Assessment Exercise 3
Programming
Date: 05/01/2019

Martha Kaitlin Garman
Student Number: 2402299G

Status Summary

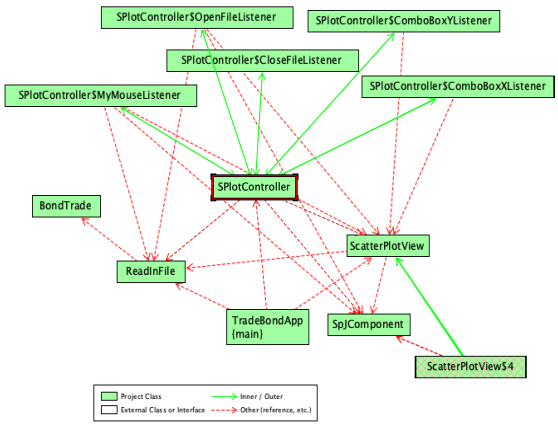
The program meets the specifications in that it 1) intakes a csv file with trade-bond information and draws graph to points, 2) all activeListeners (buttons/comboboxes/mouseListeners) are working to specifications, and 3) the layout is Model-View-Controller and is updated after receiving new information from the user. There is a +/-5% value allowance on button clicks or a +/-0.3 tolerance. If no data is uploaded (empty-graph), a mouse-click on the graph will handle an NullPointerException.

Assumptions

It was assumed the file would be in the same format, to include the same spelling/capitalisation of the column headers. It was also assumed that data would be uploaded in the same format (double in first column, int in second, and int in third); this should be checked or handled when the file is read-in. In addition, the paint component is not resizable as the other panels are. It was assumed file uploads would not contain numbers greater than 8 significant figures, other wise the numbers may eclipse the graph line.

Conclusions/Recommendations

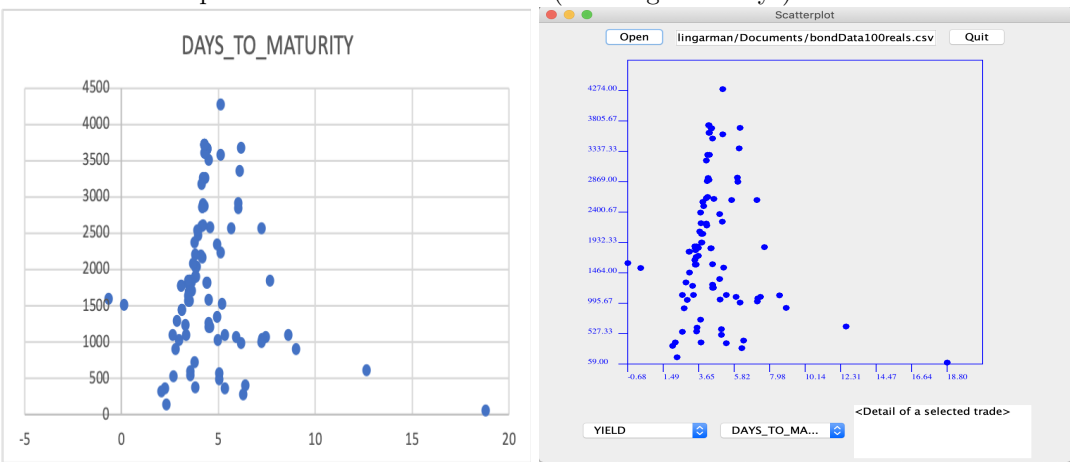
Although, exception handling was added to make sure the file read is in csv format, the program exits if true; a better handle would to allow the user to re-select a file name. Although tradeBond information of a point is returned after a mouse-click, if there is no correct x-value but a y-value is found within the arrayList, tradeBond information will load and appear incorrect. This can be fixed by rechecking the x-value where the index of the y-value was found against the xCoordinate. Some methods iterate through an arrayList, which may take more time depending on the length of the list; it would be more efficient to use an algorithm in the findIndex() method that implemented a key for a Hashmap to go straight to the value instead of iterating through multiple lists. Lines of code should be reduced by creating new classes.



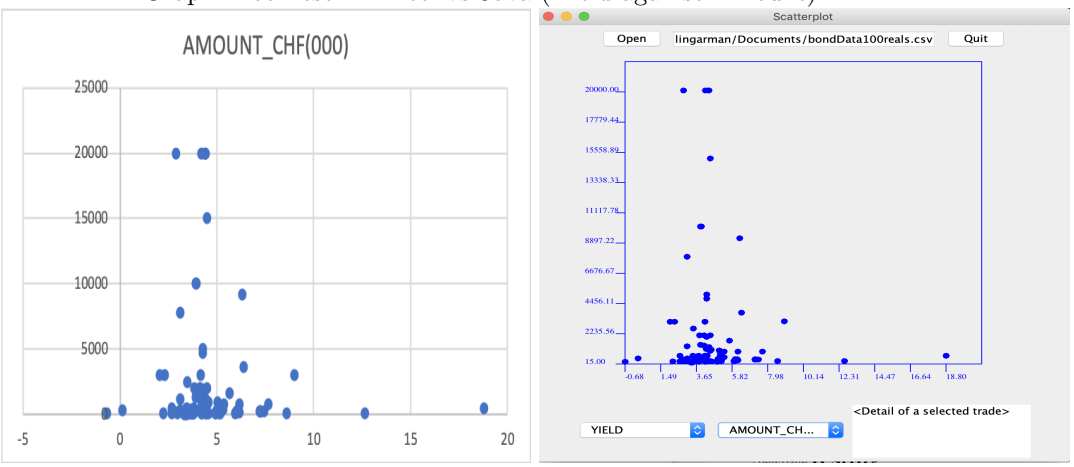
Testing

In order to test the "Open" button fileReader, the paint SpJComponent, and the comboBoxes, graphs were plotted in excel against graphs plotted within the program. Points should match to the file uploaded (Java graph included negative scale).

