```java

package vecDeffuant;

import java.awt.Color;

/**
    This class represents the colour  of a site/agent in the model.
    The colour is determined by the feature values. The aim is
    that <CODE>getColour()</CODE> returns unique colours for
    different combinations of features.
*/
public class AgentColour {

    private int featureCount;   // Number of cultural features.
    private int[] traitCounts;  // Constant vector [2,...,2],  since each feature has exactly two possible configurations in the vec  Deff model
    private float rmax=0, gmax=0, bmax=0;
    private int ri, gi, bi;

    /**
        Create the colouring object.

        @param featureCount number of cultural features possessed by agents.
        @throws IllegalArgumentException if invalid number of features.
    */
    public AgentColour( int featureCount, int traitCount ) {
        this.featureCount = featureCount;
        this.traitCounts = new int[featureCount];
        for( int i = 0; i < featureCount; i++ ) {
            this.traitCounts[i] = traitCount;
        }

        if(this.featureCount > 3) {
            int k = Math.round(featureCount/3.0f);
            ri = k; gi = ri+k; bi = featureCount;
            rmax=0; gmax=0; bmax=0;
            int i;
            for( i = 0; i < ri; i++ ) rmax += traitCounts[i]-1;
            for( i = ri; i < gi; i++ ) gmax += traitCounts[i]-1;
            for( i = gi; i < bi; i++ ) bmax += traitCounts[i]-1;
        } else if(this.featureCount == 3) {
            rmax = traitCounts[0]-1;
            gmax = traitCounts[1]-1;
            bmax = traitCounts[2]-1;
        } else if(this.featureCount == 2) {
            rmax = traitCounts[0]-1;
            gmax = traitCounts[1]-1;
        } else if(this.featureCount == 1) {
            rmax = traitCounts[0]-1;
        } else
            throw new IllegalArgumentException( "Invalid number of features." );
    }

    /**
        Return a colour that represents the configuration.
        @return the colour.
    */
    public Color getColour( int[] traits) {
        if(featureCount > 3) {
            float r=0,g=0,b=0;  // range from 0 to 1.
            int i;
            for( i = 0; i < ri; i++ ) r += traits[i];
```

```
62              for( i = ri; i < gi; i++ ) g += traits[i];
63              for( i = gi; i < bi; i++ ) b += traits[i];
64              return(new Color(r/rmax, g/gmax, b/bmax));
65          } else if(featureCount == 3)
66              return(new Color(traits[0]/rmax, traits[1]/gmax, traits[2]/bmax));
67          else if(featureCount == 2) {
68              float c1 = traits[0]/rmax;
69              float c2 = traits[1]/gmax;
70              return(new Color(c1, c2, 0.5f*(c1+c2)));
71          } else {
72              float c = traits[0]/rmax;
73              return(new Color(c, c, c));
74          }
75      }
76  }
77
78
79
```