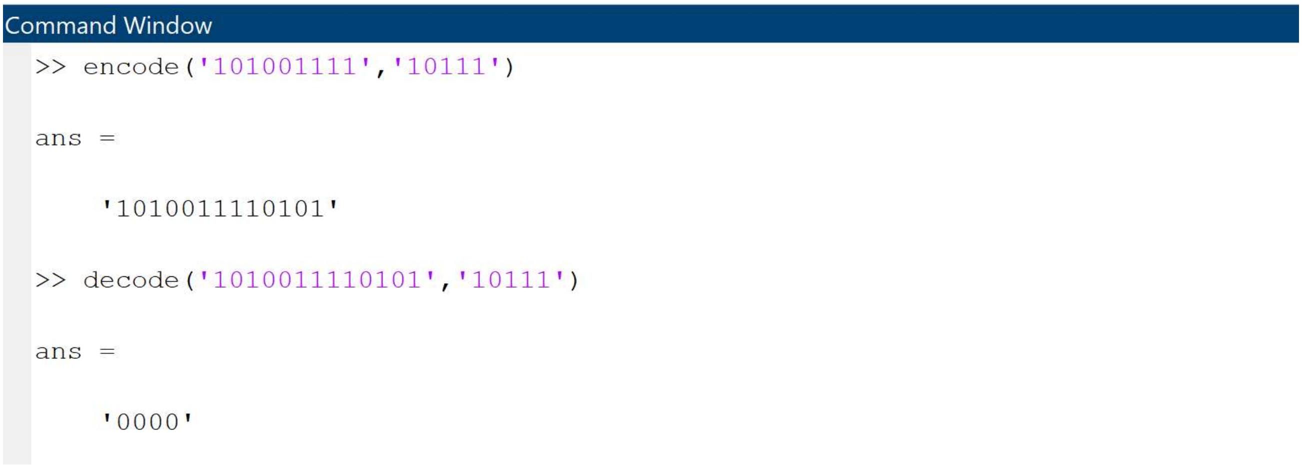
(a)encoder function

```
function codeword = encode(dataword,divisor)

dwL=strlength(dataword);            %dataword length
dvD=strlength(divisor)-1;           %divisor degree
dw=binstr2num(dataword);            %convert dataword(binary string) to decimal number
dv=binstr2num(divisor);             %convert divisor(binary string) to decimal number
dv = bitshift(dv,dwL-1);            %left shift divisor by length of dataword length-1

remainder = bitshift(dw,dvD);       %take initial remainder by left shifting the↙
dataword by length of divisor degree

for k = 1:dwL
    if bitget(remainder,dwL+dvD)
        remainder = bitxor(remainder,dv);       %bitwise xor operation

    end
    remainder = bitshift(remainder,1);          %updating the remainder
end
CRC_check_value = bitshift(remainder,-dwL);     %taking CRC by by right shifting final↙
remainder by length of dataword length
CRC=dec2bin(CRC_check_value);
if strlength(CRC)<dvD
    need=dvD-strlength(CRC);
    for j=1:need
        CRC=strcat('0',CRC);
    end
end
codeword=strcat(dataword,CRC);
end
```

Decoder function

```
function string = decode(codeword,divisor)
%we take input as codeword and divisor in binary string form and output
%syndrom in binary string format

dvD=strlength(divisor)-1;                        %divisor degree
cwL=strlength(codeword);                         %codeword length
cw=binstr2num(codeword);                         %codeword in decimal
dv=binstr2num(divisor);                          %convert divisor(binary string) to↙
decimal number
dwL=cwL-dvD;                                     %dataword length

dv=bitshift(dv,dwL-1);                           %divisor is left shifing by length of↙
dataword-1

for k = 1:dwL
    if bitget(cw,dwL+dvD)
        cw = bitxor(cw,dv);
    end
    cw = bitshift(cw,1);
end
cw=bitshift(cw,-dwL);
syndrome=dec2bin(cw);

if strlength(syndrome)<dvD
    need=dvD-strlength(syndrome);
    for j=1:need
        syndrome=strcat('0',syndrome);
    end
end
string=syndrome;
end
```

