Information Theory And Coding

Name: Hari Krishna P

Reg No: 19BCE7675

DATE: 17/11/2021

Contents:

- **TOTAL: 11 LAB Experiments**

IDE Platform : MatLab

PRESENTED TO:

Prof . Agam Das Gowswami

ITC LAB – 1

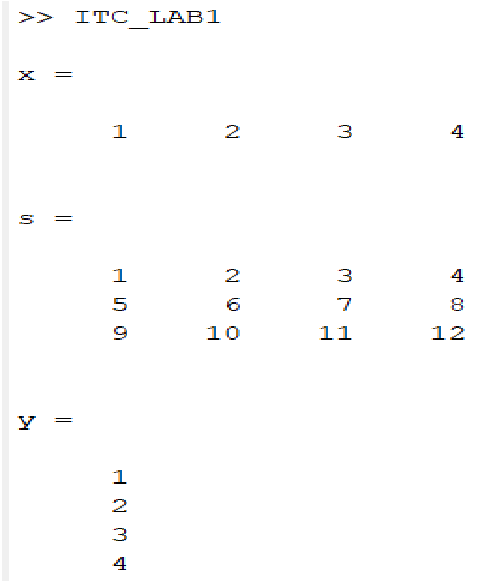
AIM: Basics of MatLab

- Numbers
- Variables
- Assigning
- Vectors

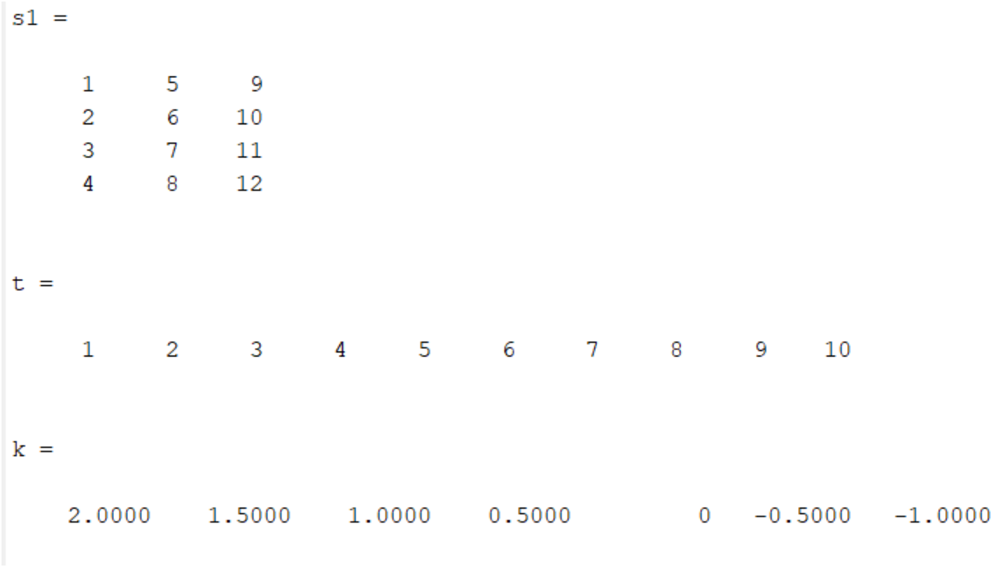
CODE:

```
x=[1 2 3 4]
s=[1 2 3 4;5 6 7 8;9 10 11 12]

y=x'

s1=s'

t = 1:10

k = 2:-0.5:-1

b = [2:1:50]

a = [1 2 3 4;5 6 7 8;9 1 2 3]

a(1,2)

a(4:1,2)

a(2,:)


a(:,3)

a(3,:)

a = [1 2 3 4;5 6 7 8;9 1 2 3]

a=1:3;
b=4:6;
C=a+b

A = [a;b]
```

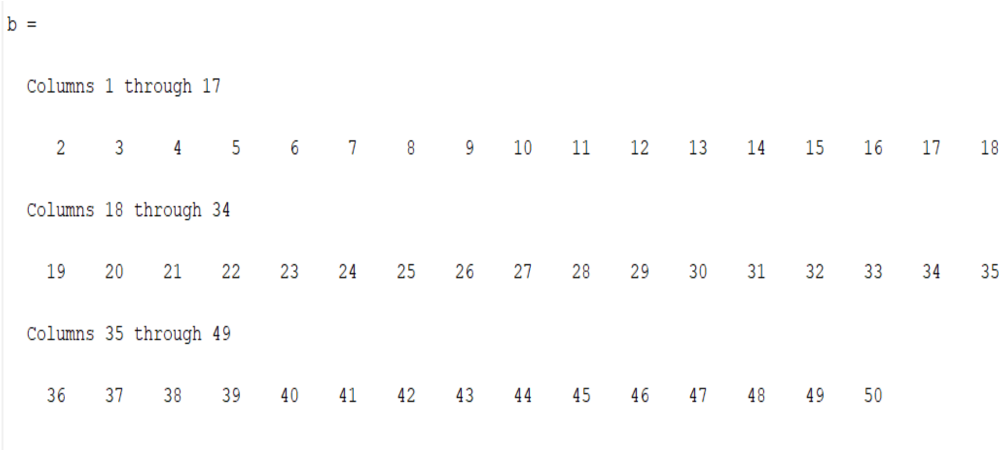
```
B=A'

c=A*B

D=A.*A
B=[1 2 3;4 5 6]

C=A./B

C=A.^B

C=A.*B

C = A+B

C = A-B

C=A'.*B'

C=A'./B'

C=[A B]

a

A

B

C=[B;A]
```



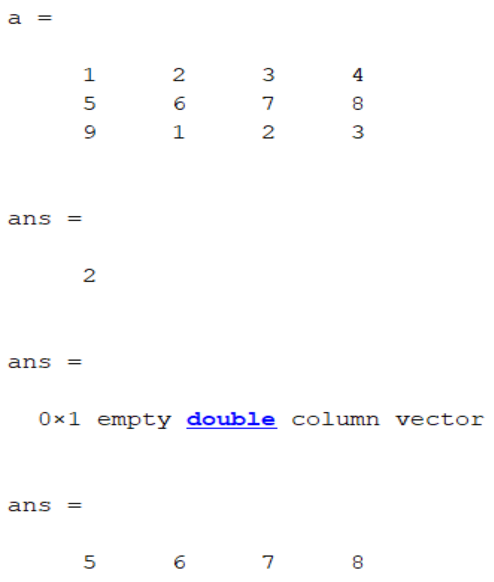
```
>> ITC_LAB1

x =

     1     2     3     4


s =

     1     2     3     4
     5     6     7     8
     9    10    11    12


y =

     1
     2
     3
     4
```



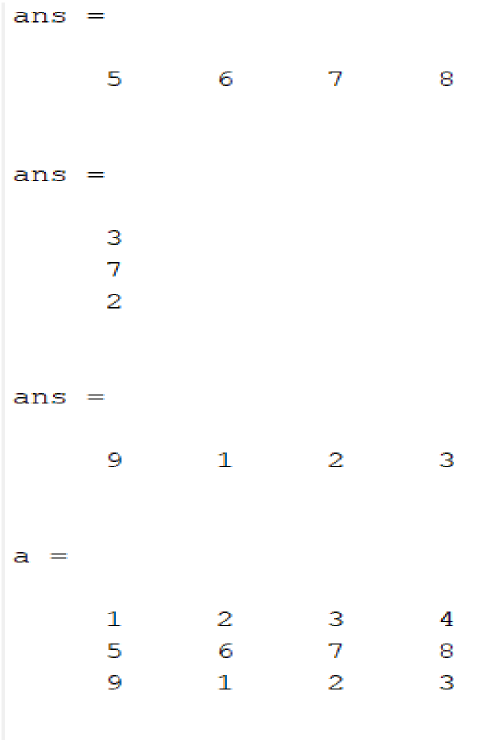
```
s1 =

     1     5     9
     2     6    10
     3     7    11
     4     8    12


t =

     1     2     3     4     5     6     7     8     9    10


k =

    2.0000    1.5000    1.0000    0.5000         0   -0.5000   -1.0000
```



```
b =

  Columns 1 through 17

     2     3     4     5     6     7     8     9    10    11    12    13    14    15    16    17    18

  Columns 18 through 34

    19    20    21    22    23    24    25    26    27    28    29    30    31    32    33    34    35

  Columns 35 through 49

    36    37    38    39    40    41    42    43    44    45    46    47    48    49    50
```

