```
package uebung4;
import java.util.*;
public class Banner {

    private List<Pennant> pennants;
    private int length;
    private Quality quality;

    public Banner(int length, List<Pennant> pennants) {
        this.length = length;
        this.pennants = pennants;
    }
    //randomly create a chain of Pennants between 1 and 10
    public Banner(int length) {
        this.length = length;
        this.pennants = new ArrayList<Pennant>();
    }
    public Quality getQuality() {
        if (this.quality == null) {
            this.assessQuality();
        }
        return this.quality;
    }
    /**
      * Assesses the quality of a Banner
      */
    public void assessQuality() {
        Quality bestQuality = new Quality(9999,0);
        List<Pennant> pennantList = this.getPennants();
        System.out.println("Pennant order: "+pennantList.toString());
        int startingIndex = 1;
        for (Pennant pennant : pennantList) {
            char colorSelect = pennant.getColor();
            int distanceBetweenLikeColors = 0;
            for (int i = startingIndex; i < pennantList.size(); i++) {
                if(
                  (pennantList.get(i).getColor() == (pennant.getColor()))
                   &&
                  (bestQuality.getDistance() >= distanceBetweenLikeColors)
                  )
                {
                    if (distanceBetweenLikeColors < bestQuality.getDistance()) {
                      bestQuality.setDistance(distanceBetweenLikeColors);
                      bestQuality.setOccurance(1);
                    } else if (distanceBetweenLikeColors == bestQuality.getDistance()){
                      bestQuality.setDistance(distanceBetweenLikeColors);
                      bestQuality.setOccurance(bestQuality.getOccurance()+1);
                    }
                    // System.out.println("Pennant: "+pennant.toString()+" - distance: "+distanceBetweenLikeColors);
                    distanceBetweenLikeColors = 0;
                    break;
                } else if ((pennantList.get(i).getColor() != (pennant.getColor()))) {
                    distanceBetweenLikeColors++;
                    continue;
                } else {
                    distanceBetweenLikeColors++;
                    continue;
                }
            }
            startingIndex++;
        }
        this.quality = bestQuality;
    }
    public Banner shift() {
      Banner bShifted = new Banner(this.getLength(), this.getPennants());
      List<Pennant> pShift = bShifted.getPennants();
      for (int i = 0; i < pShift.size(); i++) {
        for (int j = 0; j < pShift.size(); j++) {
            Collections.swap(pShift, i, j);
            bShifted.setPennants(pShift);
            if (bShifted.getQuality().compareTo(this.getQuality()) > 0) {
```

```
            this.setPennants(pShift);
            this.assessQuality();
            System.out.println("Shift found better solution! - "+this.toString()+" New Quality: "+this.getQuality());
            Collections.swap(pShift, j, i);
          } else {
            Collections.swap(pShift, j, i);
          }
      }
    }
    return this;
  }
  public int getLength() {
      return this.length;
  }
  public List<Pennant> getPennants() {
      return this.pennants;
  }
  public void setPennants(List<Pennant> pennants) {
      this.pennants = pennants;
  }
  public void addPennant(Pennant pennant) {
    pennants.add(pennant);
  }
  public String toString() {
    return pennants.toString();
  }
```

}