(b), (c)

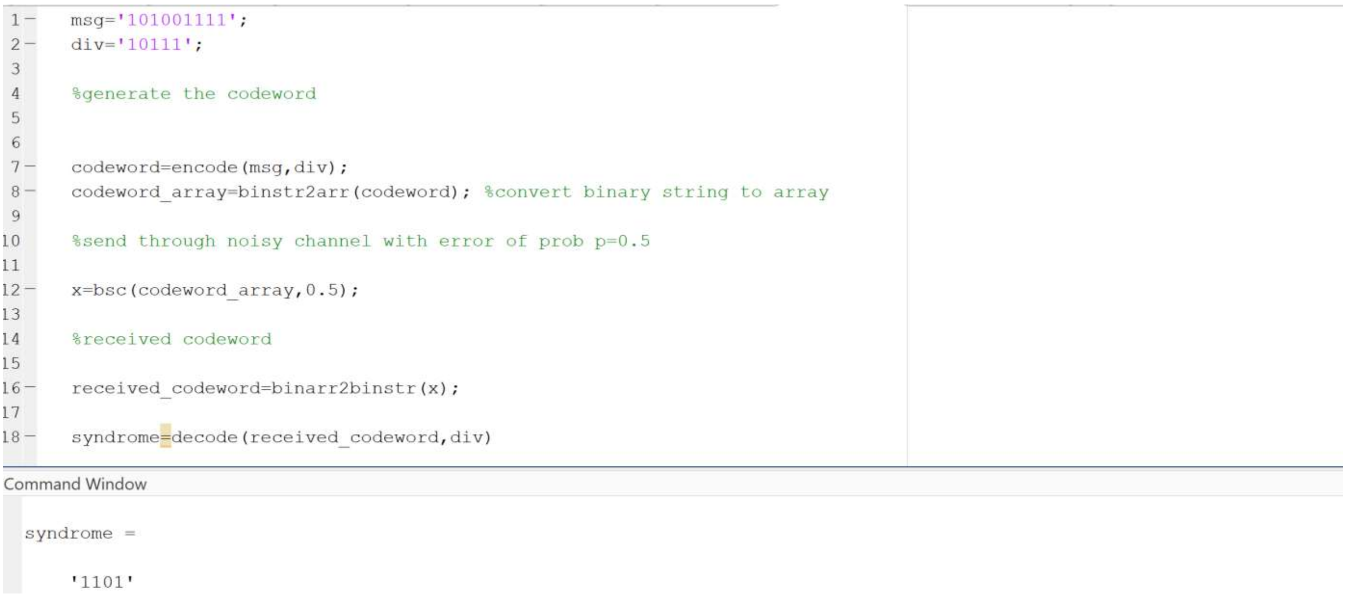


```
>> encode('101001111','10111')

ans =

    '1010011110101'

>> decode('1010011110101','10111')

ans =

    '0000'
```

Syndrome is equal to 0 that means received codeword doesn’t have ant errors in it.

(d), (e)

Part_1



```
msg='101001111';
div='10111';

%generate the codeword


codeword=encode(msg,div);
codeword_array=binstr2arr(codeword); %convert binary string to array

%send through noisy channel with error of prob p=0.5

x=bsc(codeword_array,0.5);

%received codeword

received_codeword=binarr2binstr(x);

syndrome=decode(received_codeword,div)
```

```
syndrome =

    '1101'
```

When the code word is send through the noisy channel syndrome is not equal to zero that means received codeword has errors in it.

