APPENDIX A

MATLAB is a high-performance language for technical computing. It integrates computation, visualization, and programming in an easy-to-use environment where problems and solutions are expressed in familiar mathematical notation. Typical uses include Math and computation Algorithm development Data acquisition Modeling, simulation, and prototyping Data analysis, exploration, and visualization Scientific and engineering graphics Application development, including graphical user interface building

MATLAB is an interactive system whose basic data element is an array that does not require dimensioning. This allows you to solve many technical computing problems, especially those with matrix and vector formulations, in a fraction of the time. MATLAB has evolved over a period of years with input from many users. In university environments, it is the standard instructional tool for introductory and advanced courses in mathematics, engineering, and science. In industry, MATLAB is the tool of choice for high-productivity research, development, and analysis. MATLAB features a family of add-on application-specific solutions called toolboxes. Very important to most users of MATLAB, toolboxes allow you to learn and apply specialized technology. Toolboxes are comprehensive collections of MATLAB functions (M-files) that extend the MATLAB environment to solve particular classes of problems. Areas in which toolboxes are available include signal processing, control systems, neural networks, fuzzy logic, wavelets, simulation, and many others. Encoding and Decoding Source code for Text is given below which takes the binary file as an input which is encoded to compressed file. Decoding source code reads compressed file and convert it to original file.

**Encoding Source code**

```
fid1=fopen('gtext1.m','r'); /* open gtext1 file which is binary file*/
fid3=fopen('protext1.m','w'); /* open protext1 file which is binary
file with frame of 32 */
a=length file;
```
k = 32;
b = a / k;
b1 = fix(b);
bb1 = k * b1;
b2 = mod(a, k)
b3 = a - k

while 1
    tline = fgetl(fid1); /* reads one frame as one line
    vv = length(tline)
    if ~ischar(tline), break, end
    i = 1;
    x = k;
    while (x <= 650)
        c = tline(i:x);
        fprintf(fid3, '%s
', c);
        cc = length(c);
        i = x + 1;
        x = x + k;
    end
    c = tline(bb1 + 1:650);
    fprintf(fid3, '%s
', c);
    end
    fclose(fid1);
    fclose(fid3);
cle, clear all

fid3 = fopen('protext1.m', 'r'); /* opens written protext1 in read mode
to enter in encoding algorithm*/
fid2 = fopen('resprotext132.m', 'w'); /* Result of encoding is written in resprotext1*/
while 1
    tline = fgetl(fid3);
    if ~ischar(tline), break, end

150
c = tline;
len = length(tline);
if len == mod()
j=1;ss='00';p=0;
if c(j)=='0'
    while c(j)=='0'
        p=p+1;
        j=j+1;
        if j>32
            break;
    end
end
else
    j=1; ss = '01'; p=0;
    while c(j)=='1'
        p=p+1;
        j=j+1;
        if j>32
            break;
    end
end

if(p==32)
p5=[ss];
p3=length(p5);
else
    if (p<32)
        p2=c(p+1:cc);
p5=[ss p2];
p3=length(p5);
    end
end

len=cc;ss='10';len1=0;
if (c(len)=='0')
    while c(len)=='0'
        len1=len1+1;
        len=len-1;
        if len<1
            break;
        end
    end
else
    s=cc; ss='11'; len1=0;
    while c(s)=='1'
        len1=len1+1;
        s=s-1;
        if s<1
            break;
        end
    end
z=cc-len1;
V2=c(1:z);
V5=[ss v2];
V3=length(V5);
    % to get the minimum value from comparison
if (p3 <= v3)
    en=[p5];
    fprintf (fid2,'%s\n',p5);
else
    en=[v5];
    fprintf(fid2,'%s\n',v5);
end
len1=length(tline);
cc=len;
else
%% scan from left for 0
j=1; ss='00'; p=0;
if c(j)='0'
    while c(j)='0'
        p=p+1;
        j=j+1;
        if j>32
            break;
    end
else
    j=1; ss='01'; p=0;
    while c(j)='1'
        p=p+1;
        j=j+1;
        if j>32
            break;
    end
endif
if(p==32)
p5 = [ss];
p3 = length(p5);
else
    if (p<32)
p2 = c(p+1:cc);
p5 = [ss p2];
p3 = length(p5);
end
endif
len = cc ; ss='10'; len1=0;
if (c(len)=='0')
    while c(len)='0'
end
len1=len1+1;
len=len-1;
if len<1
    break;
end
end

else
    s=cc; ss='1'; len1=0;
    while c(s)=='1'
        len1=len1+1;
        s=s-1;
        if s<1
            break;
        end
    end
end

z=cc-len1;
v2=c(1:z);
v5=[ss v2];
v3=length(v5);