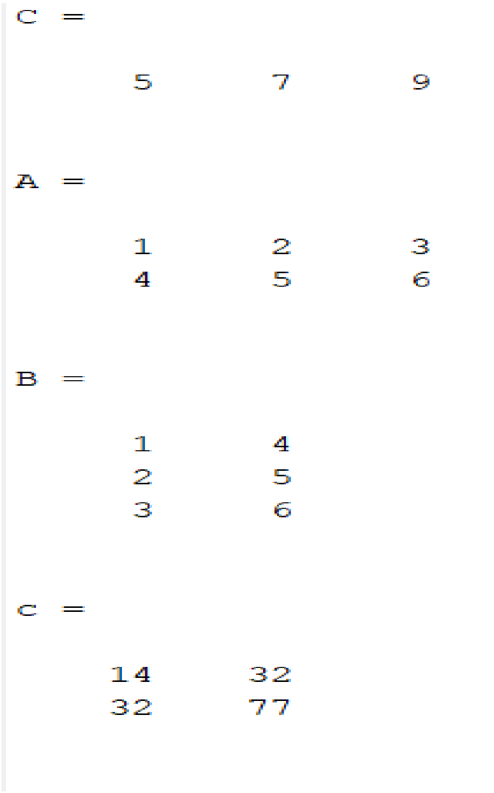


```
a =

     1     2     3     4
     5     6     7     8
     9     1     2     3


ans =

     2


ans =

  0×1 empty double column vector


ans =

     5     6     7     8
```

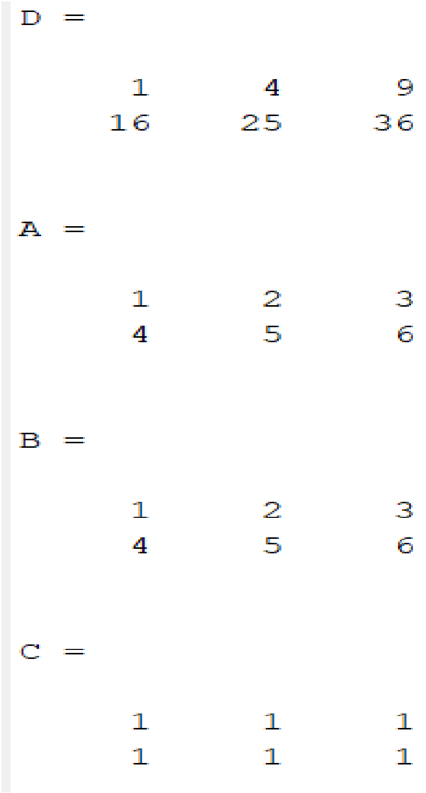


```
ans =

     5     6     7     8


ans =

     3
     7
     2


ans =

     9     1     2     3


a =

     1     2     3     4
     5     6     7     8
     9     1     2     3
```

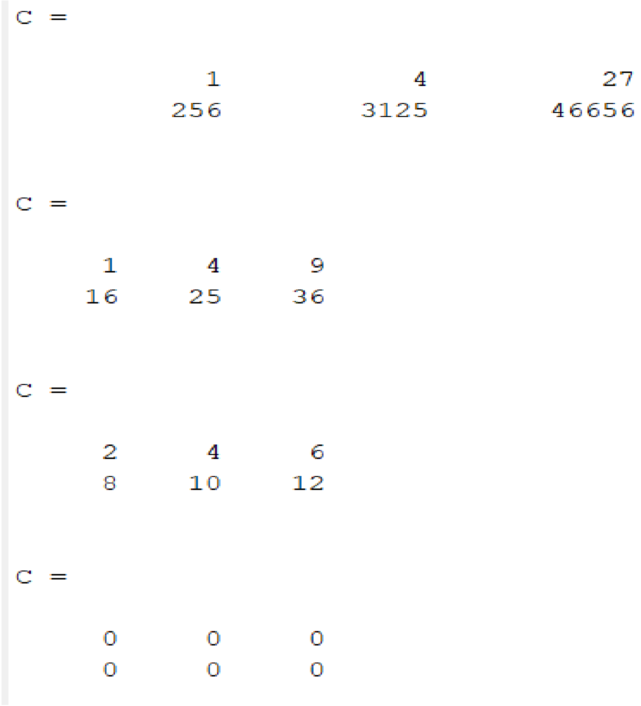
A



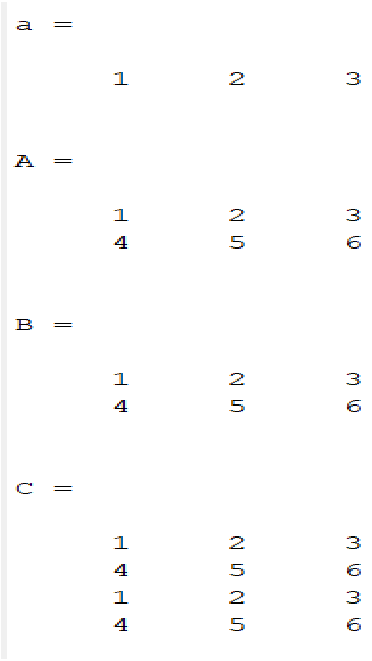
```
C =

     5     7     9


A =

     1     2     3
     4     5     6


B =

     1     4
     2     5
     3     6


c =

    14    32
    32    77
```



```
D =

     1     4     9
    16    25    36


A =

     1     2     3
     4     5     6


B =

     1     2     3
     4     5     6


C =

     1     1     1
     1     1     1
```



```
C =

           1           4          27
         256        3125       46656


C =

     1     4     9
    16    25    36


C =

     2     4     6
     8    10    12


C =

     0     0     0
     0     0     0
```



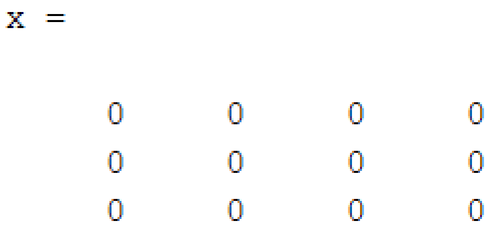
```
a =

     1     2     3


A =

     1     2     3
     4     5     6


B =

     1     2     3
     4     5     6


C =

     1     2     3
     4     5     6
     1     2     3
     4     5     6
```

ITC LAB – 2

AIM: Basics of MatLab

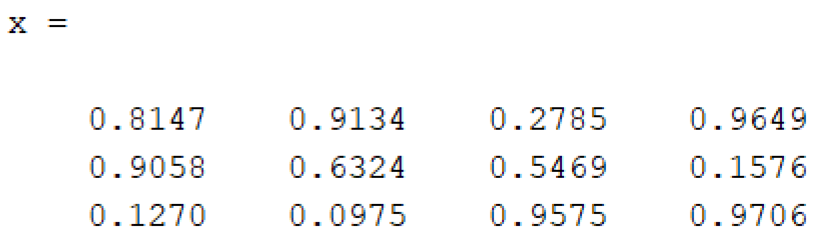
- **Matrix**
- **Vectors**
- **Arithmetic Problems**

x=zeros(3,4)



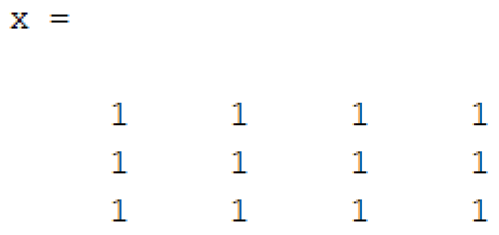
```
x =

     0     0     0     0
     0     0     0     0
     0     0     0     0
```

x=rand(3,4)



