ASSIGNMENT 2

Team: Jibin Prince, Michael Muruthi, Richard Padilla, Karl Dill, Francisco E Alderete

Software Refactoring:

Improve design of existing code without changing what it does
Making code easier to change
More loosely coupled
More cohesive modules
More comprehensive

1. At least 3 method extraction operations
    a. switch statement extracted from Customer to setPriceCode() function in Movie class
    b. Price calculation Extracted from switch statement into Price Subclasses
    c. frequesntRenterPoints becomes own method 51-58 getFrequentRenterPoints
    d. calculateFrequent has been extracted out of the Statement() method into the Price classes
    e. Extract print statement into htmlRentalReciept()
    f. Extract bonus rental points calculation out into a getCharge() method within Price.java
    g. result storage can also be a new method (print statement ???)
    h. Calculate rental price from 33-49 (Done ???)

2. At least 3 creation of new classes -> come up with argument for which is better from below

    Rationale: Want to separate concerns on 3 types of movies. If there is another Movie type we want to add, we can just add a new subclass to handle it.

    Furthermore we also wanted to seperate the concern of the Price from the Movie, so we chose to make an Abstract Price class with 3 concrete Price subclasses.

    Price.java, ChildrensPrice.java, RegularPrice.java, NewReleasePrice.java

3. At least 3 moving operations
    a. Moved Rental Price Switch logic to Movie class - shouldn't be done by the customer
    b. Moved Price Calculation into the Price superclass, since the price calculation is unique for each of the different movie categories, but very similar to the point that it only makes sense to have once

   c. Bonus Renter Point calculations got moved out of the Customer.java class into specific Price Subclasses.
4. At least 3 renaming operations
   a. rename statement() to htmlRentalReciept(). Name was ambiguous, and hurt understanding.
   b. Renamed several magic numbers from Movie.java. Without a name the purpose of the numbers was hard to guess.
   c. Renamed getFrequentRenterPoints() to getRentalPointValue(). Better clarifies what the method is doing.
5. 1-2 replacements of data types
   a. Changed Vector to List<Rentals>. Using an outdated data type could be unsafe or obsolete.
   b. Changed Enumeration data type into Rental type
6. A Main Method was implemented
7. New Print statement added was htmlRentalReceipt()