```java
//Robert Swartz
//Optimals Counter
//Copyright 2020
//Particle Physics


import java.awt.*;
import java.awt.event.*;


public class OptimalsCounter
{
   public static void main(String[] args)
   {
      AppletImage cubs = new AppletImage();
      int w = 600, h = 500, h2 = 75;
      cubs.powArray[0] = 1;
      for (int i = 1; i < 63; i++)
         cubs.powArray[i] = cubs.powArray[i-1]*2+1;
      cubs.pF.setLocation(125, 125);
      cubs.pF.setVisible(true);
      cubs.pF.setSize(w, h);
      cubs.pF.setResizable(false);
      cubs.mwa = new MyWindowAdapter(cubs);
      cubs.pF.addWindowListener(cubs.mwa);
      cubs.p0.setSize(w, h);
      cubs.pF.add(cubs.p0);
      cubs.start = new Button("Start");
      cubs.stop = new Button("Stop");
      cubs.mka1 = new MyKeyAdapter1(cubs);
      cubs.mma1 = new MyMouseAdapter1(cubs);
      cubs.mka2 = new MyKeyAdapter2(cubs);
      cubs.mma2 = new MyMouseAdapter2(cubs);
      cubs.start.setBackground(Color.green);
      cubs.start.setForeground(new Color(0, 0, 208));
      cubs.start.setFont(new Font("SansSerif", Font.BOLD, 13));
      cubs.start.addKeyListener(cubs.mka1);
      cubs.start.addMouseListener(cubs.mma1);
      cubs.stop.setBackground(Color.red);
      cubs.stop.setForeground(new Color(0, 0, 208));
      cubs.stop.setFont(new Font("SansSerif", Font.BOLD, 13));
      cubs.stop.addKeyListener(cubs.mka2);
      cubs.stop.addMouseListener(cubs.mma2);
      cubs.p1.setSize(w, h2);
      cubs.p1.setBackground(Color.lightGray);
      cubs.p1.add(new Label());
      cubs.p1.add(new Label());
      String[] instructions = {"Min # of Discs", "Max # of Discs", "# of Pegs"},
         defaults = {"1", "2000", "4"};
      for (int i = 0; i < 3; i++)
      {
         cubs.lbls[i] = new Label(instructions[i], Label.RIGHT);
         cubs.lbls[i].setFont(new Font("SansSerif", Font.PLAIN, 13));
         cubs.entries[i] = new TextField(defaults[i]);
         cubs.entries[i].setFont(new Font("SansSerif", Font.PLAIN, 13));
         cubs.p1.add(cubs.lbls[i]);
         cubs.p1.add(cubs.entries[i]);
         for (int j = 0; j < 4; j++)
            cubs.p1.add(new Label());
      }
```

```
      cubs.p1.add(cubs.start);
      cubs.p1.add(cubs.stop);
      cubs.lbls[3] = new Label("", Label.CENTER);
      cubs.lbls[3].setForeground(new Color(0, 0, 208));
      cubs.p1.add(cubs.lbls[3]);
      cubs.p1.add(new Label());
      cubs.out.setEditable(false);
      cubs.out.setSize(w, h-h2);
      cubs.out.setFont(new Font("Times", Font.PLAIN, 15));
      cubs.out.setBackground(new Color(112, 224, 192));
      cubs.p0.add(cubs.out, "Center");
      cubs.p0.add(cubs.p1, "South");
      cubs.pF.paintAll(cubs.pF.getGraphics());
      cubs.entries[0].requestFocus();
      System.gc();
   }
}


class AppletImage
{
   public OptimalsCounter2 cubs2;
   public Frame pF = new Frame("Optimals Counter");
   public Panel p0 = new Panel(new BorderLayout()), p1 = new Panel(new
      GridLayout(4, 6));
   public TextArea out = new TextArea("Robert Swartz\nwww.mathapplets.net\n"
      +"Optimals Counter\nCopyright 2020", 10, 10, TextArea.
      SCROLLBARS_VERTICAL_ONLY);
   public Button start, stop;
   public Label[] lbls = new Label[4];
   public TextField[] entries = new TextField[3];
   public MyKeyAdapter1 mka1;
   public MyMouseAdapter1 mma1;
   public MyKeyAdapter2 mka2;
   public MyMouseAdapter2 mma2;
   public MyWindowAdapter mwa;
   public boolean running;
   public long[] powArray = new long[63];
   public boolean[][] doneYet = new boolean[2000][1000];
   public long[][] moves = new long[2000][1000], opts = new long[2000][1000];
}


class OptimalsCounter2 extends Thread
{
   public AppletImage cubs;
   public int minDiscs, maxDiscs, pegs;
   public MyException myE = new MyException();

   public OptimalsCounter2(AppletImage cubs0, int min, int max, int pegs0)
   {
      cubs = cubs0;
      minDiscs = Math.max(min, pegs0);
      maxDiscs = max;
      pegs = pegs0;
   }

   public void run()
   {
      cubs.running = true;
```

```
121         cubs.lbls[3].setFont(new Font("SansSerif", Font.BOLD+Font.ITALIC, 13));
122         cubs.lbls[3].setText("Busy...");
123         String table = "\t\tNumber of Optimals for the "+pegs+
124            " Peg Problem\n\n\t\tDiscs\t\t\t\t   Optimals";
125         cubs.out.setText(table);
126         try
127         {
128            optimize(maxDiscs, pegs);
129         }
130         catch (MyException e){}
131         System.gc();
132         for (int discs = minDiscs; discs <= maxDiscs && cubs.doneYet[discs-1][pegs
133            -1]; discs++)
134            table+="\n\t\t"+discs+"\t\t\t\t   "+cubs.opts[discs-1][pegs-1];
135         cubs.out.setText(table);
136         cubs.out.requestFocus();
137         if (cubs.running)
138         {
139            cubs.lbls[3].setFont(new Font("SansSerif", Font.BOLD, 13));
140            cubs.lbls[3].setText("DONE");
141         }
142         cubs.running = false;
143      }
144
145      public void optimize(int discs2, int pegs2)throws MyException
146      {
147         int n, p, k;
