```java
import components.naturalnumber.NaturalNumber;

/**
 * Controller class.
 *
 * @author Put your name here
 */
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) {

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

```

```
43          view.updateSubtractAllowed(top.compareTo(bottom) >= 0);
44          view.updateDivideAllowed(!bottom.isZero());
45          view.updateRootAllowed(!bottom.isZero() &&
    bottom.canConvertToInt());
46          view.updatePowerAllowed(bottom.canConvertToInt());
47
48          view.updateTopDisplay(top);
49          view.updateBottomDisplay(bottom);
50      }
51
52      /**
53       * Constructor.
54       *
55       * @param model
56       *            model to connect to
57       * @param view
58       *            view to connect to
59       */
60      public NNCalcController1(NNCalcModel model, NNCalcView view) {
61          this.model = model;
62          this.view = view;
63          updateViewToMatchModel(model, view);
64      }
65
66      @Override
67      public void processClearEvent() {
68          /*
69           * Get alias to bottom from model
70           */
71          NaturalNumber bottom = this.model.bottom();
72          /*
73           * Update model in response to this event
74           */
75          bottom.clear();
76          /*
77           * Update view to reflect changes in model
78           */
79          updateViewToMatchModel(this.model, this.view);
80      }
81
82      @Override
83      public void processSwapEvent() {
84          /*
85           * Get aliases to top and bottom from model
```

```
 86          */
 87         NaturalNumber top = this.model.top();
 88         NaturalNumber bottom = this.model.bottom();
 89         /*
 90          * Update model in response to this event
 91          */
 92         NaturalNumber temp = top.newInstance();
 93         temp.transferFrom(top);
 94         top.transferFrom(bottom);
 95         bottom.transferFrom(temp);
 96         /*
 97          * Update view to reflect changes in model
 98          */
 99         updateViewToMatchModel(this.model, this.view);
100     }
101
102     @Override
103     public void processEnterEvent() {
104
105         // Copying the top from the bottom
106         this.model.top().copyFrom(this.model.bottom());
107
108         // Updating view to reflect changes in model
109         updateViewToMatchModel(this.model, this.view);
110
111     }
112
113     @Override
114     public void processAddEvent() {
115
116         // Aliasing model top and bottom
117         NaturalNumber top = this.model.top();
118         NaturalNumber bottom = this.model.bottom();
119
120         // Adding bottom to top and transferring top to bottom
121         top.add(bottom);
122         bottom.transferFrom(top);
123
124         // Updating view to reflect changes in model
125         updateViewToMatchModel(this.model, this.view);
126     }
127
128     @Override
129     public void processSubtractEvent() {
```

```
130
131         // Aliasing model top and bottom
132         NaturalNumber top = this.model.top();
133         NaturalNumber bottom = this.model.bottom();
134
135         // Subtracting bottom from top and transferring top to
    bottom
136         top.subtract(bottom);
137         bottom.transferFrom(top);
138
139         // Updating view to reflect changes in model
140         updateViewToMatchModel(this.model, this.view);
141     }
142
143     @Override
144     public void processMultiplyEvent() {
145
146         // Aliasing model top and bottom
147         NaturalNumber top = this.model.top();
148         NaturalNumber bottom = this.model.bottom();
149
150         // Multiplying bottom to top and transferring top to
    bottom
151         top.multiply(bottom);
152         bottom.transferFrom(top);
153
154         // Updating view to reflect changes in model
155         updateViewToMatchModel(this.model, this.view);
156
157     }
158
159     @Override
160     public void processDivideEvent() {
161
162         // Aliasing model top and bottom
163         NaturalNumber top = this.model.top();
164         NaturalNumber bottom = this.model.bottom();
165
166         // Dividing bottom from top and transferring top to bottom
167         NaturalNumber remainder = top.divide(bottom);
168         bottom.transferFrom(top);
169
170         // Transferring remainder to top
171         top.transferFrom(remainder);
```

```
172
173         // Updating view to reflect changes in model
174         updateViewToMatchModel(this.model, this.view);
175
176     }
177
178     @Override
179     public void processPowerEvent() {
180
181         // Aliasing model top and bottom
182         NaturalNumber top = this.model.top();
183         NaturalNumber bottom = this.model.bottom();
184
185         // Raising top to the bottom power
186         top.power(bottom.toInt());
187         bottom.transferFrom(top);
188
189         // Updating view to reflect changes in model
190         updateViewToMatchModel(this.model, this.view);
191
192     }
193
194     @Override
195     public void processRootEvent() {
196
197         // Aliasing model top and bottom
198         NaturalNumber top = this.model.top();
199         NaturalNumber bottom = this.model.bottom();
200
201         // Taking top to the bottom(th) root
202         top.root(bottom.toInt());
203         bottom.transferFrom(top);
204
205         // Updating view to reflect changes in model
206         updateViewToMatchModel(this.model, this.view);
207
208     }
209
210     @Override
211     public void processAddNewDigitEvent(int digit) {
212
213         // Aliasing model bottom
214         NaturalNumber bottom = this.model.bottom();
215
```

```
216         // Adding digit to bottom
217         bottom.multiplyBy10(digit);
218 
219         // Updating view to reflect changes in model
220         updateViewToMatchModel(this.model, this.view);
221 
222     }
223 
224 }
225 
```