APPENDIX

Code to get the seed value

```
1. private int modKunci(String kunci)
2. {
3.     int hasil=kunci.charAt(0);
4.     for(int i=0;i<kunci.length()-1;i++)
5.     {
6.         hasil=hasil^kunci.charAt(i+1);
7.     }
8.     return hasil;
9. }
```

Code to generate random numbers with LCG

```
10.   private int[] LCG(int hasilkunci)
11.   {
12.       int a=49;
13.       int c=7;
14.       int gen=4;
15.       int m=120;
16.       int tampung=hasilkunci;
17.       int hasil[]=new int[gen];
18.       for(int i=0;i<gen;i++)
19.       {
20.           tampung=(a*tampung+c)%m;
21.           hasil[i]=tampung;
22.       }
23.       return hasil;
24.   }
```

Code to convert generated random numbers to binary

```
25.   private int[] biner_lcg(int[] Pseudo)
26.   {
27.       int [] bit_lcg=new int[Pseudo.length*8];
28.       int k=0;
29.       for(int i=0;i<Pseudo.length;i++)
30.       {
31.                                                    String
  Pseudo_biner=Integer.toBinaryString(Pseudo[i]);
32.           while(Pseudo_biner.length()!=8)
33.           {
34.               Pseudo_biner='0'+Pseudo_biner;
35.           }
36.           for(int j=0;j<Pseudo_biner.length();j++)
37.           {
```

```
38.                 bit_lcg[k]=Integer.parseInt(String.valueOf(Pseud
  o_biner.charAt(j)));
39.                 k++;
40.             }
41.         }
42.         return bit_lcg;
43.     }
```

Code to convert message to binary

```
44.     private int[] bit(String bit_s)
45.     {
46.         int j = 0;
47.         int [] b_s = new int[bit_s.length()*8];
48.         for(int i=0;i<bit_s.length();i++)
49.         {
50.             int x = bit_s.charAt(i);
51.             String x_s = Integer.toBinaryString(x);
52.             while(x_s.length()!=8)
53.             {
54.                 x_s = '0'+x_s;
55.             }
56.             for(int k=0;k<8;k++)
57.             {
58.                                                 b_s     [j]     =
  Integer.parseInt(String.valueOf(x_s.charAt(k)));
59.                 j++;
60.             }
61.         }
62.         return b_s;
63.     }
```

Code to spread the binary of message with 4 scale

```
64.     private int[] Spread(int[] bit_pesan)
65.     {
66.         int spread = 4;
67.         int [] bit_spread = new int[bit_pesan.length*spread];
68.         int k=0;
69.         for(int i=0;i<bit_pesan.length;i++)
70.         {
71.             for(int j=0;j<spread;j++)
72.             {
73.                 bit_spread[k]=bit_pesan[i];
74.                 k++;
75.             }
76.         }
77.         return bit_spread;
78.     }
```

Code to XOR the binary of spreaded message with the binary of generated random numbers

```
79.   private int[] ModPesanLCG(int[] bit_lcg, int[] bit_spread)
80.   {
81.       int k=0;
82.       int [] hasil_spread=new int[bit_spread.length];
83.       for(int i=0;i<bit_spread.length;i++)
84.       {
85.           if(k<bit_lcg.length)
86.           {
87.               hasil_spread[i]=bit_spread[i]^bit_lcg[k];
88.               k++;
89.           }
90.           else
91.           {
92.               k=0;
93.               hasil_spread[i]=bit_spread[i]^bit_lcg[k];
94.               k++;
95.           }
96.       }
97.       return hasil_spread;
98.   }
```