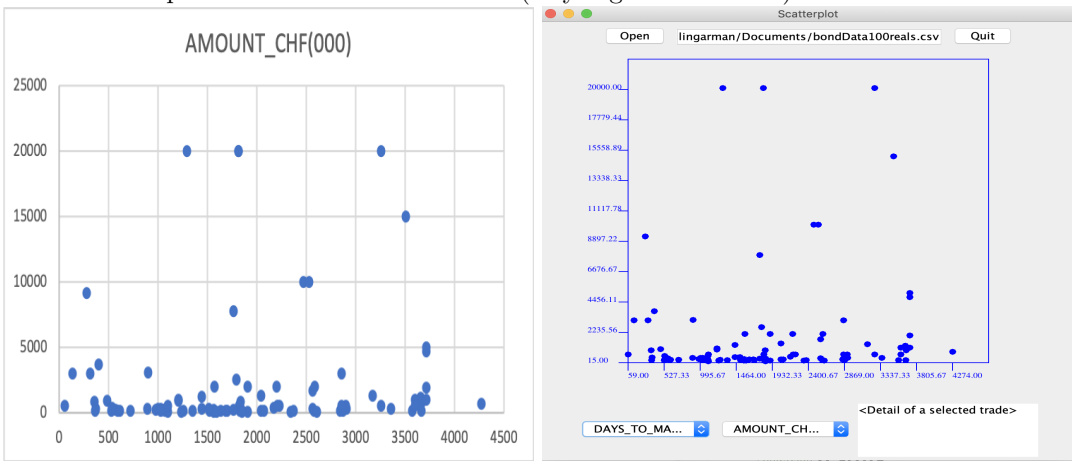
"Graph Plot Test 1: Excel vs Java (Yield against Days)"



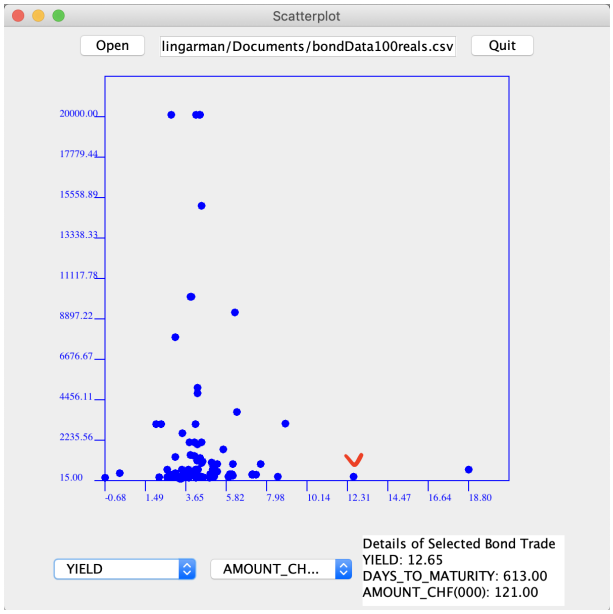
"Graph Plot Test 2: Excel vs Java (Yield against Amount)"



"Graph Plot Test 3: Excel vs Java (Days against Amount)"

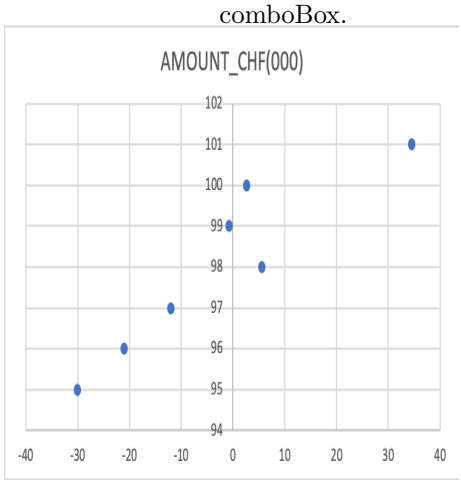


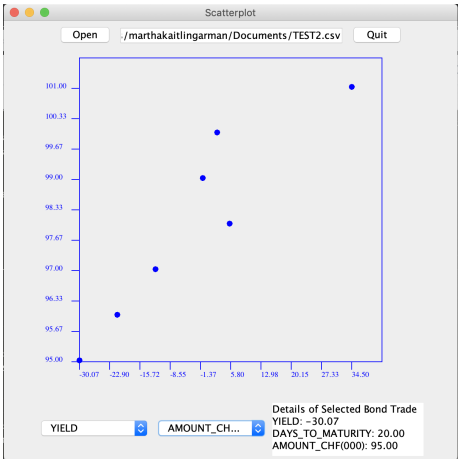
To test the mouseListener, selected points were compared to actual data within the csv file. The point below equals YIELD: 12.649, DAYS-TO-MATURITY: 613 AMOUNT: 121. A Decimal Formatter is used in the Java program to round to two decimal places.



Additional tests included making sure the "Quit" button exited the program as expected and new file uploads would clear all data and redraw using new points. The comboBoxes were reset to new fields. New points were then compared to an excel plot.

New File uploaded after populated by "BondData100reals", with amount selected for y-axis to test comboBox.





Exception Handling was also tested and found to work for:

- Clicking on an empty graph - NullPointerException
- Opening incorrect file format - InputMismatchException (exit program)
- Opening incorrect file format after loading correct file - InputMismatchException (exit program)

Attachments

- BondTrade.java
- ReadInFile.java (Model)
- SPlotController.java (Controller)
- ScatterPlotView.java (Viewer)
- SpJComponent.java
- TradeBondApp.java