***Second partial assignment for hw2.***

***Due date: October 23rd by 11:59pm***

Grade: 10% of the total grade (which is one third of the total 30% for hw2)

In this part of hw 2 the students should implement the rest of the MVC architecture, extending

the given classes.

This means that controllers should be written for the controller package and GUI classes (most

likely just the view classes) for the view package.

As you implement the rest of the MVC architecture you should test its functionality manually,

by running your application and checking its behavior.

You can consider this as acceptance testing, as if a client tests your application.

1. Controller package

Controller classes should extend AbstractController class.

There should be a controller class per a view.

So, at least two controllers: for AccountSelectionView and AccountView.

Also, there should be an instance of a controller per an instance of a view.

For instance, if two instances of AccountView are open on a single account then each should have

an instance of a controller.

A controller in the Calculator example can be used for reference. As you can see it has a dispatcher

method called "operation" that takes a string as input (this string describes the UI component) and decides

on a manipulation of the model in a series of "if then else" statements.

2. View package

View classes should extend the abstract JFrameView.

There should be at least two view classes, say, AccountSelectionView and AccountView.

While in general, classes for UI components other than the view (i.e. application window) can be

in the view package, in our simple application subclasses of JFrameView are enough.

You can use CalculatorView as an example to write subclasses of JFrameView.

It shows how basic Swing classes can be used to implement a UI with a JTextField, JButtons, a hierarchy of

UI components via JPanel classes with GridBagLayout.

It also shows the use of an ActionListener to implement interactivity.

AccountSelectionView will need a JComboBox too. Its example was in Java Foundations code, for instance, the JukeBox example.

Also use Calculator example to see how instances of MVC related classes are instantiated, the sequence in

which the instantiations are done.

You can place the main method in the AccountSelectionView class (similar to Calculator example) or

in a dedicated Main class.

Please note that the View classes register as listeners in the corresponding model classes

in the constructor of JFrameView class.

If your view class does not subclass from JFrameView they will not

register as listeners.

You can implement currency conversions in an AccountView class.

For acceptance testing:

1. Check that while several account windows are open on a single account,

they all change the balance properly if a withdraw or deposit is made in one of them.

This should check that MVC communication functions properly.

2. Please also test that withdrawing more funds than available results in an exception and

a corresponding pop-up window is shown.

3. If you close the account selection window, all the currently open windows of your application

should be closed.

4. When closing the application, the current account balances should be saved to a file.