Code to embed the result of code above into the image

```
99.   private void embed(int[] hasil_spread, String ci)
100.  {
101.      BufferedImage CoverGambar;
102.      int width;
103.      int height;
104.      try
105.      {
106.          File image = new File(ci);
107.          CoverGambar = ImageIO.read(image);
108.          width = CoverGambar.getWidth();
109.          height = CoverGambar.getHeight();
110.          int count = 0;
111.          String red,green,blue;
112.          String rednew = null;
113.          String greennew = null;
114.          String bluenew = null;
115.          for(int i=0;i<height;i++)
116.          {
117.              if(count!=hasil_spread.length)
118.              {
119.                  for(int j=0;j<width;j++)
120.                  {
121.                                                  Color  c  =  new
  Color(CoverGambar.getRGB(j,i));
```

```
122.                                                                red   =
  Integer.toBinaryString(c.getRed());
123.                                                              green   =
  Integer.toBinaryString(c.getGreen());
124.                                                               blue   =
  Integer.toBinaryString(c.getBlue());
125.                        if(count!=hasil_spread.length)//red
126.                        {
127.                                                             rednew   =
  red.substring(0,red.length()-
  1)+Integer.toString(hasil_spread[count]);
128.                                                 Color newColor = new
  Color(Integer.parseInt(rednew,2),c.getGreen(),c.getBlue());
129.                                              CoverGambar.setRGB(j, i,
  newColor.getRGB());
130.                            count++;
131.                        }
132.                        if(count!=hasil_spread.length)//green
133.                        {
134.                                                           greennew   =
  green.substring(0,green.length()-
  1)+Integer.toString(hasil_spread[count]);
135.                                                 Color newColor = new
  Color(Integer.parseInt(rednew,2),Integer.parseInt(greennew,2),c
  .getBlue());
136.                                              CoverGambar.setRGB(j, i,
  newColor.getRGB());
137.                            count++;
138.                        }
139.                        if(count!=hasil_spread.length)//blue
140.                        {
141.                                                            bluenew   =
  blue.substring(0,blue.length()-
  1)+Integer.toString(hasil_spread[count]);
142.                                                 Color newColor = new
  Color(Integer.parseInt(rednew,2),Integer.parseInt(greennew,2),I
  nteger.parseInt(bluenew,2));
143.                                              CoverGambar.setRGB(j, i,
  newColor.getRGB());
144.                            count++;
145.                        }
146.                        else
147.                        {
148.                            break;
149.                        }
150.                    }
151.                }
152.                else
153.                {
154.                    break;
155.                }
156.            }
157.            String namafile = ci.substring(0,ci.length()-4);
```

```
158.                 String typefile = ci.substring(ci.length()-
  4,ci.length());
159.           namafile = namafile+"_stegano.png";
160.           File output = new File(namafile);
161.           ImageIO.write(CoverGambar, "png", output);
162.           mse_psnr(ci,namafile);
163.       }
164.       catch(IOException e)
165.       {
166.                     JOptionPane.showMessageDialog(this,"Gagal
  Menyisipkan","Error",JOptionPane.ERROR_MESSAGE);
167.       }
168.   }
```

Code PSNR

```
169.   private void mse_psnr(String ci, String namafile)
170.   {
171.       BufferedImage CoverGambar,SteganoGambar;
172.       int width;
173.       int height;
174.       double mse = 0,mse_r = 0,mse_g = 0,mse_b = 0;
175.       double psnr = 0,psnr_r = 0,psnr_g = 0,psnr_b = 0;
176.       double max_rc = 0,max_gc = 0,max_bc = 0;
177.       double max_rs = 0,max_gs = 0,max_bs = 0;
178.       DecimalFormat dfmse = new DecimalFormat("#.##########");
179.       DecimalFormat dfpsnr = new DecimalFormat("#.##");
180.       try
181.       {
182.           File image1 = new File(ci);
183.           File image2 = new File(namafile);
184.           CoverGambar = ImageIO.read(image1);
185.           SteganoGambar = ImageIO.read(image2);
186.           width = CoverGambar.getWidth();
187.           height = CoverGambar.getHeight();
188.           for(int i=0;i<height;i++)
189.           {
190.               for(int j=0;j<width;j++)
191.               {
192.                                                   Color   c   =   new
  Color(CoverGambar.getRGB(j,i));
193.                                                   Color   s   =   new
  Color(SteganoGambar.getRGB(j,i));
194.                   double redc = (double)(c.getRed());
195.                   double reds = (double)(s.getRed());
196.                   double greenc = (double)(c.getGreen());
197.                   double greens = (double)(s.getGreen());
198.                   double bluec = (double)(c.getBlue());
199.                   double blues = (double)(s.getBlue());
200.                   if(redc>max_rc)
201.                   {
202.                     max_rc=redc;
```

```
                    }
                    if(greenc>max_gc)
                    {
                      max_gc=greenc;
                    }
                    if(bluec>max_bc)
                    {
                      max_bc=bluec;
                    }
                    if(reds>max_rs)
                    {
                      max_rs=reds;
                    }
                    if(greens>max_gs)
                    {
                      max_rs=reds;
                    }
                    if(blues>max_bs)
                    {
                      max_rs=reds;
                    }
                    mse_r=mse_r+Math.pow((reds-redc),2);
                    mse_g=mse_g+Math.pow((greens-greenc),2);
                    mse_b=mse_b+Math.pow((blues-bluec),2);
                }
            }
            mse_r = mse_r/(width*height);
            mse_g = mse_g/(width*height);
            mse_b = mse_b/(width*height);
            mse = (mse_r+mse_g+mse_b)/3;
                                        psnr_r    =    10.0    *
  logbase10(Math.pow(Math.max(max_rc,max_rs), 2) / mse_r);
                                        psnr_g    =    10.0    *
  logbase10(Math.pow(Math.max(max_gc,max_gs), 2) / mse_g);
                                        psnr_b    =    10.0    *
  logbase10(Math.pow(Math.max(max_bc,max_bs), 2) / mse_b);
            psnr = (psnr_r+psnr_g+psnr_b)/3;
                    JOptionPane.showMessageDialog(this,"Berhasil
  Menyisipkan  \nMSE       :  "+dfmse.format(mse)+"\nPSNR       :
  "+dfpsnr.format(psnr),"Success",JOptionPane.INFORMATION_MESSAGE
  );
    }
    catch(IOException e)
    {}
  }

  private double logbase10(double x)
  {
      return Math.log(x) / Math.log(10);
  }
```