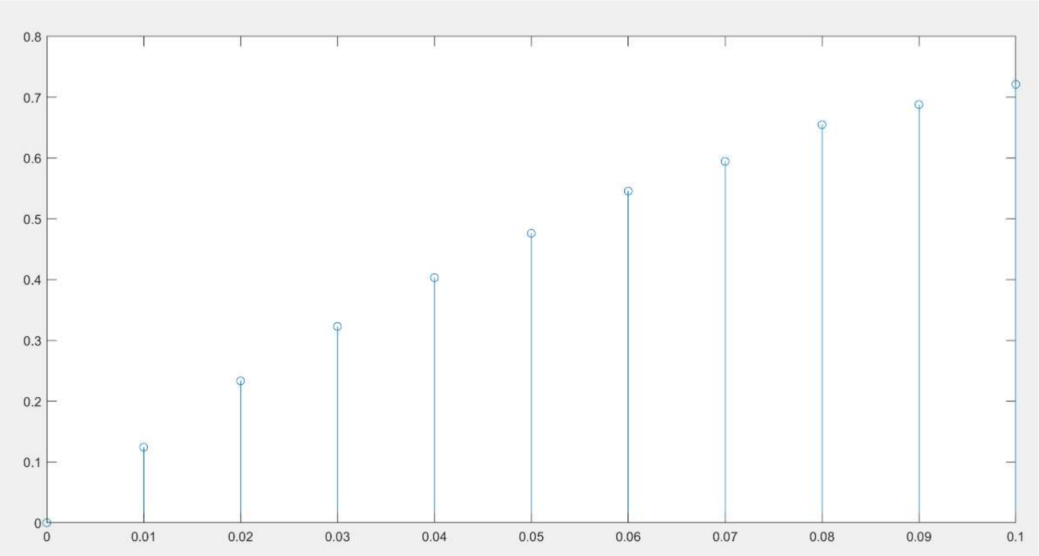
Part_2

Running the above code 10 times

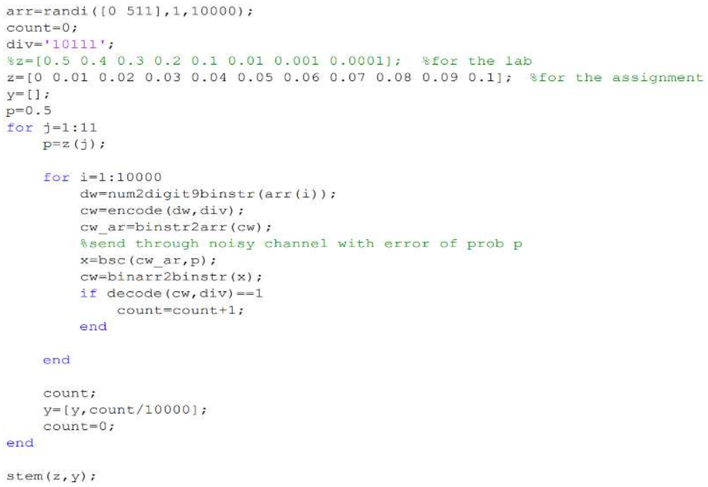
Sending Codeword	Received Codeword	Syndrome
‘1010011110101’	'1001011000011'	'0011'
‘1010011110101’	'0111001101010'	'0100'
‘1010011110101’	'0000110111001'	'0011'
‘1010011110101’	'0000001110110'	'0100'
‘1010011110101’	'0010110000010'	'1001'
‘1010011110101’	'0110011110001'	'1101'
‘1010011110101’	'1000101110111'	'1001'
‘1010011110101’	'0010011110011'	'1000'
‘1010011110101’	'0000001001010'	'0001'
‘1010011110101’	'0100111011011'	'0011'

Since the binary symmetric channel has an error probability of 0.5, if a codeword of n bits gets transmitted then the received codeword can have a maximum of n/2 errors. Thus, even if one bit gets errored then the syndrome will become non zero.

(f)



Code



```
arr=randi([0 511],1,10000);
count=0;
div='10111';
%z=[0.5 0.4 0.3 0.2 0.1 0.01 0.001 0.0001];  %for the lab
z=[0 0.01 0.02 0.03 0.04 0.05 0.06 0.07 0.08 0.09 0.1];  %for the assignment
y=[];
p=0.5
for j=1:11
    p=z(j);

    for i=1:10000
        dw=num2digit9binstr(arr(i));
        cw=encode(dw,div);
        cw_ar=binstr2arr(cw);
        %send through noisy channel with error of prob p
        x=bsc(cw_ar,p);
        cw=binarr2binstr(x);
        if decode(cw,div)==1
            count=count+1;
        end

    end

    count;
    y=[y,count/10000];
    count=0;
end

stem(z,y);
```

(g)Assumptions

1.Assume that time-out duration is reasonable and time-out will never occur
2.Assume that No packet lost will happen

The algorithm is implemented in the for loop in below code snippet.