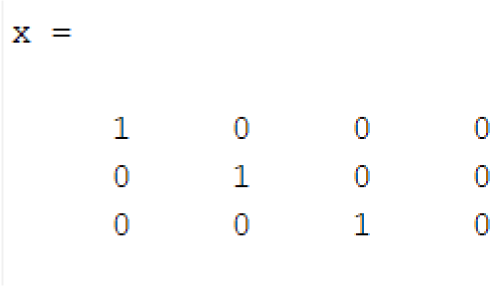
```
x =

    0.8147    0.9134    0.2785    0.9649
    0.9058    0.6324    0.5469    0.1576
    0.1270    0.0975    0.9575    0.9706
```

x=ones(3,4)



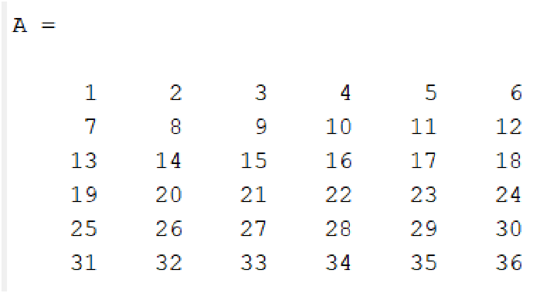
```
x =

     1     1     1     1
     1     1     1     1
     1     1     1     1
```

x=eye(3,4)



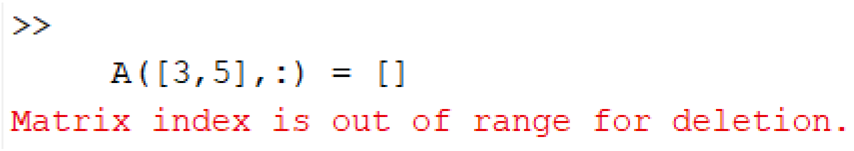
```
x =

     1     0     0     0
     0     1     0     0
     0     0     1     0
```

A=[1 2 3 4 5 6;7 8 9 10 11 12;13 14 15 16 17 18;19 20 21 22 23 24;25 26 27 28 29 30;31 32 33 34 35 36]



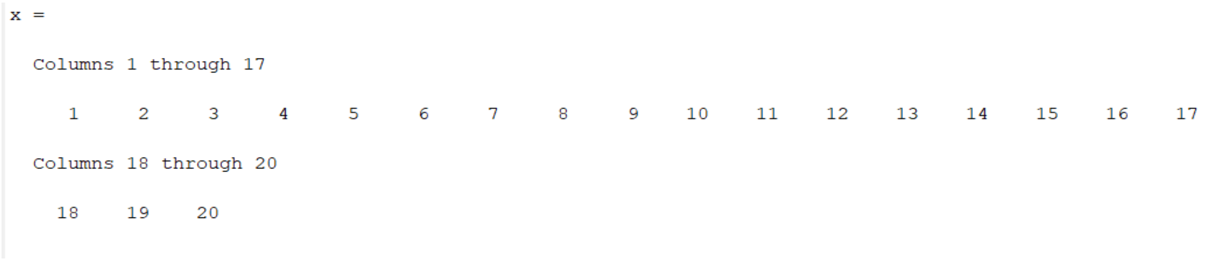
```
A =

     1     2     3     4     5     6
     7     8     9    10    11    12
    13    14    15    16    17    18
    19    20    21    22    23    24
    25    26    27    28    29    30
    31    32    33    34    35    36
```

A([3,5],:) = []



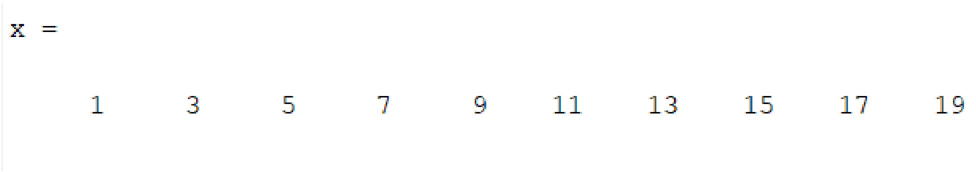
```
>>
     A([3,5],:) = []
Matrix index is out of range for deletion.
```

x = (1:20)



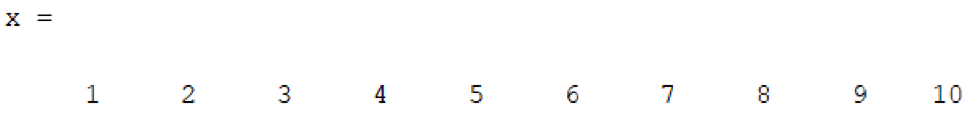
```
x =

  Columns 1 through 17

     1     2     3     4     5     6     7     8     9    10    11    12    13    14    15    16    17

  Columns 18 through 20

    18    19    20
```

x = (1:2:20)



```
x =

     1     3     5     7     9    11    13    15    17    19
```

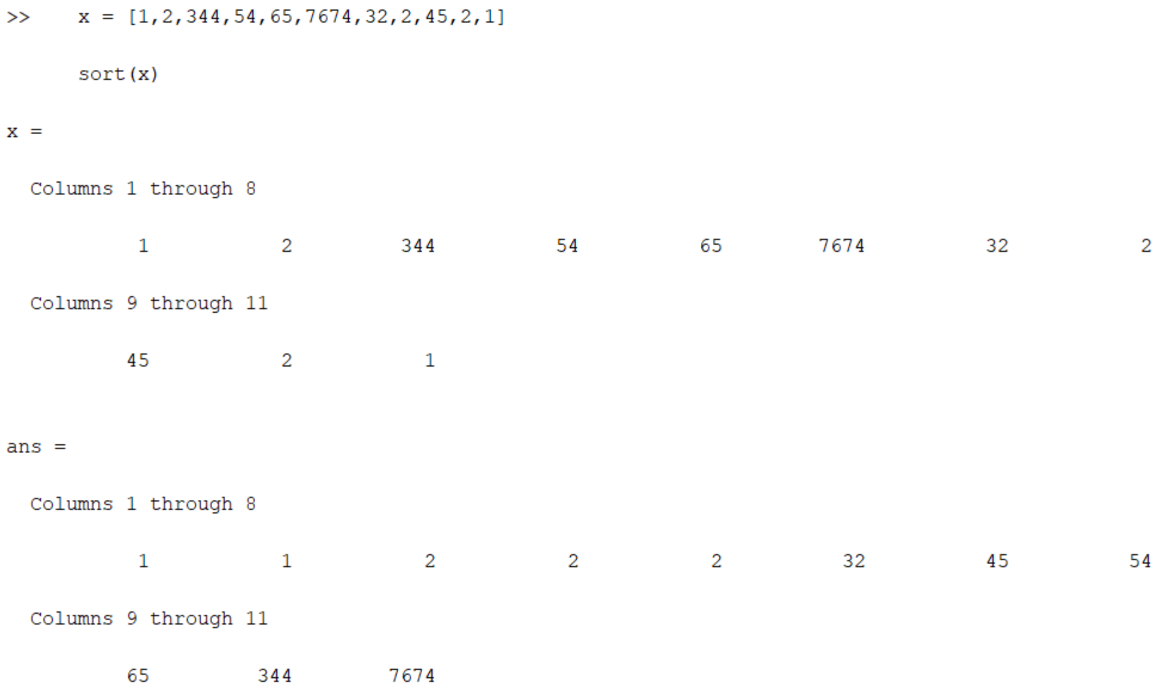
x = (1:10)



```
x =

     1     2     3     4     5     6     7     8     9    10
```

