MODULE 7 - SHEET 1

```
// This program is due to Dr A.C. Norman

import java.applet.Applet;
import java.awt.Graphics;

public class Turtle extends Applet
 { private static final double SIZE = 5d;       // Try changing
   private static final double INC = 11d;       // these three
   private static final int N = 5000;           // values.

   public void paint(Graphics g)
    { double x = 200d, y = 300d, th1 = 0d, th2 = 0d, th3 = 0d;

      for (int i=0; i<N; i++)
       { th3 = th3 + INC;
         th2 = th2 + th3;
         th1 = th1 + th2;
         double x1 = x + SIZE*Math.cos(Math.PI*th1/180d);
         double y1 = y + SIZE*Math.sin(Math.PI*th1/180d);

         g.drawLine((int)x, (int)y, (int)x1, (int)y1);

         x = x1;
         y = y1;
       }
    }
 }

// It would be safer to normalise each angle after each increment
// by including statements such as:
//
//       if (th3 >= 180d)
//          th3 = th3 - 360d;
```

Key this source into the file Turtle.html

```
<HTML>
  <BODY>
    <APPLET code="Turtle.class" width=400 height=400>
            Java is not available.
    </APPLET>
  </BODY>
</HTML>
```

Give the following command:

```
$ appletviewer Turtle.html &
```

The result is a sequence of straight-line segments which form a remarkably symmetrical figure given that there is no apparent attempt in the program to produce a result which has such symmetry,

```
public class EightQueens
 { private static int count=0;

   public static void main(String[] args)
    { tryIt(0,0,0);
      System.out.printf("There are %d solutions%n", count);
    }

   private static void tryIt(int left, int above, int right)
    { if (above==0xFF)
         count++;
      else
       { int poss = ~(left | above | right) & 0xFF;
         while (poss != 0)
          { int place = poss & (-poss);
            tryIt((left|place)<<1, above|place, (right|place)>>1);
            poss = poss & (~place);
          }
       }
      return;
    }
 }
```

```
import java.util.Calendar;
import java.util.GregorianCalendar;

public class Friday13
 { public static void main(String[] args)
    { int[] days = {0,0,0,0,0,0,0,0};

      GregorianCalendar cal = new GregorianCalendar(1900,0,13);  // Start on 13.1.1900

      for (int m=0; m<4800; m++)                      // For 4800 months:
       { days[cal.get(Calendar.DAY_OF_WEEK)]++;       //    Note day of week of 13th
         cal.add(Calendar.MONTH, 1);                  //    Advance to next month
       }

      System.out.printf("Sunday    %d%n", days[1]);      // Print out the day counts
      System.out.printf("Monday    %d%n", days[2]);      // of the thirteenths.
      System.out.printf("Tuesday   %d%n", days[3]);
      System.out.printf("Wednesday %d%n", days[4]);
      System.out.printf("Thursday  %d%n", days[5]);
      System.out.printf("Friday    %d%n", days[6]);
      System.out.printf("Saturday  %d%n", days[7]);

      cal.add(Calendar.DAY_OF_MONTH, -7*20871);       // Go back 20871 weeks...
```

```
     System.out.printf(cal.get(Calendar.DAY_OF_MONTH) + "." +      // which takes
                       (cal.get(Calendar.MONTH)+1) + "." +         // us back to
                        cal.get(Calendar.YEAR) + "%n");            // 13.1.1900

   }
}
```