(h)

```
arr=randi([0 65535],1,256);
reattempts=0;

for i=1:255
    %for one dataword
    time=0;                                     %time for one packet in microseconds
    p=0.00001;
    div='100000111';                            %CRC-8 divisor
    dw=binstr2binstr16(dec2bin(arr(i)));
    sn=i;
    rn=i;
    retrans_attempts=0;
    frame=tx_frame(dw,sn);
    %send through a noisy channel

    time=time+25+15;
    frame_array=binstr2arr(frame);              %convert binary string to array
    x=bsc(frame_array,p);

    %recieved frame
    recieved_frame=binarr2binstr(x);

    syndrome=decode(recieved_frame,div);

    if syndrome==0
        %accepts the frame
        rn=rn+1;
    else
        %neglect the frame
    end

    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%ack/nak sending stage

    ack_frame=rx_frame(recieved_frame);

    %send through a noisy channel

    time=time+25+15;

    ack_frame_array=binstr2arr(ack_frame);             %convert binary string to array
    x=bsc(ack_frame_array,p);

    %recieved ack frame
    recieved_ack_frame=binarr2binstr(x);

    syndrome=decode(strcat(recieved_ack_frame(1:16),recieved_ack_frame(25:32)),div);

    while syndrome~=0
        retrans_attempts=retrans_attempts+1;
        %send through a noisy channel
```

```
            time=time+25+15;
            frame_array=binstr2arr(frame);              %convert binary string to array
            x=bsc(frame_array,p);
            %recieved frame
            recieved_frame=binarr2binstr(x);

            syndrome=decode(recieved_frame,div);

            if syndrome==0
                if rn==sn
                    %accecpt the frame
                    rn=rn+1;
                else
                    %neglect the frame
                end
            else
                %neglect the frame
            end

            %%%%%%%%%%%%%%%%%%%%%%%%ack/nak sending phase

            ack_frame=rx_frame(recieved_frame);

            %send through a noisy channel

            time=time+15+25;
            ack_frame_array=binstr2arr(ack_frame);              %convert binary string to↙
    array
            x=bsc(ack_frame_array,p);

            %recieved frame
            recieved_ack_frame=binarr2binstr(x);

            syndrome=decode(strcat(recieved_ack_frame(1:16),recieved_ack_frame(25:32)),div);
        end
        reattempts=reattempts+retrans_attempts;
    end
```

Answers for (h),(i),(j)

Number of Datawords	Error probability	Results
256	0.00001	Total attempts=256 Expected number of transmissions=1 Expected number of retransmissions=0 Efficiency=31.25%
256	0.0001	Total attempts=257 Expected number of transmissions=1.00390625 Expected number of retransmissions=0.00390625 Efficiency=31.12840467%

Total attempts=256+number of retransmissions

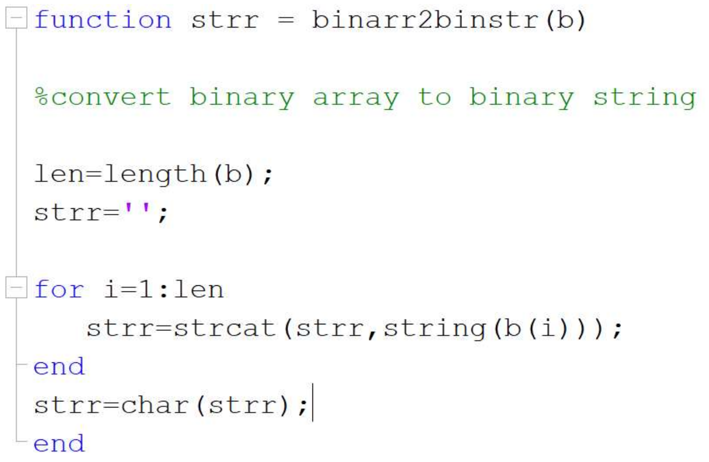
Expected number of transmissions=Total attempts/256