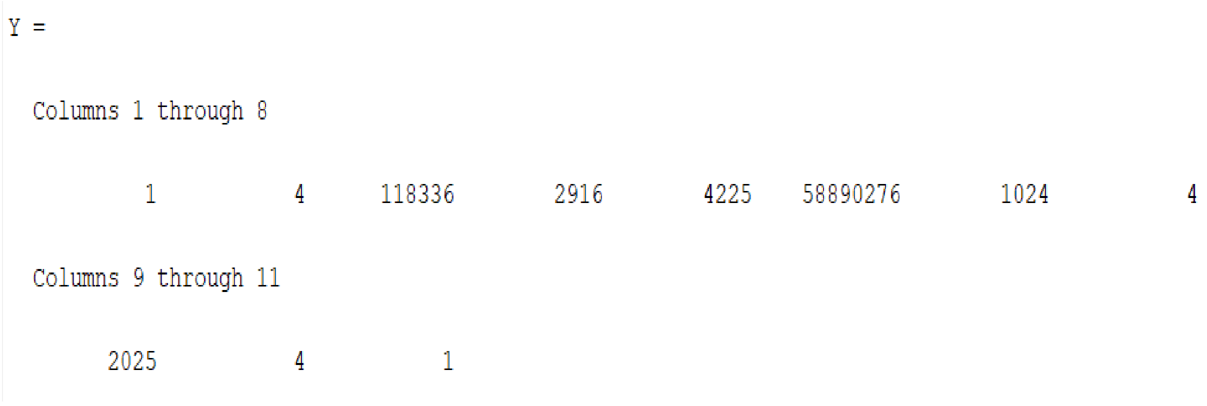
x = [1,2,344,54,65,7674,32,2,45,2,1]

sort(x)



```
>>    x = [1,2,344,54,65,7674,32,2,45,2,1]

      sort(x)

x =

  Columns 1 through 8

           1           2         344          54          65        7674          32           2

  Columns 9 through 11

          45           2           1


ans =

  Columns 1 through 8

           1           1           2           2           2          32          45          54

  Columns 9 through 11

          65         344        7674
```

Y=x.*x



```
Y =

  Columns 1 through 8

           1           4      118336        2916        4225    58890276        1024           4

  Columns 9 through 11

        2025           4           1
```

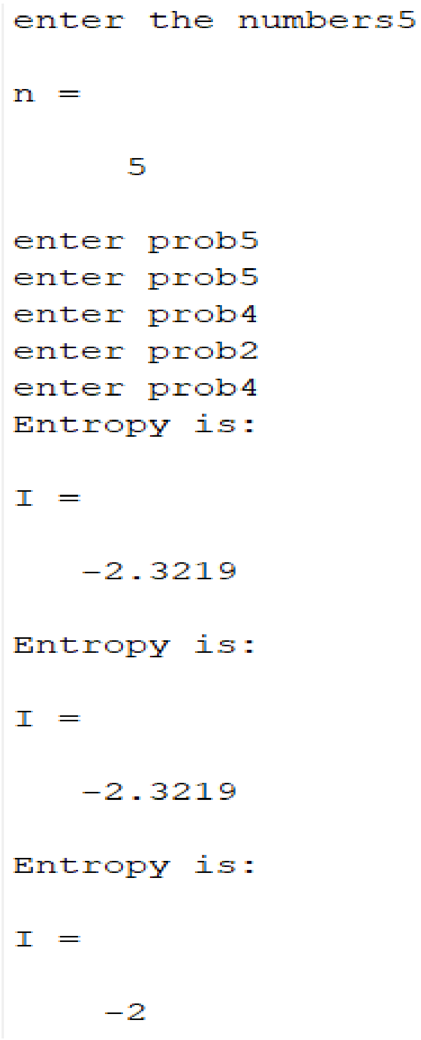
ITC LAB – 3

AIM: Finding The Entropy From A Vector By User Defined Method

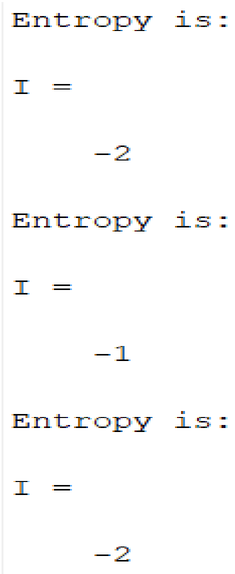
CODE:

```
clc
n=input('enter the numbers')
for i=1:n
    a(i)=input('enter prob');
end
s=0;
for i=i:n
    s=s+log(1/a(i))/log(2);
end

for i=1:n
    I=log(1/a(i))/log(2);
    disp('Entropy is: ')
    display(I)
end
```

OUTPUT:



```
enter the numbers5

n =

     5

enter prob5
enter prob5
enter prob4
enter prob2
enter prob4
Entropy is:

I =

   -2.3219

Entropy is:

I =

   -2.3219

Entropy is:

I =

    -2
```



```
Entropy is:

I =

    -2

Entropy is:

I =

    -1

Entropy is:

I =

    -2
```

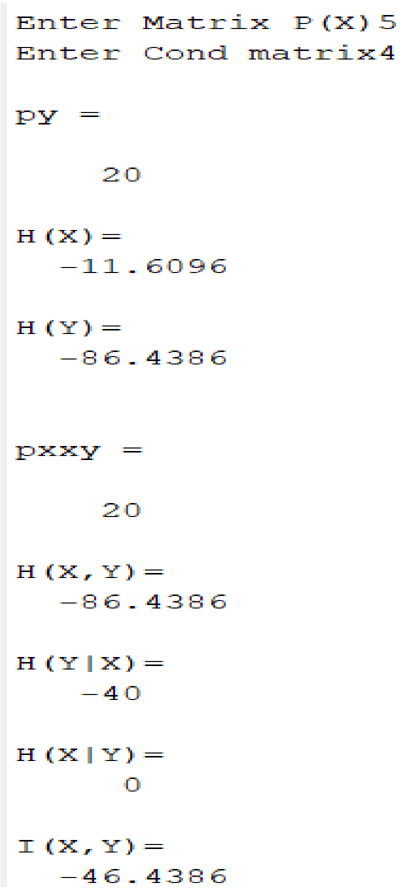
ITC LAB – 4

AIM: "Finding The Entropy And Conditional Entropy For A Matrix"

CODE:

```
clc
clear all
px=input('Enter Matrix P(X)');
pyx=input('Enter Cond matrix');
py=px*pyx
sum=0;
sum1=0;
r=300;
for i=1:width(px)
sum=sum+px(i)*log2(1/px(i));
end
%finding entropy of h and y
disp('H(X)=')
disp(sum)
for i=1:width(py)
sum1=sum1+py(i)*log2(1/py(i));
end
disp('H(Y)=')
disp(sum1)
for i=1:height(pyx)
for j=1:width(pyx)
    pxxy(i,j)=pyx(i,j)*px(i);
end

end
pxxy%joint probability
sum2=0;
for i=1:height(pxxy)
for j=1:width(pxxy)
    sum2=sum2+pxxy(i,j)*log2(1/pxxy(i,j));
end
end
disp('H(X,Y)=')
disp(sum2)
sum3=0;
for i=1:height(pyx)
for j=1:width(pyx)
    sum3=sum3+pxxy(i,j)*log2(1/pyx(i,j));
end
end
disp('H(Y|X)=')
disp(sum3)
disp('H(X|Y)=')
disp(sum2-sum1)
disp('I(X,Y)=')
disp(sum1-sum3)
```

OUTPUT:



```
Enter Matrix P(X)5
Enter Cond matrix4

py =

    20

H(X)=
  -11.6096

H(Y)=
  -86.4386


pxxy =

    20

H(X,Y)=
  -86.4386

H(Y|X)=
   -40

H(X|Y)=
     0

I(X,Y)=
  -46.4386
```

