```
 1 import java.awt.Color;
 9
10 public class PieChart
11 {
12     public PieChart(HashMap <String,Integer> frequencyEvents, int frequencyAll, int
   canvasWidth, int canvasHeight, int nMostFrequent)
13     {
14         this.frequencyEvents = frequencyEvents;
15         this.frequencyAll = frequencyAll;
16         this.canvasWidth = canvasWidth;
17         this.canvasHeight = canvasHeight;
18         this.nMostFrequent = nMostFrequent;
19     }
20
21     public double probability(String s)
22     {
23         if(frequencyEvents.get(s) == null)
24             return 0;
25
26         return (double)frequencyEvents.get(s) / (double)frequencyAll;
27     }
28
29     public String getMostFrequent()
30     {
31         /*String maxKey = null;
32         int maxVal = 0;
33
34         Iterator it = frequencyEvents.entrySet().iterator();
35         while (it.hasNext()) {
36             if(frequencyEvents.get(maxKey) != null)
37                 maxVal = frequencyEvents.get(maxKey);
38             Map.Entry pair = (Map.Entry)it.next();
39             if(pair.getValue() != null && maxVal < (int)pair.getValue())
40             {
41                 maxKey = (String)pair.getKey();
42                 maxVal = (int)pair.getValue();
43             }
44              it.remove(); // avoids a ConcurrentModificationException*/
45
46         Map.Entry<String, Integer> maxEntry = null;
47
48         for (Map.Entry<String, Integer> entry : frequencyEvents.entrySet())
49         {
50             if (maxEntry == null || entry.getValue().compareTo(maxEntry.getValue()) >= 0)
51             {
52                 maxEntry = entry;
53             }
54         }
55
56
57         //System.out.println(maxEntry.getValue());
58
59
60         return maxEntry.getKey();
61     }
62
63     public void removeMostFrequent()
```

```
64      {
65          frequencyEvents.remove(getMostFrequent());
66      }
67
68      public void draw(Graphics g)
69      {
70          int currentAngle = 0;
71
72          g.setColor(Color.BLUE);
73
74
75
76          for(int i = 0; i < nMostFrequent; i++)
77          {
78              if(i%9 == 0)
79                  g.setColor(Color.blue);
80              if(i%9 == 1)
81                  g.setColor(Color.black);
82              if(i%9 == 2)
83                  g.setColor(Color.cyan);
84              if(i%9 == 3)
85                  g.setColor(Color.red);
86              if(i%9 == 4)
87                  g.setColor(Color.green);
88              if(i%9 == 5)
89                  g.setColor(Color.orange);
90              if(i%9 == 6)
91                  g.setColor(Color.yellow);
92              if(i%9 == 7)
93                  g.setColor(Color.pink);
94              if(i%9 == 8)
95                  g.setColor(Color.magenta);
96
97              g.fillArc(canvasWidth/4,canvasHeight/4,canvasWidth/2,canvasHeight/2,currentAngle,(
    int)(probability(getMostFrequent())*360));
98              currentAngle+=(int)(probability(getMostFrequent())*360);
99              int fontSize = 20;
100
101             g.setFont(new Font("TimesRoman", Font.PLAIN, fontSize));
102             g.setColor(Color.blue);
103             //System.out.println(getMostFrequent());
104             //System.out.println(currentAngle);
105             g.drawString(getMostFrequent() + ": " + (int)(probability(getMostFrequent())*360)
    + "%", canvasWidth/2 + (int)(Math.cos(Math.toRadians(currentAngle))*canvasWidth/3),
    canvasHeight/2 + (int)(-Math.sin(Math.toRadians(currentAngle))*canvasHeight/3));
106             removeMostFrequent();
107         }
108
109         g.setColor(Color.gray);
110         g.fillArc(canvasWidth/4, canvasHeight/4, canvasWidth/2, canvasHeight/2, currentAngle,
    360-currentAngle);
111         g.setColor(Color.blue);
112         g.drawString("Leftover: " + ((int)(((double)360-currentAngle)/360*100)) + "%",
    canvasWidth/2 + canvasWidth/3, canvasHeight/2);
113     }
114
115     protected HashMap <String,Integer> frequencyEvents;
```

```
116     protected int frequencyAll;
117     protected int nMostFrequent;
118     protected int canvasWidth;
119     protected int canvasHeight;
120 }
121
```