Expected number of retransmissions= number of retransmissions/256

Efficiency=(256*Dtp)/(Total attempts*S)   where S=2 x Dtp + 2 x Dp

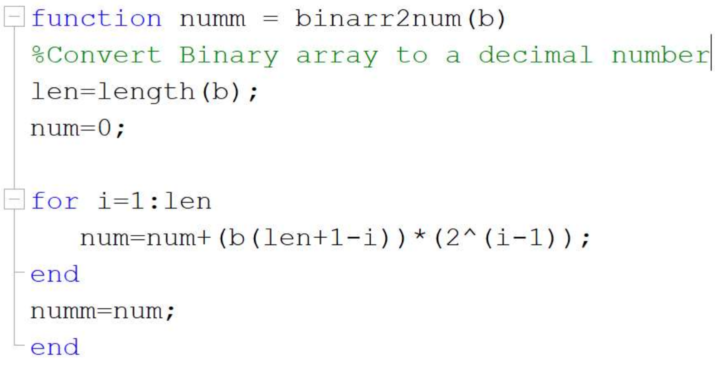
Codes of the supporting functions

1.



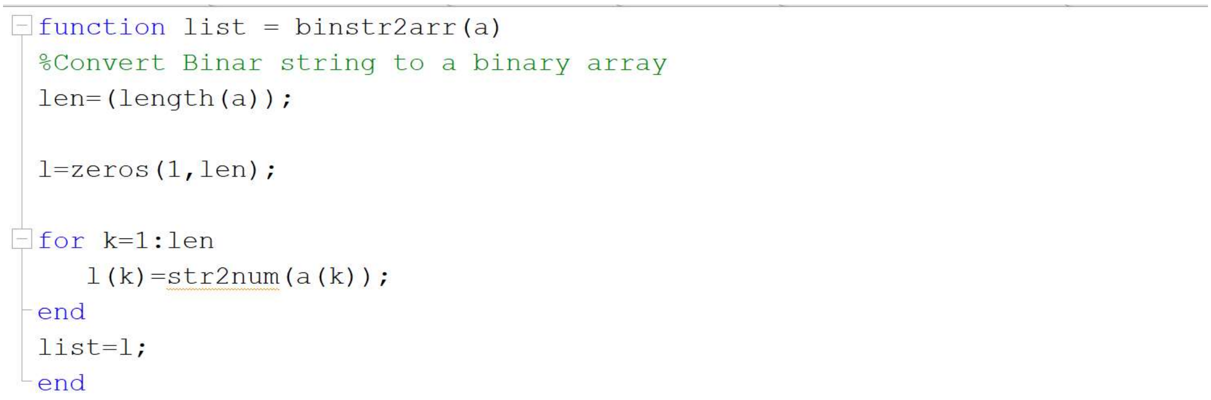
```
function strr = binarr2binstr(b)

%convert binary array to binary string

len=length(b);
strr='';

for i=1:len
    strr=strcat(strr,string(b(i)));
end
strr=char(strr);
end
```

2.



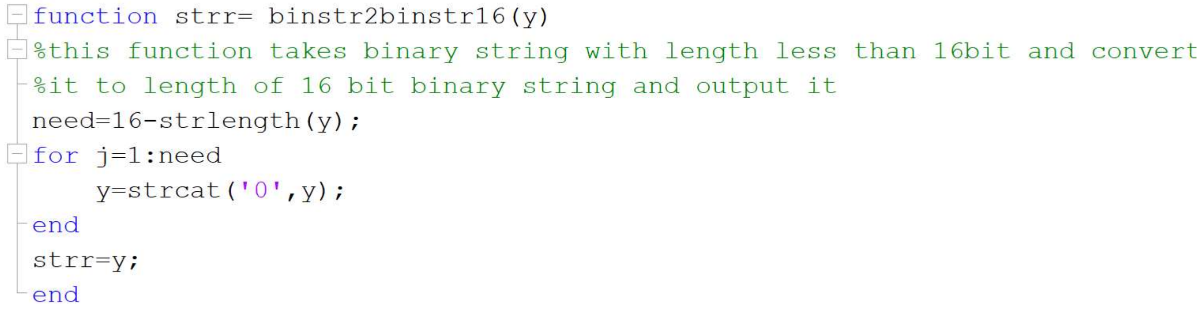
```
function numm = binarr2num(b)
%Convert Binary array to a decimal number
len=length(b);
num=0;

for i=1:len
    num=num+(b(len+1-i))*(2^(i-1));
end
numm=num;
end
```

3.



