```java
import components.naturalnumber.NaturalNumber;

/**
 * Controller class1
 *
 * @author Micah Casey-Fusco
 */
public final class NNCalcController1 implements NNCalcController {

    /**
     * Model object.
     */
    private final NNCalcModel model;

    /**
     * View object.
     */
    private final NNCalcView view;

    /**
     * Useful constants.
     */
    private static final NaturalNumber TWO = new NaturalNumber2(2),
            INT_LIMIT = new NaturalNumber2(Integer.MAX_VALUE);

    /**
     * Updates this.view to display this.model, and to allow only operations
     * that are legal given this.model.
     *
     * @param model
     *            the model
     * @param view
     *            the view
     * @ensures [view has been updated to be consistent with model]
     */
    private static void updateViewToMatchModel(NNCalcModel model,
            NNCalcView view) {

        //assign model info to top and bottom

        NaturalNumber top = model.top();
        NaturalNumber bottom = model.bottom();

        // update both displays to represent changes

        view.updateTopDisplay(top);
        view.updateBottomDisplay(bottom);

        // Boolean statements to check operators that can be used

        if (top.compareTo(bottom) >= 0) {
            view.updateSubtractAllowed(true);
        } else {
            view.updateSubtractAllowed(false);
        }

        if (!bottom.isZero()) {
```

```
59              view.updateDivideAllowed(true);
60          } else {
61              view.updateDivideAllowed(false);
62          }
63
64          if (bottom.compareTo(INT_LIMIT) <= 0) {
65              view.updatePowerAllowed(true);
66          } else {
67              view.updatePowerAllowed(false);
68          }
69
70          if (bottom.compareTo(TWO) >= 0 && bottom.compareTo(INT_LIMIT) <= 0) {
71              view.updateRootAllowed(true);
72          } else {
73              view.updateRootAllowed(false);
74          }
75
76      }
77
78      /**
79       * Constructor.
80       *
81       * @param model
82       *            model to connect to
83       * @param view
84       *            view to connect to
85       */
86      public NNCalcController1(NNCalcModel model, NNCalcView view) {
87          this.model = model;
88          this.view = view;
89          updateViewToMatchModel(model, view);
90      }
91
92      @Override
93      public void processClearEvent() {
94          /*
95           * Get alias to bottom from model
96           */
97          NaturalNumber bottom = this.model.bottom();
98          /*
99           * Update model in response to this event
100          */
101         bottom.clear();
102         /*
103          * Update view to reflect changes in model
104          */
105         updateViewToMatchModel(this.model, this.view);
106     }
107
108     @Override
109     public void processSwapEvent() {
110         /*
111          * Get aliases
112          */
113         NaturalNumber top = this.model.top();
114         NaturalNumber bottom = this.model.bottom();
115         /*
```

```
116          * Update model in response to this event
117          */
118         NaturalNumber temp = top.newInstance();
119         temp.transferFrom(top);
120         top.transferFrom(bottom);
121         bottom.transferFrom(temp);
122         /*
123          * Update view
124          */
125         updateViewToMatchModel(this.model, this.view);
126     }
127
128     @Override
129     public void processEnterEvent() {
130         /*
131          * Get aliases
132          */
133         NaturalNumber top = this.model.top();
134         NaturalNumber bottom = this.model.bottom();
135         /*
136          * Update model in response to this event
137          */
138         top.copyFrom(bottom);
139
140         /*
141          * Update view
142          */
143         updateViewToMatchModel(this.model, this.view);
144
145     }
146
147     @Override
148     public void processAddEvent() {
149
150         NaturalNumber top = this.model.top();
151         NaturalNumber bottom = this.model.bottom();
152
153         /*
154          * Update model in response to this event(Addition)
155          */
156         top.add(bottom);
157         bottom.transferFrom(top);
158
159         /*
160          * Update view
161          */
162         updateViewToMatchModel(this.model, this.view);
163
164     }
165
166     @Override
167     public void processSubtractEvent() {
168
169         NaturalNumber top = this.model.top();
170         NaturalNumber bottom = this.model.bottom();
171
172         /*
```

```
173          * Update model in response to this event(top-bottom)
174          */
175         top.subtract(bottom);
176         bottom.transferFrom(top);
177 
178         /*
179          * Update view
180          */
181         updateViewToMatchModel(this.model, this.view);
182 
183     }
184 
185     @Override
186     public void processMultiplyEvent() {
187 
188         NaturalNumber top = this.model.top();
189         NaturalNumber bottom = this.model.bottom();
190 
191         /*
192          * Update model in response to this event(Multiplication)
193          */
194         top.multiply(bottom);
195         bottom.transferFrom(top);
196 
197         /*
198          * Update view
199          */
200         updateViewToMatchModel(this.model, this.view);
201     }
202 
203     @Override
204     public void processDivideEvent() {
205 
206         NaturalNumber top = this.model.top();
207         NaturalNumber bottom = this.model.bottom();
208 
209         /*
210          * Update model in response to this event(top divided by bottom)
211          */
212         top.divide(bottom);
213         bottom.transferFrom(top);
214         /*
215          * Update view
216          */
217         updateViewToMatchModel(this.model, this.view);
218 
219     }
220 
221     @Override
222     public void processPowerEvent() {
223 
224         NaturalNumber top = this.model.top();
225         NaturalNumber bottom = this.model.bottom();
226 
227         /*
228          * Update model in response to this event (top power bottom)
229          */
```

```
230         top.power(bottom.toInt());
231         bottom.transferFrom(top);
232         /*
233          * Update view
234          */
235
236         updateViewToMatchModel(this.model, this.view);
237
238     }
239
240     @Override
241     public void processRootEvent() {
242
243         NaturalNumber top = this.model.top();
244         NaturalNumber bottom = this.model.bottom();
245
246         /*
247          * Update model in response to this event (top root bottom)
248          */
249         top.root(bottom.toInt());
250         bottom.transferFrom(top);
251
252         /*
253          * Update view
254          */
255         updateViewToMatchModel(this.model, this.view);
256
257     }
258
259     @Override
260     public void processAddNewDigitEvent(int digit) {
261
262         NaturalNumber bottom = this.model.bottom();
263
264         assert digit >= 0 && digit < 10 : "Violation of: 0 <= digit < 10";
265
266         bottom.multiplyBy10(digit);
267
268         /*
269          * Update view
270          */
271         updateViewToMatchModel(this.model, this.view);
272
273     }
274
275 }
276
```