

```

1 package tpe.collections.minions;
2
3 import tpe.collections.minions.domain.Minion;
4 import tpe.collections.minions.domain.MinionFarbe;
5 import tpe.collections.minions.domain.MinionIterable;
6
7 import java.util.HashSet;
8 import java.util.Set;
9
10 public class Main {
11     /**
12      * Gibt einen String zur Ausgabe aller errechneten Werte
13      * zurück.
14      *
15      * @param gesamtOD      Minions ohne Dubletten
16      * @param gelbOD         Gelbe Minions ohne Dubletten
17      * @param gelbeODP       Gelbe Minions ohne Dubletten in
18      *                        Prozent
19      * @param violettOD      Violette Minions ohne Dubletten
20      * @param violettODP     Violette Minions ohne Dubletten in
21      *                        Prozent
22      * @param gelbviolettODV Verhältnis Gelbe und Violette
23      *                        Minions
24      * @param gesamtMD       Minions mit Dubletten
25      * @param gelbMD         Gelbe Minions mit Dubletten
26      * @param gelbeMDP       Gelbe Minions mit Dubletten in
27      *                        Prozent
28      * @param violettMD      Violette Minions mit Dubletten
29      * @param violettMDP     Violette Minions mit Dubletten in
30      *                        Prozent
31      * @param gelbviolettMDV Verhältnis Gelbe und Violette
32      *                        Minions
33      * @param fehler         Fehler bei der Zählung
34      *
35      * @return Formatierter String mit allen wichtigen Infos
36      */
37     public static String outputString(int gesamtOD, int gelbOD,
38         double gelbeODP,
39         int violettOD, double
40         violettODP, double gelbviolettODV, int gesamtMD,
41         int gelbMD, double gelbeMDP
42         , int violettMD, double violettMDP,
43         double gelbviolettMDV,
44         double fehler) {
45         /**
46          * Ohne Dubletten: 1128
47          * Minions 717 gelbe (64%) und 411 violett (36%),
48          * gelb/violett 1,74
49          * -----
50          * Mit Dubletten: 1184 Minions 758 gelbe (64%) und 426

```

```

39 violett (36%),
40     * gelb/violett 1,78
41     * -----
42     * Fehler bei der Zählung: 4,7%
43     */
44     return ("Ohne Dubletten: " + gesamtOD + " Minions\n" +
    gelbOD
45         + " gelbe (" + gelbeODP + "%) und " + violettOD +
    " violett ("
46         + violettODP + "%), gelb/violett " +
    gelbviolettODV + "\n"
47         +
    "-----\n"
48         + "Mit Dubletten: " + gesamtMD + " Minions\n" +
    gelbMD
49         + " gelbe (" + gelbeMDP + "%) und " + violettMD +
    " violett ("
50         + violettMDP + "%), gelb/violett " +
    gelbviolettMDV + "\n"
51         +
    "-----\n"
52         + "Fehler bei der Zählung: " + fehler + "\n");
53 }
54
55 /**
56  * ermittelt Gesamtanzahl Minions
57  *
58  * @param factory
59  *
60  * @return integer die Anzahl
61  */
62 public static int getAnzahlGesamt(MinionIterable factory) {
63     int anzahl = 0;
64     for (Minion m : factory) {
65         anzahl++;
66     }
67     return anzahl;
68 }
69
70 /**
71  * Zähle gelbe Minions mit Dupletten
72  *
73  * @param factory
74  *
75  * @return integer die Anzahl
76  */
77 public static int getGelbeMinionsMD(MinionIterable factory) {
78     int anzahl = 0;
79     for (Minion n : factory) {
80         if (n.getFarbe() == (MinionFarbe.GELB)) {