% to get the minimum value from comparison
if (p3 <= v3)
    en = [p5];
    fprintf(fid2,'%s
',p5);
else
    en=[v5];
    fprintf(fid2,'%s
',v5);
end
end
end
fclose (fid2);
fclose (fid3);
Decoding Source code

clear all
fid=fopen('resprotext132.m','r'); /* opens resprotext132 in read
mode to enter in decompression algorithm*/
 fid1=fopen('decomtext1.m','w'); /* opens decomtext1 as an
intermediate file*/
while ~feof(fid)
tline = fgetl(fid);
if ~ischar(tline), break, end
l=length(tline);
x=tline(1:2);
if x=='00'
    a=0;
    x1=tline(3:l);
    x2 = length(x1);
    x3 = (32-x2);
p=1;
    while p<=x3
        a(p) ='0';
        y = [a x1];
        p=p+1;
    end
end
end
if x=='01'
    b=0;
    x1=tline(3:l);
    x2=length(x1);
    x3=(32-x2);
p=1;
    while p<=x3
        b(p) ='1';
    end
end
\begin{verbatim}
y=[b x1];
p=p+1;
end
end
if x=='10'
c=0;
x1=tline(3:l);
x2=length(x1);
x3=(32-x2);
p=1;
while p<=x3
c(p)='0';
y=[x1 c];
p=p+1;
end
end
if x=='11'
d=0;
x1=tline(3:l);
x2=length(x1);
x3=(32-x2);
p=1;
while p<=x3
d(p)='1';
y=[x1 d];
p=p+1;
end
fprintf(fid1, '\%s \\
y');
end
fclose(fid);
fclose (fid1);
clc, clear all, close all
\end{verbatim}
fid1=fopen ('decomtext1.m','r'); /* opens intermediate decomtext1 file to get original binary file */
fid2=fopen ('xx.m','w'); /* opens xx file to get original binary file */
f=fread (fid1);
h=char (f);
l1 = length (h);
i=1;
while i<=l1
z = h (i);
if abs(z) == 32;
i = i+1;
else
fwrite(fid2,z);
end
i=i+1;
end
fclose(fid1);
fclose (fid2);

clc, close all, clear all
fid3=fopen ('xx.m','r');
fid4=fopen ('decompressedtext1.m','w'); /* original text file obtained in decompressedtext1 from xx file */
tline = fgetl(fid3);
vv =length(tline);

i =1;
x=650;

while (x<=650)
c=tline (i : x);
fprintf (fid4, '%s\n', c);
cc =length(c);
i=x+1;
x=x+650;
APPENDIX A

k=x;
end

fclose (fid3);
fclose (fid4);
Source code for Run length technique
clc,clear,close all
fid=fopen('grice.m','r');
fid1=fopen('runrice.m','w');
while 1
    tline = fgetl(fid);
    l=length(tline)
    if ~ischar(tline), break, end
    l=length(tline);
    l1=1;
    i=1;
    j=1;
k=1;
    while i<l
        j=i+1;
        while tline(i)==tline(j) & j<l-1
            l1=l1+1;
            j=j+1;
        end
        if i+1==l
            l1=l1+1;
        end
        b(k)=tline(i);
        L(k)=l1;
        fprintf(fid1,'%d%s
', L(k), b(k));
        l1=1;
        k=k+1;
        i=j;
    end
end
a=[L;str2num(b)]
fclose(fid);
fclose(fid1);
APPENDIX B

DEMO CREATED FOR THE PROPOSED COMPRESSION TECHNIQUE

The presented Demo takes the input of the file given in edit box, ask for the name of compressed file. By pressing the compress, button the compressed file with given name is created which in turn can be decompressed.
APPENDIX C

We are proposing an approach for cryptography. It deals with the binary form of the text. The encoding algorithm reads the binary file and converts it into cipher text. The Decoding algorithm takes the cipher text and gets back the original file.

