Lalith Vennapusa and Aarya Patel

CS 3354.HON

Professor Nguyen

12 April 2025

Explanation of Refactored Code for Movie-Rental

A) Method Extraction

1. Calculate rental price 🡪 Extracting this method from the switch statement in the statement method from the original program helps remove the code smell, being the switch statement, from the program, thus improving readability of the program as the method is not as large.
2. Calculate frequenter points 🡪 Extracting this method from the statement method of the original program, helps reduce the responsibilities of method and improves readability as the method is not as large (removes a code smell).
3. Output formatting 🡪 Extracting this method from the statement method of the original program, improves readability as the method is not as large (removes a code smell) and the output can be formatted in a formal structure.

B) New Classes

1. Movie.java 🡪 Movie is a super class to RegularMovie, NewReleaseMovie, and ChildrenMovie, which provides the method structure for calculating and returning the rental fee per movie and the amount of renter points given per movie.

2. RegularMovie.java 🡪 RegularMovie is a sub class to Movie and is used to calculate and return the rental fee and frequent renter points for a regular movie.

3. NewRealeaseMovie.java 🡪 NewReleaseMovie is a sub class to Movie and is used to calculate and return the rental fee and frequent renter points for a new release movie.

4. ChildrenMovie.java 🡪 ChildrenMovie is a sub class to Movie and is used to calculate and return the rental fee and frequent renter points for a children’s movie.

Creating the following classes helps in the refactoring process by making Customer more readable as it has less responsibilities to perform as a class, improving code organization, and improving the maintainability of program by removing code smells such as large methods and classes.

C) Method Move

1. Calculate rental price (From Customer to Rental) 🡪 Calculating the rental price should be the role of the Rental class not the Customer class, thus the method is moved to the Rental class.
2. Calculate frequenter points (from Customer to Rental) 🡪 Calculating the frequenter renter points should be the role of the Rental class not the Customer class, thus the method is moved to the Rental class.
3. Get rental price (from Movie to Movie subclasses) 🡪 Getting the rental price for a movie was individualized to the type of movie, thus the method was moved to each Movie subclass.

D) Renaming Operations

1. Statement() to generateTextStatement() and generateXmlStatement() 🡪 This change provides a more understandable output method which improves the readability of the program.
2. totalAmount to totalRentalFees 🡪 This change provides a more understandable variable name which improves the readability of the program. “totalAmount” is too ambiguous to what it could mean, but totalRentalFees directly states what the variable is for, being the total fee for all movie rentals.
3. thisAmount to rentalFee 🡪 This change provides a more understandable variable name which improves the readability of the program. “thisAmount” is too ambiguous to what it could mean, but rentalFee directly states what the variable is for, being the rental fee of the movie.
4. String result to String customerRentalStatement 🡪 This change provides a more understandable variable name which improves the readability of the program. “result” is too ambiguous to what it could mean, but customerRentalStatement directly states what the variable is for, being the output statement.

E) Replacement of Data Types

1. int priceCode to double priceCode 🡪 priceCode should not be an int as it stores the rental fee of a movie which can be a floating-point number (double). Thus, the data type was changed from an int to a double.