**WALSH CODE:**

clc;

clear all;

h=input('Enter the Kernel matrix : ');

d=input('Enter the square matrix dimension : ');

H=h;

D=log2(d);

for i=1:(D-1)

H1=h(1,1).\*H;

H2=h(1,2).\*H;

H3=h(2,1).\*H;

H4=h(2,2).\*H;

H=[H1 H2;H3 H4];

end

H=[H1 H2;H3 H4];

A=[];B=[];

for i=1:d

A = strvcat(A,sprintf('%+5.2g',H(i,:)));

end

A;

[x v]=size(A);

s=[];b=[];m=4;walsh\_code=[];

for j=1:x

for i=0:5:(v-1)

if (i == 0)

b=[b A(j,m)];

%b=[b [A(j,(i+1)) blanks(0) A(j,m)]];

end

if (i ~= 0) && (m ~= (v-1))

m=m+5;

b=[b [A(j,i) blanks(1) A(j,m)]];

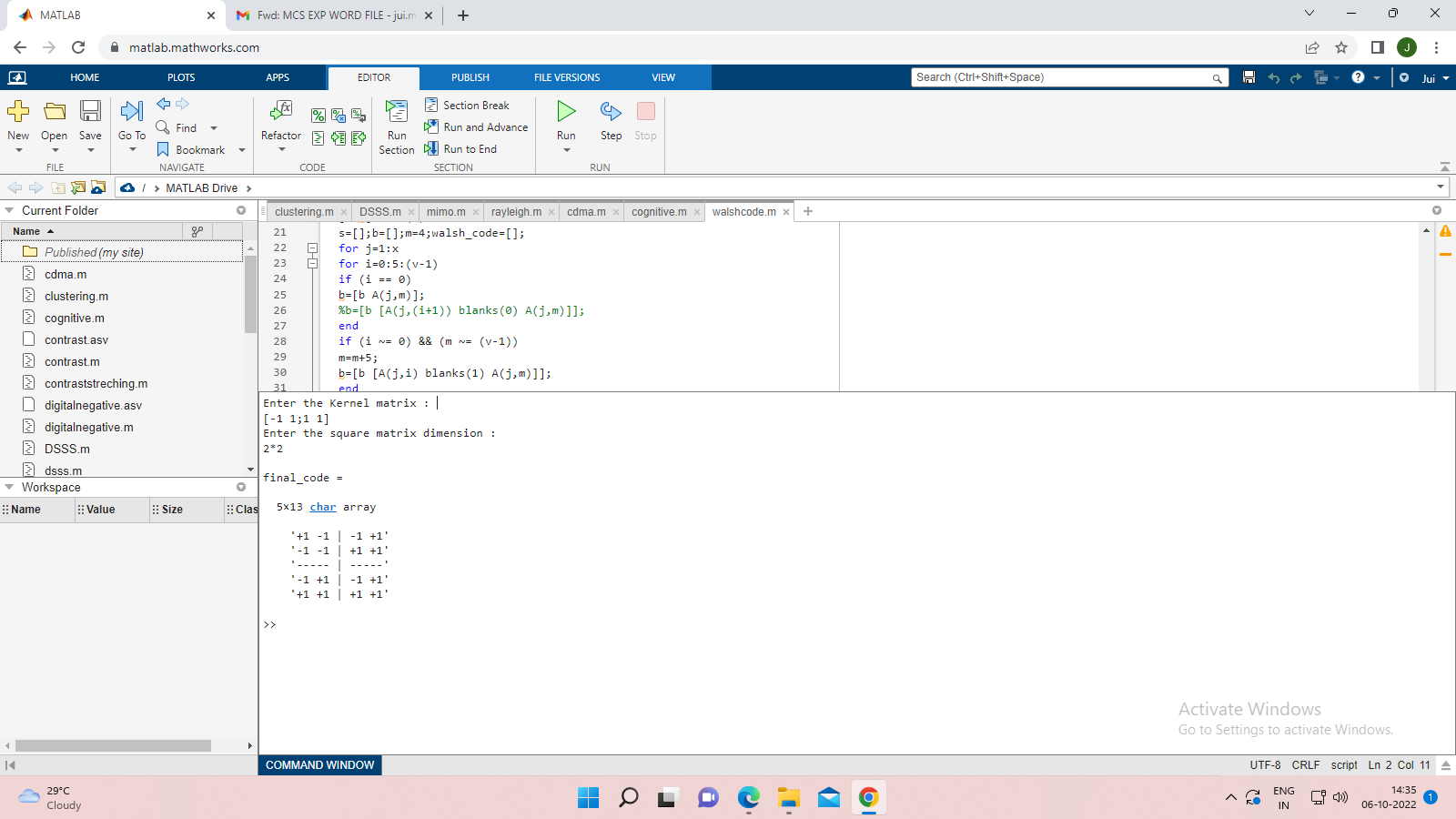
end

if (m == (v-1))

b=[b A(j,v)];

end

b;



**P-n sequence :**

clc

clear all

reg=input('Enter the initial state of register : ');

limit=length(reg);

REG=zeros(1,limit);

output\_codeword=[];

for i=1:((2^limit)-1)

REG=zeros(1,limit);

REG(1)=xor(reg(i,limit),reg(i,(limit-1)));

output\_codeword=[output\_codeword REG(1)];

for j=1:(limit-1)

REG(j+1)=reg(i,j);

end

reg=[reg;REG];

end

reg

output\_codeword

%Balance property

zeros=0;ones=0;

for i=1:((2^limit)-1)

if (output\_codeword(i) == 0)

zeros=zeros+1;

end

if (output\_codeword(i) == 1)

ones=ones+1;

end

end

zeros

ones

if (zeros == (ones-1))

disp('No. of zeros is equal to one less than No. of once hence code is satisfying the balance property')

else

disp('Since No. of zeros is not equal to one less than No. of once hence code is not satisfying the balance property')

end