ITC LAB – 5

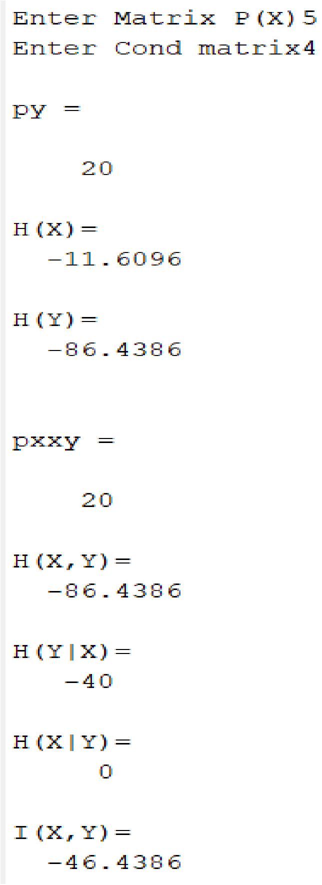
AIM: **“Finding The Entropy,Conditional Entropy & Joint Probablity For A Matrix”**

CODE:

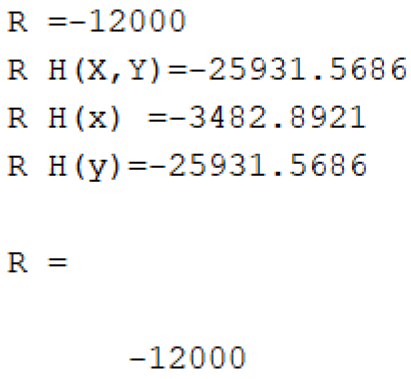
```
clc

clear all
px=input('Enter Matrix P(X)');
pyx=input('Enter Cond matrix');
py=px*pyx
sum=0;
sum1=0;
r=300;
for i=1:width(px)
sum=sum+px(i)*log2(1/px(i));
end
%finding entropy of h and y
disp('H(X)=')
disp(sum)
for i=1:width(py)
sum1=sum1+py(i)*log2(1/py(i));
end
disp('H(Y)=')
disp(sum1)
for i=1:height(pyx)
for j=1:width(pyx)
    pxxy(i,j)=pyx(i,j)*px(i);
end

end
pxxy%joint probability
sum2=0;
for i=1:height(pxxy)
for j=1:width(pxxy)
    sum2=sum2+pxxy(i,j)*log2(1/pxxy(i,j));
end
end
disp('H(X,Y)=')
disp(sum2)
sum3=0;
```

```
for i=1:height(pyx)
for j=1:width(pyx)
    sum3=sum3+pxxy(i,j)*log2(1/pyx(i,j));
end
end
disp('H(Y|X)=')
disp(sum3)
disp('H(X|Y)=')
disp(sum2-sum1)
disp('I(X,Y)=')
disp(sum1-sum3)

disp("R =" +r*sum3);
disp("R H(X,Y)=" +r*sum2);
disp("R H(x) =" +r*sum);
disp("R H(y)=" +r*sum1);
R=r*sum3
```

OUTPUT:



```
Enter Matrix P(X)5
Enter Cond matrix4

py =

    20

H(X)=
  -11.6096

H(Y)=
  -86.4386


pxxy =

    20

H(X,Y)=
  -86.4386

H(Y|X)=
   -40

H(X|Y)=
     0

I(X,Y)=
  -46.4386
```



```
R =-12000
R H(X,Y)=-25931.5686
R H(x) =-3482.8921
R H(y)=-25931.5686

R =

     -12000
```

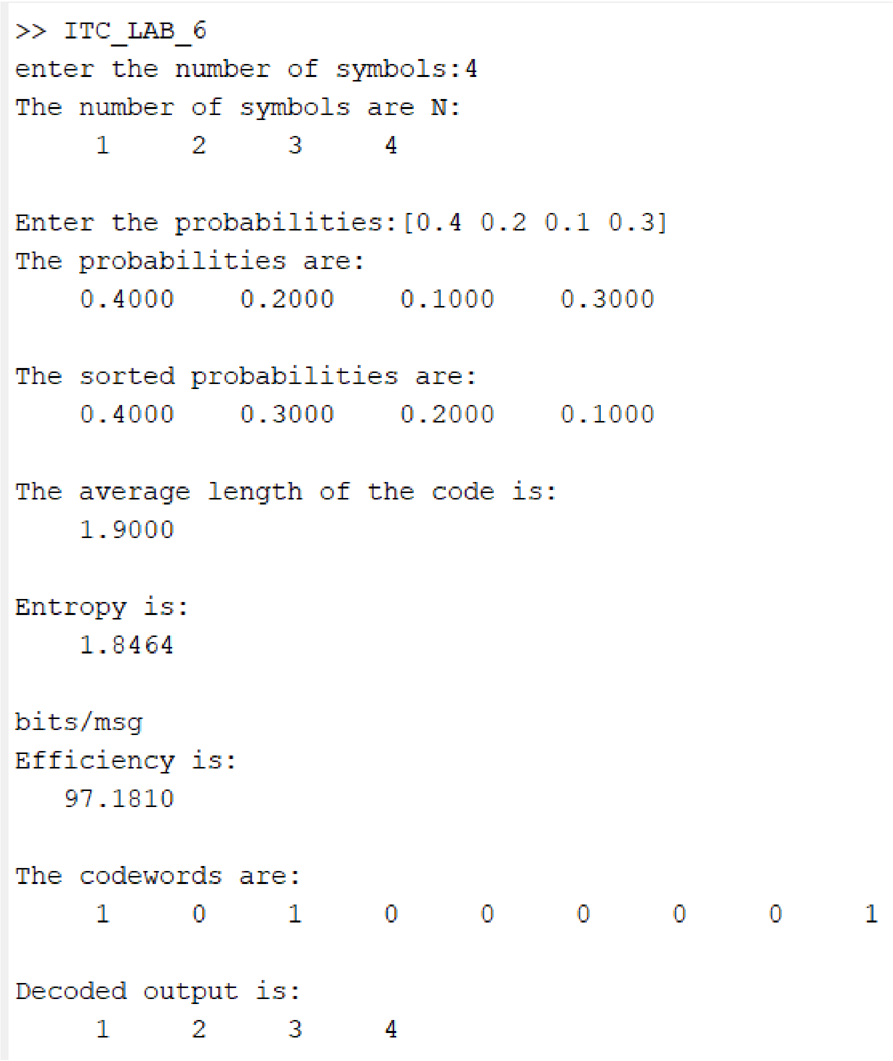
ITC LAB – 6

AIM: "Give The Encoded & Decoded Messages Using Huffman Coding"

CODE:

```
clear all
x=input('enter the number of symbols:');
N=1:x;
disp('The number of symbols are N:');
disp(N);
P=input('Enter the probabilities:');
disp('The probabilities are:');
disp(P);
S=sort(P,'descend');
disp('The sorted probabilities are:');
disp(S);
[dict,avglen]=huffmandict(N,S);
disp('The average length of the code is:');
disp(avglen);
H=0;
for i=1:x
    H=H+(P(i)*log2(1/P(i)));
end
disp('Entropy is:');
disp(H);
disp('bits/msg');
E=(H/avglen)*100;
disp('Efficiency is:')
disp(E);
codeword=huffmanenco(N,dict);
```

```
disp('The codewords are:');
disp(codeword);
decode=huffmandeco(codeword,dict);
disp('Decoded output is:');
disp(decode);
```



```
>> ITC_LAB_6
enter the number of symbols:4
The number of symbols are N:
     1     2     3     4

Enter the probabilities:[0.4 0.2 0.1 0.3]
The probabilities are:
    0.4000    0.2000    0.1000    0.3000

The sorted probabilities are:
    0.4000    0.3000    0.2000    0.1000

The average length of the code is:
    1.9000

Entropy is:
    1.8464

bits/msg
Efficiency is:
   97.1810

The codewords are:
     1     0     1     0     0     0     0     0     1

Decoded output is:
     1     2     3     4
```

ITC LAB – 7

AIM: “Finding The Arithmetic Encoding and Decoding For the User Defined Word”