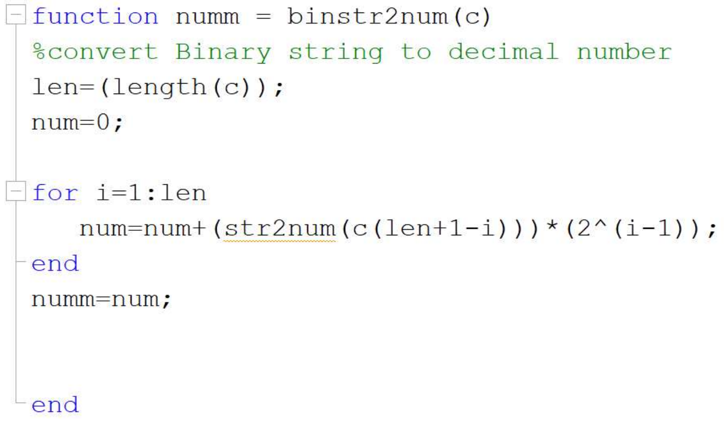
```
function list = binstr2arr(a)
%Convert Binar string to a binary array
len=(length(a));

l=zeros(1,len);

for k=1:len
    l(k)=str2num(a(k));
end
list=l;
end
```

4.



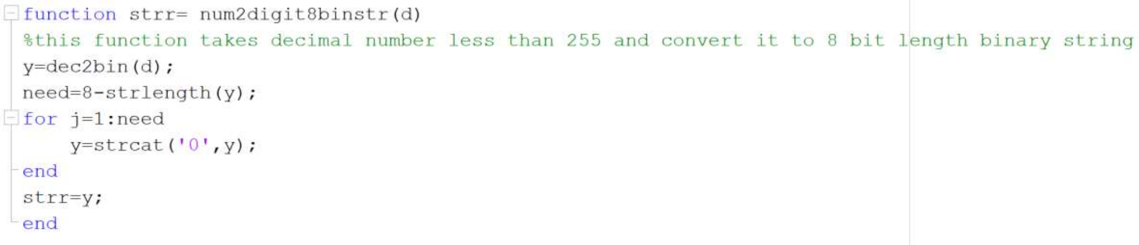
```
function strr= binstr2binstr16(y)
%this function takes binary string with length less than 16bit and convert
%it to length of 16 bit binary string and output it
need=16-strlength(y);
for j=1:need
    y=strcat('0',y);
end
strr=y;
end
```

5.



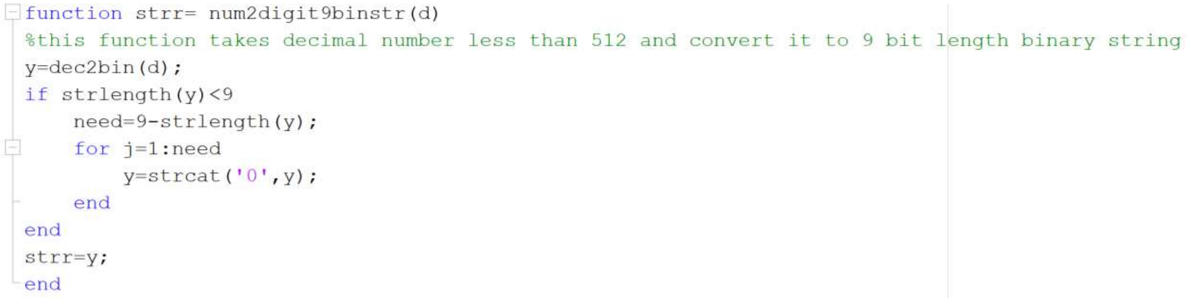
```
function numm = binstr2num(c)
%convert Binary string to decimal number
len=(length(c));
num=0;

for i=1:len
   num=num+(str2num(c(len+1-i)))*(2^(i-1));
end
numm=num;


end
```

6.