148         long count1, count2;
149         $towers: for (n = 1; n <= discs2; n++)
150            for (p = 4; p <= pegs2; p++)
151               if (!cubs.doneYet[n-1][p-1])
152               {
153                  count1 = Long.MAX_VALUE;
154                  k = 0;
155                  if (p == 4 && n > 61)
156                     k = n-61;
157                  for (; k < n; k++)
158                  {
159                     if (!cubs.running)
160                        throw myE;
161                     count2 = 0;
162                     if (k > 0)
163                     {
164                        if (k >= p)
165                           count2 = 2*cubs.moves[k-1][p-1];
166                        else
167                           count2 = 4*k-2;
168                     }
169                     if (p > 4)
170                     {
171                        if (n-k > p-2)
172                           count2+=cubs.moves[n-k-1][p-2];
173                        else
174                           count2+=(n-k)*2-1;
175                     }
176                     else
177                        count2+=cubs.powArray[n-k-1];
178                     if (count2 == count1)
179                        cubs.opts[n-1][p-1]++;
180                     else if (count2 < count1)
```

```
                {
                   cubs.opts[n-1][p-1] = 1;
                   count1 = count2;
                }
             }
             if (count1 < 1)
                break $towers;
             cubs.moves[n-1][p-1] = count1;
             cubs.doneYet[n-1][p-1] = true;
          }
   }
}


class MyException extends Exception{}


class MyKeyAdapter1 extends KeyAdapter
{
   public AppletImage cubs;

   public MyKeyAdapter1(AppletImage cubs0)
   {
      cubs = cubs0;
   }

   public void keyReleased(KeyEvent e)
   {
      int minDiscs = 0, maxDiscs = 0, pegs = 0;
      boolean error = false;
      if ((e.getKeyCode() == KeyEvent.VK_ENTER || e.getKeyCode() ==
         KeyEvent.VK_SPACE) && !cubs.running)
      {
         try
         {
            minDiscs = Integer.parseInt(cubs.entries[0].getText());
            maxDiscs = Integer.parseInt(cubs.entries[1].getText());
            pegs = Integer.parseInt(cubs.entries[2].getText());
         }
         catch (NumberFormatException e2)
         {
            error = true;
         }
         if (error || minDiscs < 1 || maxDiscs < 1 || maxDiscs < minDiscs ||
            pegs < 4 || minDiscs > 2000 || maxDiscs > 2000 || pegs > 1000)
         {
            cubs.lbls[3].setFont(new Font("Times", Font.BOLD, 12));
            cubs.lbls[3].setText("INVALID ENTRY!!!");
         }
         else
         {
            cubs.cubs2 = new OptimalsCounter2(cubs, minDiscs, maxDiscs, pegs);
            cubs.cubs2.start();
         }
      }
   }
}


class MyMouseAdapter1 extends MouseAdapter
```

```
{
   public AppletImage cubs;

   public MyMouseAdapter1(AppletImage cubs0)
   {
      cubs = cubs0;
   }

   public void mousePressed(MouseEvent e)
   {
      int minDiscs = 0, maxDiscs = 0, pegs = 0;
      boolean error = false;
      if (e.getButton() == MouseEvent.BUTTON1 && !cubs.running)
      {
         try
         {
            minDiscs = Integer.parseInt(cubs.entries[0].getText());
            maxDiscs = Integer.parseInt(cubs.entries[1].getText());
            pegs = Integer.parseInt(cubs.entries[2].getText());
         }
         catch (NumberFormatException e2)
         {
            error = true;
         }
         if (error || minDiscs < 1 || maxDiscs < 1 || maxDiscs < minDiscs ||
            pegs < 4 || minDiscs > 2000 || maxDiscs > 2000 || pegs > 1000)
         {
            cubs.lbls[3].setFont(new Font("Times", Font.BOLD, 12));
            cubs.lbls[3].setText("INVALID ENTRY!!!");
         }
         else
         {
            cubs.cubs2 = new OptimalsCounter2(cubs, minDiscs, maxDiscs, pegs);
            cubs.cubs2.start();
         }
      }
   }
}


class MyKeyAdapter2 extends KeyAdapter
{
   public AppletImage cubs;

   public MyKeyAdapter2(AppletImage cubs0)
   {
      cubs = cubs0;
   }

   public void keyPressed(KeyEvent e)
   {
      if (e.getKeyCode() == KeyEvent.VK_ENTER || e.getKeyCode() ==
         KeyEvent.VK_SPACE)
      {
         cubs.running = false;
         cubs.lbls[3].setText("");
      }
   }
}

```

```
301
302 class MyMouseAdapter2 extends MouseAdapter
303 {
304    public AppletImage cubs;
305
306    public MyMouseAdapter2(AppletImage cubs0)
307    {
308       cubs = cubs0;
309    }
310
311    public void mousePressed(MouseEvent e)
312    {
313       if (e.getButton() == MouseEvent.BUTTON1)
314       {
315          cubs.running = false;
316          cubs.lbls[3].setText("");
317       }
318    }
319 }
320
321
322 class MyWindowAdapter extends WindowAdapter
323 {
324    public AppletImage cubs;
325
326    public MyWindowAdapter(AppletImage cubs0)
327    {
328       cubs = cubs0;
329    }
330
331    public void windowClosing(WindowEvent e)
332    {
333       cubs.running = false;
334       cubs.pF.dispose();
335    }
336 }
```