CODE:

```
clc;
clear all;
prompt=' Enter the Arithmetic Word: ';
str=input(prompt,'s');
arith=str;
length1=size(str);
len=length1(2);
count=[];
disp('Arithmatic Encoding');
for i=1:len-1
count(i)=1;
for j=i+1:len
if str(i)==str(j)
str(j)=0;
count(i)=count(i)+1;
end
end
end
if(str(len)~=0)
count(len)=1;
end
j=1;


%Encoding

for i=1:len
if(str(i)~=0)
new(j)=str(i);
p(j)=count(i)/len;
if(j>1)
```

```
ar(j)=ar(j-1)+p(j);
else
ar(j)=p(j);
end
disp(['Aruthmetic Probability for ',str(i),' is ',num2str(p(j))]);
j=j+1;
end
end
arithmetic=size(new);
l=[];u=[];
l(1)=0;
u(1)=ar(1);
for i=2:len
for j=1:arithmetic(2)
if(arith(i)==new(j))
l(i)=l(i-1)+(u(i-1)-l(i-1))*(ar(j)-p(j));
u(i)=l(i-1)+(u(i-1)-l(i-1))*ar(j);
end
end
end
tag=(l(i)+u(i))/2;
disp(['The tag Value is = ',num2str(tag)]);


%Decoding

disp('Arithmatic Decoding');
rec='a';
tagr=tag;
for i=1:len
for j=1:arithmetic(2)
if(tagr<ar(j) && tagr>(ar(j)-p(j)))
rec(i)=new(j);
nm=j;
end
end
if(nm>1)
tagr=(tagr-ar(nm-1))/p(nm);
else
tagr=tagr/p(nm);
end
end

disp(['Decoded word is : ',rec]);
if(rec==arith)
disp('Succesfull');
else
end
```

OUTPUT:

```
 Enter the Arithmetic Word: hari
Arithmatic Encoding
Aruthmetic Probability for h is 0.25
Aruthmetic Probability for a is 0.25
Aruthmetic Probability for r is 0.25
Aruthmetic Probability for i is 0.25
The tag Value is = 0.10742
Arithmatic Decoding
Decoded word is : hari
Succesfull
```

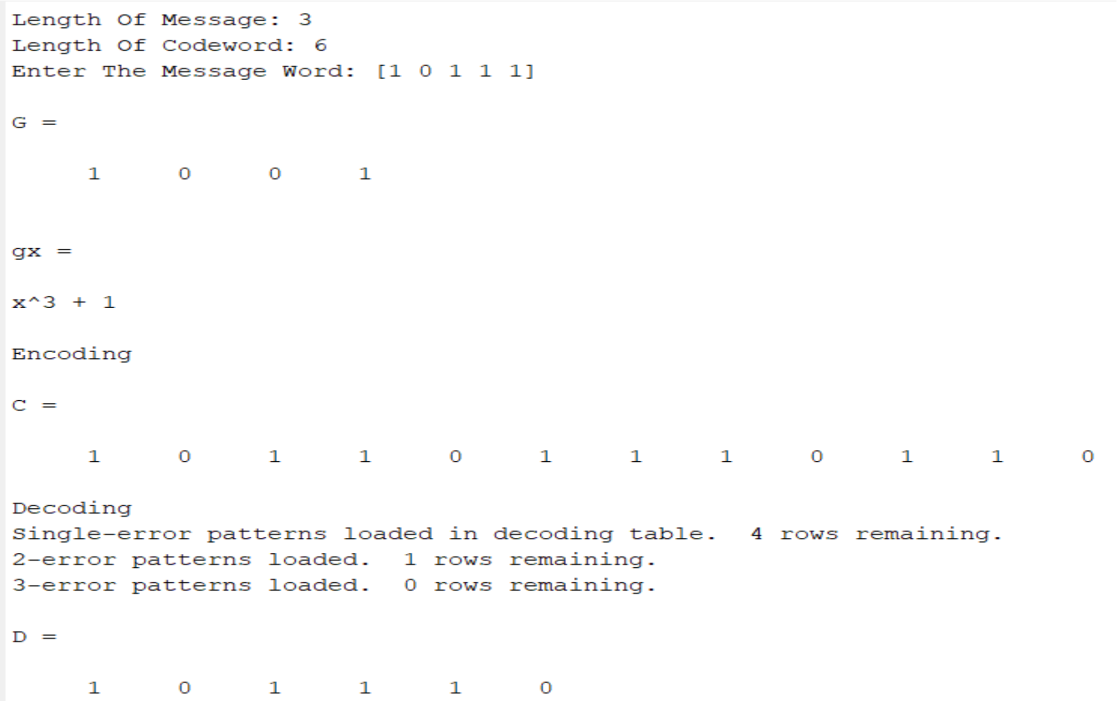
ITC LAB – 8

AIM: “Finding Cyclic Encoding & Decoding From User Defined Message”

CODE:

```
clc;
clear all;

k=input('Length Of Message: ');
n=input('Length Of Codeword: ');
m=input('Enter The Message Word: ');
G=cyclpoly(n,k,'max')
gx=poly2sym(G)

disp('Encoding')
C = encode(m,n,k,'cyclic',G)
disp('Decoding')
D = decode(C,n,k,'cyclic',G)
```

OUTPUT:



```
Length Of Message: 3
Length Of Codeword: 6
Enter The Message Word: [1 0 1 1 1]

G =

     1     0     0     1


gx =

x^3 + 1

Encoding

C =

     1     0     1     1     0     1     1     1     0     1     1     0

Decoding
Single-error patterns loaded in decoding table.  4 rows remaining.
2-error patterns loaded.  1 rows remaining.
3-error patterns loaded.  0 rows remaining.

D =

     1     0     1     1     1     0
```

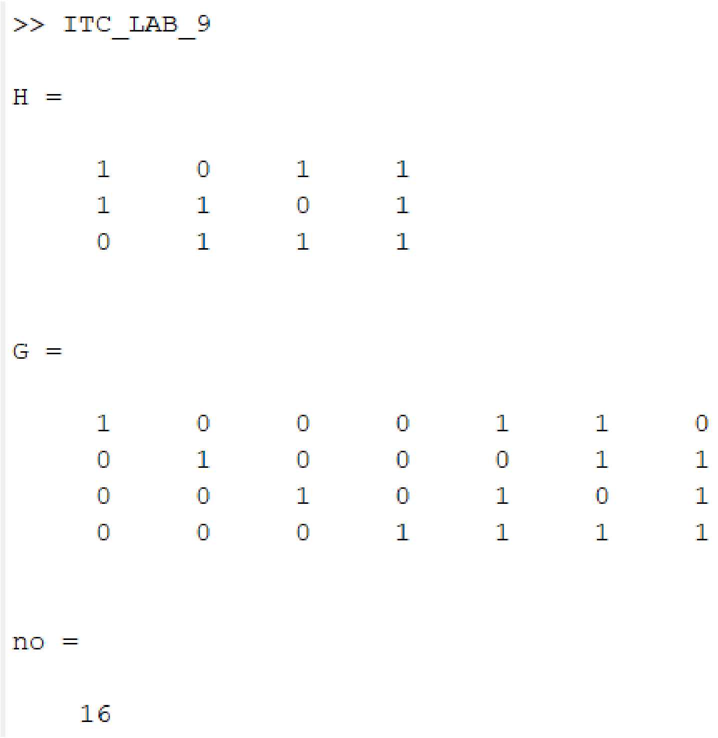
ITC LAB – 9

AIM: "Finding The Linear Block Encoding From The Given Binary Matrix"