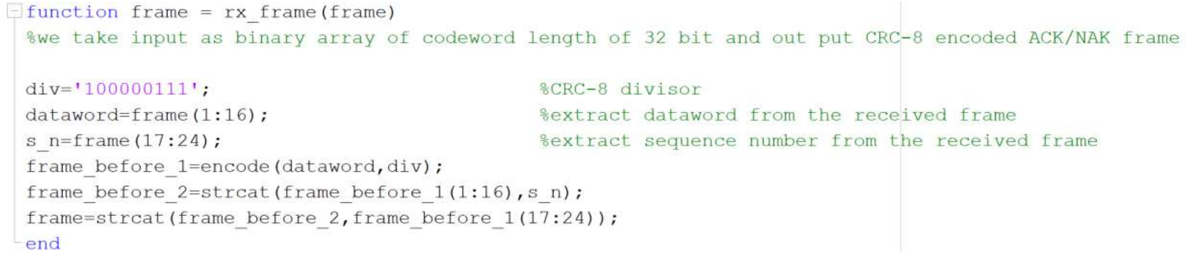
```
function strr= num2digit8binstr(d)
%this function takes decimal number less than 255 and convert it to 8 bit length binary string
y=dec2bin(d);
need=8-strlength(y);
for j=1:need
    y=strcat('0',y);
end
strr=y;
end
```

7.



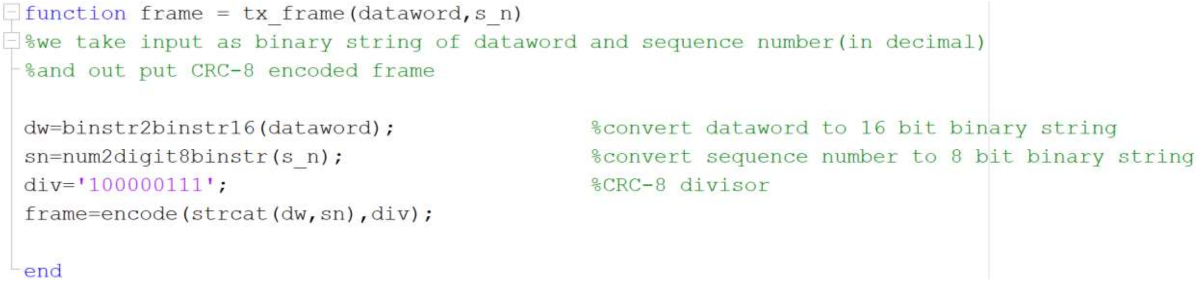
```
function strr= num2digit9binstr(d)
%this function takes decimal number less than 512 and convert it to 9 bit length binary string
y=dec2bin(d);
if strlength(y)<9
    need=9-strlength(y);
    for j=1:need
        y=strcat('0',y);
    end
end
strr=y;
end
```

8.



```
function frame = rx_frame(frame)
%we take input as binary array of codeword length of 32 bit and out put CRC-8 encoded ACK/NAK frame

div='100000111';                              %CRC-8 divisor
dataword=frame(1:16);                         %extract dataword from the received frame
s_n=frame(17:24);                             %extract sequence number from the received frame
frame_before_1=encode(dataword,div);
frame_before_2=strcat(frame_before_1(1:16),s_n);
frame=strcat(frame_before_2,frame_before_1(17:24));
end
```

9.



```
function frame = tx_frame(dataword,s_n)
%we take input as binary string of dataword and sequence number(in decimal)
%and out put CRC-8 encoded frame

dw=binstr2binstr16(dataword);               %convert dataword to 16 bit binary string
sn=num2digit8binstr(s_n);                   %convert sequence number to 8 bit binary string
div='100000111';                            %CRC-8 divisor
frame=encode(strcat(dw,sn),div);

end
```