Source code for cryptographic algorithm

```c
fid=fopen('1.m','r'); /* Reads the Input Text file*/
fid1=fopen ( '13.m','w'); /* Binary form is written to the file*/
f=fread(fid);
h=abs(f);
n=dec2bin(h);
p=n';
p1=p(1,:);
j=length(p1);
a=85;
i=1;
x=14;
while(i<=j)
b=n(i:x);
b1=bin2dec(b);
w=( b1/a);
w1=mod(b1,a);
quo=dec2bin(w);
z=length(quo);
z1=dec2bin(z,4);
remainder=dec2bin(w1);
r=length(remainder);
r1=dec2bin(r,4);
i=x+1;
x=i+13;
```
enco=[z1 quo r1 remainder];
fprintf(fid1,%s/n,enco);
disp (enco)
end
fclose(fid)
fclose(fid1)
clc,clear all, close all
fid=fopen('t3.m',r); /* binary file is read*/
fid1=fopen('cryp.m',w); /* converted to crypted form*/
while ~feof(fid)
f=fgetl(fid);
f1=length(f);
if(f1<21)
c=f(1:7);
d=bin2dec(c);
d1=char(d);
disp(d1)
fwrite(fid1,d1);
c=f(8:14);
d=bin2dec(c);
d1=char(d);
disp(d1)
fwrite(fid1,d1);
d2=f(15:f1);
disp(d2)
fwrite(fid1,d2);
end
if(f1>21)
c=f(1:7);
d=bin2dec(c);
d1=char(d);
fwrite(fid1,d1);
disp(d1)
c=f(8:14);
d=bin2dec(c);
d1=char(d);
disp(d1)
fwrite(fid1,d1);
c=f(15:21);
d=bin2dec(c);
d1=char(d);
disp(d1)
fwrite(fid1,d1);
d2=f(22:f1);
disp(d2)
fwrite(fid1,d2);
end
if(f1==21)
c=f(1:7);
d=bin2dec(c);
d1=char(d);
fwrite(fid1,d1);
disp(d1)
c=f(8:14);
d=bin2dec(c);
d1=char(d);
disp(d1)
fwrite(fid1,d1);
c=f(15:21);
d=bin2dec(c);
d1=char(d);
disp(d1)
fwrite(fid1,d1);
end
end
fclose(fid);
fclose(fid1);
clc,clean_all, close_all
fid=fopen('cryp.m','r'); /* crypted file is read*/
fid1=fopen('crypt.m','w'); /* binary intermediate file is created*/
while ~feof(fid)
    f=fgetl(fid);
    h=char(f);
    h1=length(h);
    i=1;
    while(i<=h1)
        if((h(i)=='0')|| (h(i)=='1'))
            d=(h(i));
            fprintf(fid1,'%s\n',d);
        else
            q=abs(h(i));
            q1=dec2bin(q)
            fprintf(fid1,'%s\n',q1);
        end
        i=i+1;
    end
    f=f+1;
end
fclose(fid);
fclose(fid1);
clear close all, clc
fid5=fopen('crypt.m','r'); /* binary file is read*/
fid6=fopen('dcm1.m','w'); /* binary file is aligned properly*/
a=85;
i=1;
while ~feof(fid5)
    f=fgetl(fid5);
    q1=bin2dec(f);
    f=fgetl(fid5);
    r1= bin2dec(f);
    deco = (a*q1+r1);
deco1 = dec2bin(deco);
deco2 = length(deco1);
if (deco2 < 14)
  p = 14 - deco2;
  k = 1; s = '0';
while (k <= p)
  s(k) = '0';
  k = k + 1;
  deco3 = [s deco1];
end
fprintf(fid6, '%s
', deco3);
else
  fprintf(fid6, '%s
', deco1);
end
end
fclose(fid5);
close(fid6);
clc, clear all
fid = fopen('dce1.m', 'r'); /* aligned Binary file is read */
fid1 = fopen('dce2.m', 'w'); /* Original file is obtained */
while ~feof(fid)
  f = fgets(fid);
  c = f(1:7);
  c1 = bin2dec(c);
  c2 = char(c1);
  fwrite(fid1, c2);
  c3 = f(8:14);
  c4 = bin2dec(c3);
  c5 = char(c4)
  fwrite(fid1, c5);
end
fclose(fid); fclose(fid1);