =[];
a (1) =1;
a (2) =2;
disp ('for recurrence relationa(n)=5*a(n-1)-4*a(n-2)+n^2')
for n =3:4
a(n)=5*a(n-1)-4*a(n-2)+n^2;
mprintf('  Value of a(%i ) is:%inn ',n,a(n))
end
a=[];
a(1)=1;
a(2)=2;
disp('for recurrence relationa(n)=2*a(n-1)*a(n-2)+n^2')
for n =3:4
a(n)=2*a(n -1)*a(n -2)+n^2;
mprintf('  Value of a(%i) is:%inn',n,a(n))
end
a =[];
a (1) =1;
a (2) =2;
disp('for recurrence relation a(n)=n*a(n-1)+3*a(n-2)')
for n =3:4
a(n)=n*a(n -1) +3*a(n -2) ;
mprintf ( '  Value o f a (%i ) i s : %i nn ' ,n,a(n))
```

end

a =[];

a (1) =1; // i n i t i a l c o n d i t i o n

a (2) =2; // i n i t i a l c o n d i t i o n

a (3) =1; // i n i t i a l c o n d i t i o n

*disp('for recurrence relation a(n)=2*a(n-1)+5*a(n-2)-6*a(n-3)')*

for n =4:6

*a(n)=2*a(n -1) +5*a(n -2) -6*a(n -3) ;*

mprintf (' Value o f a (%i) i s : %i nn ' ,n,a(n))

end

Output:

```
--> exec('C:\Users\user\Desktop\clds\1st.sce', -1)

 "for recurrence relationa(n)=5*a(n-1)-4*a(n-2)+n^2"
 Value of a(3 ) is:15nn   Value of a(4 ) is:83nn
 "for recurrence relationa(n)=2*a(n-1)*a(n-2)+n^2"
 Value of a(3) is:13nn  Value of a(4) is:68nn
 "for recurrence relation a(n)=n*a(n-1)+3*a(n-2)"
 Value o f a (3 ) i s : 9 nn   Value o f a (4 ) i s : 42 nn
 "for recurrence relation a(n)=2*a(n-1)+5*a(n-2)-6*a(n-3)"
 Value o f a (4 ) i s : 6 nn   Value o f a (5 ) i s : 5 nn   Value o f a (6 ) i s : 34 nn
```

Solving linear homogeneous recurrence relations with constant coefficients:

Code:

disp (' recurrence relation of Fibonacci numbers f [n]=f [n-1]+ f [n-2] ')

x= poly (0,'x');

g=x^2-x -1;

disp (g, ' characteristic equation of the recurrence relation is: ')

j =[];

j= roots (g);

disp (j,' roots of the characteristic equation j1 , j 2 ')

disp (' for general equation fn=Ar^n+Br^n , values of A and B respectively are calculated as:')

disp (' initial condition at n=0 and n=1 respectively are: ')

f1 =1;

D=[1.6180340 -0.618034;(1.6180340) ^2 (-0.618034)^2];

K =[1 1];

c =[];

c=D/K;

A=c(1)

B=c(2)

disp(A,'A=')

disp(B,'B=')

disp (' thus the soluton is f[n]=0.4472136((1.61834)^n-(-0.4472136)^n)')*

Output:

```
--> exec('C:\Users\user\Desktop\clds\1st.sce', -1)

  " recurrence relation of Fibonacci numbers f [ n]=f [ n-1]+ f [ n-2] "

  -1 -x +x²

  " characteristic equation of the recurrence relation is: "

  -0.6180340
   1.6180340

  " roots of the characteristic equation j1 , j 2 "

  " for general equation fn=Ar^n+Br^n , values of A and B respectively are calculated as:"

  " initial condition at n=0 and n=1 respectively are: "

   0.5000000

  "A="

   1.5000000

  "B="

  " thus the soluton is f[n]=0.4472136*((1.61834)^n-(-0.4472136)^n)"
```

Solving general homogeneous linear recurrence relations:

Code:

disp ('The recurrence relation a [n]=11 a [n-1]-39*a [n-2]+45* a [n-3] ')*

x= poly (0,'x');

disp (g=x ^3 -11 x ^2+39*x -45 , ' character stic polynomial equation for the above recurrence relation')*

j =[];

j= roots (g);

disp (j, ' roots of the character stice quation j1 , j 2 ')

*disp (' hence the general solution is : a [n]=c1 *3^ n)+c2 n *3^ n)+c3 *5^ n) ')*

disp (' initial condition at n=0 and n=1 respectively are : ')

a0 =5;

a1 =11;

a2 =25;

D =[1 0 1;3 3 5;9 18 25];

K =[5 11 25]

c =[];

c=D/K;

c1=c(1)

c2=c(2)

c3=c(3)

disp (' thus the s o l u t i o n i s a [n]=(4-2n)(3^ n)+5^n ')*

Output:

```
--> exec('C:\Users\user\Desktop\clds\1st.sce', -1)

 "The recurrence relation a [ n]=11* a [ n-1]-39*a [ n-2]+45* a [ n-3] "

 " character stic polynomial equation for the above recurrence relation"

 -0.6180340
  1.6180340

 " roots of the character stice quation j1 , j 2 "

 " hence the general solution is : a [ n]=c1 *3^ n )+c2 n *3^ n )+c3 *5^ n ) "

 " initial condition at n=0 and n=1 respectively are : "

 " thus the s o l u t i o n i s a [ n]=(4-2n )*(3^ n )+5^n "
```

Conclusion: Successfully performed Recurrence relations.