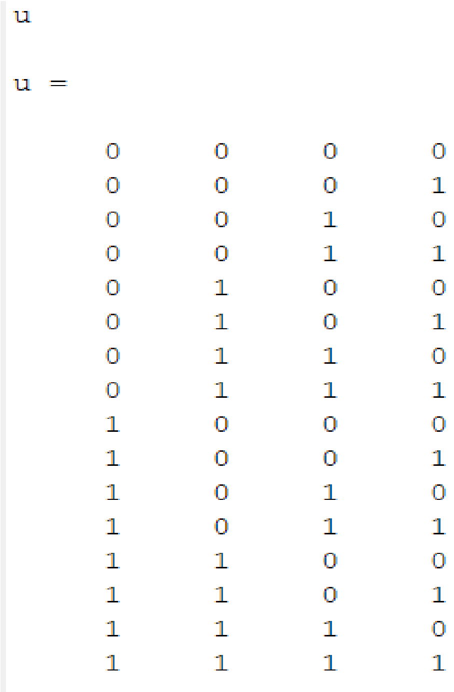
CODE:

```
H = [1 0 1 1;
     1 1 0 1;
     0 1 1 1]

k = 4;
n = 4;



% H Matrix Transpose
P = H';

%copy of H Transpose Matrix
L = P;

% Taking the last 4 rows of L and storing
L((5:4), : ) = [];

% Creating a Identity matrix of size K x K
I = eye(k);

% Making a 4 x 7 Matrix
G = [I L]

% Generate U data vector, denoting all information sequences
no = 2 ^ k

% Iterate through an Unit-Spaced Vector
for i = 1 : 2^k

  % Iterate through Vector with Specified Increment
  % or in simple words here we are decrementing 4 till we get 1
  for j = k : -1 : 1
    if rem(i - 1, 2 ^ (-j + k + 1)) >= 2 ^ (-j + k)
      u(i, j) = 1;
    else
      u(i, j) = 0;
    end

    % To avoid displaying each iteration/loop value
    echo off;
  end
end

echo on;
u

% Generate CodeWords
c = rem(u * G, 2)

% Find the min distance
w_min = min(sum((c(2 : 2^k, :))'))

% Given Received codeword
r = [0 0 0 1];
r

p = [G(:, n - k + 2 : n)];

%Find Syndrome
ht = transpose(H)

s = rem(r * ht, 2)

for i = 1 : 1 : size(ht)
  if(ht(i,1:3)==s)
    r(i) = 1-r(i);
    break;
  end
end

disp('The Error is in bit:')
disp(i)

disp('The Corrected Codeword is :')
disp(r)
```

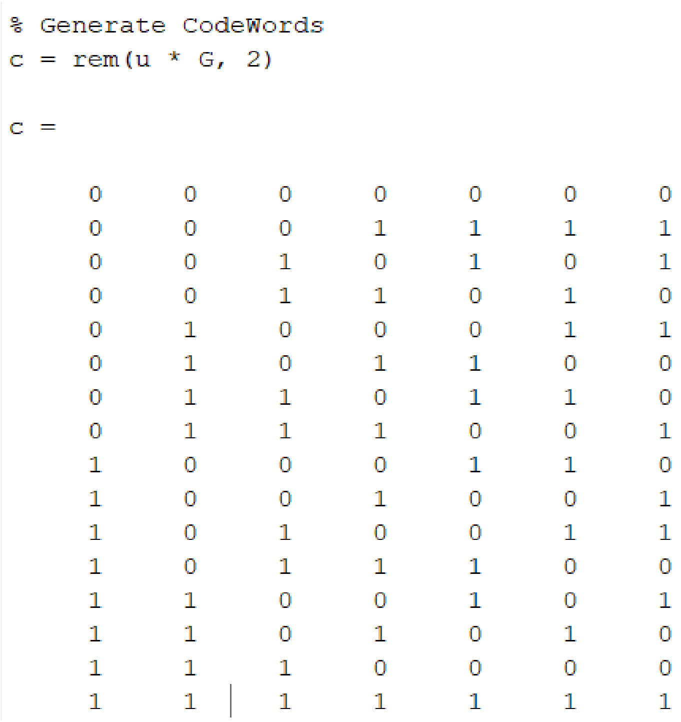
OUTPUT:



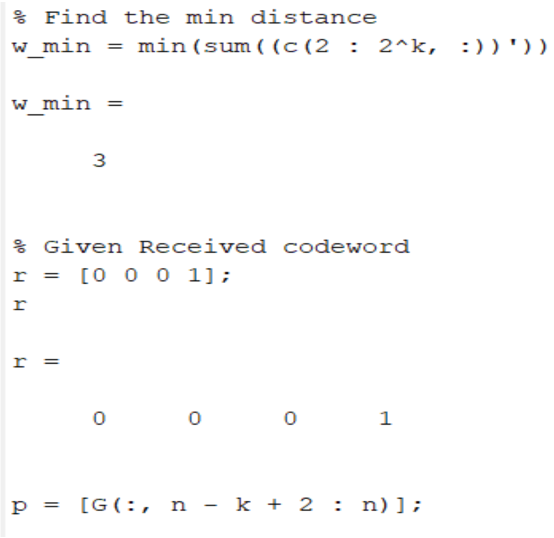
```
>> ITC_LAB_9

H =

     1     0     1     1
     1     1     0     1
     0     1     1     1


G =

     1     0     0     0     1     1     0
     0     1     0     0     0     1     1
     0     0     1     0     1     0     1
     0     0     0     1     1     1     1


no =

    16
```



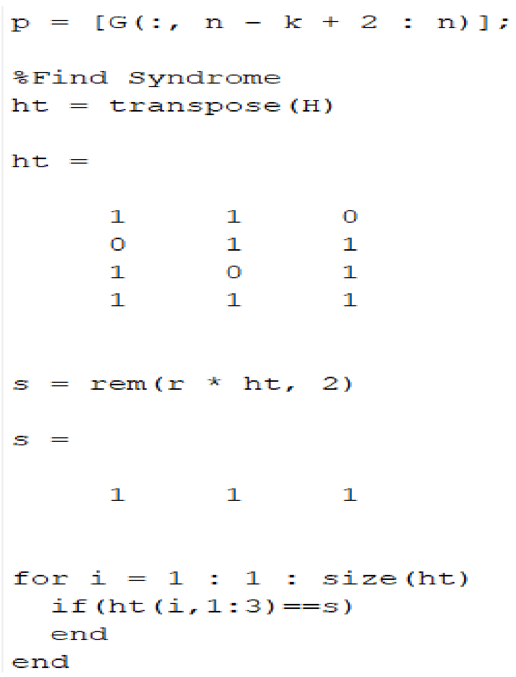
```
u

u =

     0     0     0     0
     0     0     0     1
     0     0     1     0
     0     0     1     1
     0     1     0     0
     0     1     0     1
     0     1     1     0
     0     1     1     1
     1     0     0     0
     1     0     0     1
     1     0     1     0
     1     0     1     1
     1     1     0     0
     1     1     0     1
     1     1     1     0
     1     1     1     1
```



```
% Generate CodeWords
c = rem(u * G, 2)

c =

     0     0     0     0     0     0     0
     0     0     0     1     1     1     1
     0     0     1     0     1     0     1
     0     0     1     1     0     1     0
     0     1     0     0     0     1     1
     0     1     0     1     1     0     0
     0     1     1     0     1     1     0
     0     1     1     1     0     0     1
     1     0     0     0     1     1     0
     1     0     0     1     0     0     1
     1     0     1     0     0     1     1
     1     0     1     1     1     0     0
     1     1     0     0     1     0     1
     1     1     0     1     0     1     0
     1     1     1     0     0     0     0
     1     1     1     1     1     1     1
```



```
% Find the min distance
w_min = min(sum((c(2 : 2^k, :))'))

w_min =

     3


% Given Received codeword
r = [0 0 0 1];
r

r =

     0     0     0     1


p = [G(:, n - k + 2 : n)];
```



```
p = [G(:, n - k + 2 : n)];

%Find Syndrome
ht = transpose(H)

ht =

     1     1     0
     0     1     1
     1     0     1
     1     1     1


s = rem(r * ht, 2)

s =

     1     1     1


for i = 1 : 1 : size(ht)
  if(ht(i,1:3)==s)
  end
end
```

```
  if(ht(i,1:3)==s)
  end
end
  if(ht(i,1:3)==s)
  end
end
  if(ht(i,1:3)==s)
    r(i) = 1-r(i);
    break;

disp('The Error is in bit:')
The Error is in bit:
disp(i)
     4


disp('The Corrected Codeword is :')
The Corrected Codeword is :
disp(r)
     0     0     0     0
```

