```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) {

        /*
         * Get aliases to top and bottom from model
         */
        NaturalNumber top = model.top();
        NaturalNumber bottom = model.bottom();

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

        NaturalNumber temp = new NaturalNumber2(bottom);
        view.updateSubtractAllowed(top.compareTo(temp) >= 0);
        view.updateDivideAllowed(!bottom.isZero());
        view.updatePowerAllowed(bottom.compareTo(INT_LIMIT) <= 0);
        view.updateRootAllowed(
                bottom.compareTo(TWO) >= 0 && bottom.compareTo(INT_LIMIT) <= 0);
    }

    /**
     * Constructor.
```

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

```
116          */
117         NaturalNumber temp = new NaturalNumber2(bottom);
118         top.transferFrom(temp);
119         /*
120          * Update view to reflect changes in model
121          */
122         updateViewToMatchModel(this.model, this.view);
123     }
124
125     @Override
126     public void processAddEvent() {
127         /*
128          * Get aliases to top and bottom from model
129          */
130         NaturalNumber top = this.model.top();
131         NaturalNumber bottom = this.model.bottom();
132         /*
133          * Update model in response to this event
134          */
135         bottom.add(top);
136         top.clear();
137         /*
138          * Update view to reflect changes in model
139          */
140         updateViewToMatchModel(this.model, this.view);
141     }
142
143     @Override
144     public void processSubtractEvent() {
145         /*
146          * Get aliases to top and bottom from model
147          */
148         NaturalNumber top = this.model.top();
149         NaturalNumber bottom = this.model.bottom();
150         /*
151          * Update model in response to this event
152          */
153         top.subtract(bottom);
154         bottom.transferFrom(top);
155         /*
156          * Update view to reflect changes in model
157          */
158         updateViewToMatchModel(this.model, this.view);
159     }
160
161     @Override
162     public void processMultiplyEvent() {
163         /*
164          * Get aliases to top and bottom from model
165          */
166         NaturalNumber top = this.model.top();
167         NaturalNumber bottom = this.model.bottom();
168         /*
169          * Update model in response to this event
170          */
171         top.multiply(bottom);
172         bottom.transferFrom(top);
```

```
173         /*
174          * Update view to reflect changes in model
175          */
176         updateViewToMatchModel(this.model, this.view);
177     }
178
179     @Override
180     public void processDivideEvent() {
181         /*
182          * Get aliases to top and bottom from model
183          */
184         NaturalNumber top = this.model.top();
185         NaturalNumber bottom = this.model.bottom();
186         /*
187          * Update model in response to this event
188          */
189         NaturalNumber temp = new NaturalNumber2(top.divide(bottom));
190         bottom.transferFrom(top);
191         top.transferFrom(temp);
192         /*
193          * Update view to reflect changes in model
194          */
195         updateViewToMatchModel(this.model, this.view);
196     }
197
198     @Override
199     public void processPowerEvent() {
200         /*
201          * Get aliases to top and bottom from model
202          */
203         NaturalNumber top = this.model.top();
204         NaturalNumber bottom = this.model.bottom();
205         /*
206          * Update model in response to this event
207          */
208         top.power(bottom.toInt());
209         bottom.transferFrom(top);
210         /*
211          * Update view to reflect changes in model
212          */
213         updateViewToMatchModel(this.model, this.view);
214     }
215
216     @Override
217     public void processRootEvent() {
218         /*
219          * Get aliases to top and bottom from model
220          */
221         NaturalNumber top = this.model.top();
222         NaturalNumber bottom = this.model.bottom();
223         /*
224          * Update model in response to this event
225          */
226         top.root(bottom.toInt());
227         bottom.transferFrom(top);
228         /*
229          * Update view to reflect changes in model
```

```
230          */
231         updateViewToMatchModel(this.model, this.view);
232     }
233 
234     @Override
235     public void processAddNewDigitEvent(int digit) {
236         /*
237          * Get aliases to top and bottom from model
238          */
239         NaturalNumber bottom = this.model.bottom();
240         /*
241          * Update model in response to this event
242          */
243         bottom.multiplyBy10(digit);
244         /*
245          * Update view to reflect changes in model
246          */
247         updateViewToMatchModel(this.model, this.view);
248     }
249 
250 }
251 
```