```

```

81         anzahl++;
82     }
83 }
84     return anzahl;
85 }
86
87 /**
88  * Zähle violette Minions mit Dupletten
89  *
90  * @param factory
91  *
92  * @return integer die Anzahl
93  */
94     public static int getVioletteMinionsMD(MinionIterable
factory) {
95         int anzahl = 0;
96         for (Minion n : factory) {
97             if (n.getFarbe() == (MinionFarbe.VIOLETT)) {
98                 anzahl++;
99             }
100         }
101         return anzahl;
102     }
103
104 /**
105  * Zähle alle Minions ohne Dupletten
106  *
107  * @param factory
108  *
109  * @return integer die Anzahl
110  */
111     public static int getMinionsOD(MinionIterable factory) {
112
113         Set<Minion> set = new HashSet<Minion>();
114
115         for (Minion n : factory) {
116             set.add(n);
117         }
118
119         return set.size();
120     }
121
122 /**
123  * Zähle alle gelbe Minions ohne Dupletten
124  *
125  * @param factory
126  *
127  * @return
128  */
129     public static int getMinionsODGelb(MinionIterable factory) {

```

```

130         Set<Minion> set = new HashSet<Minion>();
131
132         for (Minion n : factory) {
133             if (n.getFarbe() == MinionFarbe.GELB)
134                 set.add(n);
135         }
136
137         return set.size();
138     }
139
140     /**
141      * Zähle alle violetten Minions ohne Dupletten
142      *
143      * @param factory
144      *
145      * @return
146      */
147     public static int getMinionsODViolett(MinionIterable factory
148 ) {
149         Set<Minion> set = new HashSet<Minion>();
150
151         for (Minion n : factory) {
152             if (n.getFarbe() == MinionFarbe.VIOLETT)
153                 set.add(n);
154         }
155
156         return set.size();
157     }
158
159     /**
160      * Berechnet Prozentwert der Eingabe *
161      *
162      * @param gesamt 100 %
163      * @param wert    wieviel prozent sind das?
164      *
165      * @return integer den Prozentsatz
166      */
167     public static double berechneProzent(double gesamt, double
168 wert) {
169         return Math.round((100.0 * wert) / gesamt);
170     }
171
172     /**
173      * Berechnet Differenz zweier Werte in Prozent
174      *
175      * @param gesamt    alle zusammen
176      * @param ohneDupletten keine Dupletten enthalten
177      *
178      * @return integer die Anzahl ohne Dupletten
179      */

```

```

178     public static double berechneFehlerQuote(double gesamt,
179         double ohneDupletten) {
180         double tmp = gesamt - ohneDupletten;
181
182         return Math.round((100.0 * tmp) / gesamt);
183     }
184
185     /**
186      * Ermittelt das Verhältnis von a zu b
187      *
188      * @param a erster Parameter
189      * @param b zweiter Parameter
190      *
191      * @return double das Verhältnis von a zu b
192      */
193     public static double verhaeltniss(double a, double b) {
194
195         return (double) Math.round(a / b * 100) / 100;
196     }
197
198     public static void main(String[] args) {
199
200         MinionIterable factory = new MinionIterable();
201
202         // OHNE DUBLETTEN
203         int gesamtOD = getMinionsOD(factory);
204         int gelbOD = getMinionsODGelb(factory);
205         double gelbODP = berechneProzent(gesamtOD, gelbOD);
206         int violettOD = getMinionsODViolett(factory);
207         double violettODP = berechneProzent(gesamtOD, violettOD)
208     ;
209
210         double gelbviolettODV = verhaeltniss(gelbOD, violettOD);
211
212         // MIT DUBLETTEN
213         int gesamtMD = getAnzahlGesamt(factory);
214         int gelbMD = getGelbeMinionsMD(factory);
215         double gelbMDP = berechneProzent(gesamtMD, gelbMD);
216         int violettMD = getVioletteMinionsMD(factory);
217         double violettMDP = berechneProzent(gesamtMD, violettMD)
218     ;
219
220         double gelbviolettMDV = verhaeltniss(gelbMD, violettMD);
221
222         //FEHLER
223         double fehler = berechneFehlerQuote(gesamtMD, gesamtOD);
224
225         System.out.println(outputString(
226             gesamtOD, gelbOD, gelbODP, violettOD, violettODP
227             , gelbviolettODV,
228             gesamtMD, gelbMD, gelbMDP, violettMD, violettMDP

```

```
223 , gelbviolettMDV,  
224         fehler  
225     ));  
226  
227     }  
228 }  
229
```