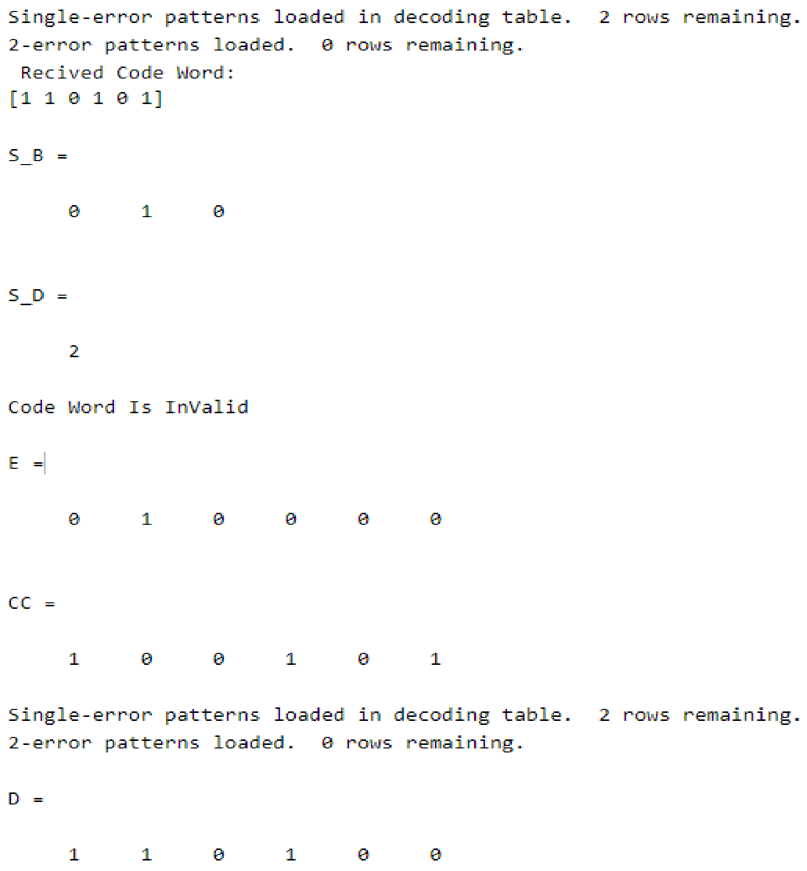
ITC LAB – 10

AIM: “Finding The Linear Block Decoding From The Given Binary Matrix”

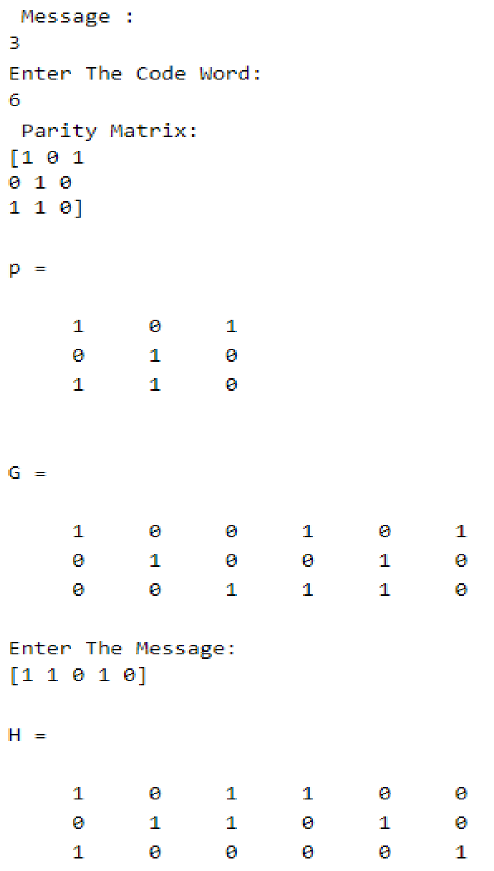
CODE:

```
clc;
clear all;
k= input('Enter The Message Word: ');
n=input('Enter The Code Word: ');
p=input('Enter The Parity Matrix: ')
G = [eye(k) p]
m=input('Enter The Message: ');
C=encode(m,n,k,'linear',G)

H= [p' eye(n-k )]
dtable=syndtable(H)
R=input('Enter The Recived Code Word: ');
S_B=rem(R*H',2)
S_D=bi2de(S_B,'left-msb')

if(S_D==0)
    disp('Code Word Is Valid')
else
    disp('Code Word Is InValid')
    E=dtable(S_D+1,:)
    CC=rem(R+E,2)
end

D=decode(C,n,k,'linear',G)
```

OUTPUT:



```
Single-error patterns loaded in decoding table.  2 rows remaining.
2-error patterns loaded.  0 rows remaining.
 Recived Code Word:
[1 1 0 1 0 1]

S_B =

     0     1     0


S_D =

     2

Code Word Is InValid

E =

     0     1     0     0     0     0


CC =

     1     0     0     1     0     1

Single-error patterns loaded in decoding table.  2 rows remaining.
2-error patterns loaded.  0 rows remaining.

D =

     1     1     0     1     0     0
```



```
 Message :
3
Enter The Code Word:
6
 Parity Matrix:
[1 0 1
0 1 0
1 1 0]

p =

     1     0     1
     0     1     0
     1     1     0


G =

     1     0     0     1     0     1
     0     1     0     0     1     0
     0     0     1     1     1     0

Enter The Message:
[1 1 0 1 0]

H =

     1     0     1     1     0     0
     0     1     1     0     1     0
     1     0     0     0     0     1
```

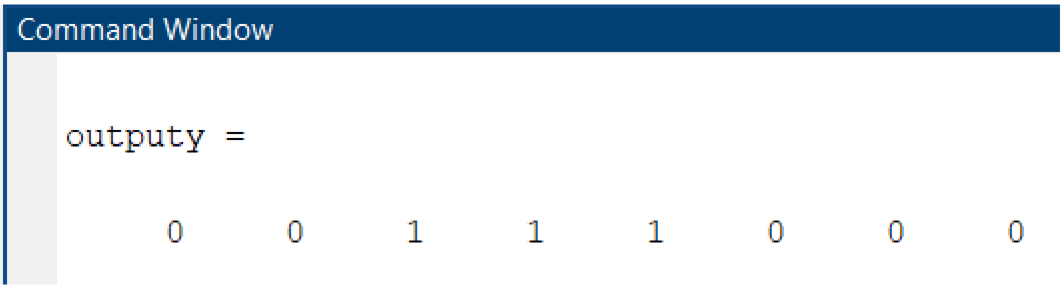
ITC LAB – 11

AIM: “Finding The Convolutional Encoding From The Given Binary Inputs”

CODE:

```
clc
clear
g=[1 1 1;1 0 1];%generator polynomials
[n,K] = size(g);
m = K-1;%number of registers
state = zeros(1,m);%set registers to zero
inputx=[0 1 0 1 1 1 0 0 1 0 1 0 0 0 1];%encoder input source code
[trash,h]=size(inputx);
outputy=[];
for x=1:h%h=number of input bits
    input=inputx(1,x);
for i=1:n
   output(i) = g(i,1)*input;
   for j = 2:K
       z=g(i,j)*state(j-1);
      output(i) = xor(output(i),z);
   end;
end
state = [input, state(1:m-1)];
outputy=[outputy,output];%new element added to sequence
end
output
```

OUTPUT:



```
outputy =

     0     0     1     1     1     0     0     0
```

THANK YOU :)