Brianna Gerads

Refactoring Changes

Due: 02/15/2016

Method Extraction:

1. calculateCosts()
   1. The switch statement was specific to calculating the cost of a movie and could be pulled outside.
2. releaseBonus()
   1. The chunk of code was for adding a bonus for a two day new release rental so pulled it outside.
3. showFigures()
   1. The code fragment could be grouped together and would make the code cleaner.
4. printFooter()
   1. The code fragment could be grouped together and would make the code cleaner.

Creation of New Classes:

1. Statement.java
   1. I created this because the statement function needed refactoring and had some very function specific variables. Creating the new class made it easier to reference the variables and keep the functions clean.
2. Cost.java
   1. I created this because the cost was spread across Movie, Statement and Rental, however how the three classes were set up, it seemed like they are meant more to be “libraries” of information for these items. Creating this class allows for the calculations to be contained in one area.
3. FrequentRenterPoints.java
   1. The frequent renter points are specific to a customer, but can be calculated in any class and has a few functions of it’s own. Seperating it and making it it’s own class makes it easier to locate and track this information.

Moved Method Operations:

1. calculateCost()
   1. It was directly relevant to a Movie and
2. releaseBonus()
   1. It is relevant to tracking frequent renter points, so I moved it to the FrequentRenterPoints class.
3. statement()
   1. All of this was being done in Customer, but it made more sense to make its own class for clarity in the code and maintenance purposes so this function was moved to Statement.java.

Renaming Operations:

Replacement of Data Types:

1. Enumeration rentals
   1. Changed to a Vector to match the rest of the rental variables. I kept Vector because it has better storage management than an Enumeration.