Code to extract the embedded binary of image

```
247.  private String Extract(String si)
248.  {
249.      BufferedImage SteganoGambar;
250.      int width;
251.      int height;
252.      String biner_extract="";
253.      try
254.      {
255.          File image = new File(si);
256.          SteganoGambar = ImageIO.read(image);
257.          width = SteganoGambar.getWidth();
258.          height = SteganoGambar.getHeight();
259.          String red,green,blue;
260.          for(int i=0;i<height;i++)
261.          {
262.              for(int j=0;j<width;j++)
263.              {
264.                                          Color   c   =   new
  Color(SteganoGambar.getRGB(j,i));
265.                  red = Integer.toBinaryString(c.getRed());
266.                                                  green   =
  Integer.toBinaryString(c.getGreen());
267.                  blue = Integer.toBinaryString(c.getBlue());
268.                              biner_extract  =  biner_extract  +
  red.substring(red.length()-1,red.length());
269.                              biner_extract  =  biner_extract  +
  green.substring(green.length()-1,green.length());
270.                              biner_extract  =  biner_extract  +
  blue.substring(blue.length()-1,blue.length());
271.              }
272.          }
273.      }
274.      catch(IOException e)
275.      {
276.      }
277.      return biner_extract;
278.  }
```

Code to decrypt

```
279.  private  void  Decrypt(int[]  hasilXOR,  String  pesan,  int[]
  bit_lcg)
280.  {
281.      String hasil="";
282.      String [] tampung=new String[(hasilXOR.length/4)/8];
283.      for(int i=0;i<hasilXOR.length;i++)
284.      {
285.          if((i+1)%4==0)
286.          {
287.              hasil=hasil+Integer.toString(hasilXOR[i]);
288.          }
289.      }
```

```
290.      int k = 0;
291.      for(int i=0;i<hasil.length();i++)
292.      {
293.          if((i+1)%8==0)
294.          {
295.              tampung[k]=hasil.substring(i-7,i+1);
296.              k++;
297.          }
298.      }
299.      hasil="";
300.      String endoftext="*888#";
301.      int ascii;
302.      char kar;
303.      int cek =0;
304.      for(int i=0;i<tampung.length;i++)
305.      {
306.          ascii = Integer.parseInt(tampung[i],2);
307.          kar = (char) ascii;
308.          hasil = hasil+kar;
309.          if(hasil.contains(endoftext)==true)
310.          {
311.              cek=1;
312.              hasil = hasil.substring(0,hasil.length()-5);
313.                  JOptionPane.showMessageDialog(this,"Hasil
  Ekstraksi          :          "+hasil,"Ekstraksi          Pesan
  Berhasil",JOptionPane.PLAIN_MESSAGE);
314.              break;
315.          }
316.      }
317.      if(cek==0)
318.      {
319.          System.out.println(hasil.substring(0,50));
320.                  JOptionPane.showMessageDialog(this,"Hasil
  Ekstraksi     :     "+hasil.substring(0,50),"Ekstraksi     Pesan
  Berhasil",JOptionPane.ERROR_MESSAGE);
321.      }
322.  }
```

UNICHECK

Submission author:
15k10060 ANDY WIJAYA

Check date:
17.01.2020 11:30:54 GMT+0

Report date:
17.01.2020 23:29:54 GMT+0

Check ID:
15995180

Check type:
Doc vs Internet + Library

User ID:
29127

Unika SOEGIJAPRANATA
Talenta pro patria et humanitate

File name:15.K1.0060_Andy Wijaya.doc

File ID:20297363 Page count:9 Word count:3973 Character count:22665 File size:85.00 KB

2.16% Matches

Highest match: 0.7% with library source. File ID: 13401556

1.41% Internet Matches	8	Page 11
0.76% Library matches	38	Page 11

1.06% Quotes

Quotes	1	Page 12

No references found

0% Exclusions

No exclusions found

Replacement

No replaced characters found