Inf2-SEPP 2025-26

Tutorial 5 (Week 8)

Good code, Code Smells, Refactoring

Study this tutorial sheet and make notes of your answers BEFORE the tutorial.

Introduction

The purpose of this tutorial is to get you to put into practice concepts that you have studied in the lectures on construction and refactoring. It also gives you a glimpse into some of the important types of tasks that will be part of your Coursework 3.

1 Task 1: Coffee calculator tool

1.1 Task 1.1: High quality code, code review

You are provided with the following very messy piece of code by a colleague who is developing a coffee caffeine estimation tool.

```
double getCaffeine(Coffee c, int gs){//getting the coffee
and glass size
//cases for the coffee type
 switch(c.type){case espresso: return c.getecaffeine();//just espresso
//cappuccino
//case cappuccino:return c.getecaffeine*0.33+c.getmilk*0.33 +
//c.getfoam()*0.37;
 case cappuccino:return c.getecaffeine()*0.33*gs;//a
 little espresso
//espresso plus chocolate
  case moka: return ( c.getecaffeine()*0.33 + c.getccaffeine()
  *0.17 )*gs;
case red_eye:return (c.getfcaffeine()*0.7+c.getecaffeine()*0.3)
 *gs;}//filter coffee plus espresso
throw new RuntimeException ("Incorrect type of coffee provided");}
```

1. Advise your colleague what to change to make it cleaner and easier to read.
2. Re-write the code. This will also help you in the following tasks.

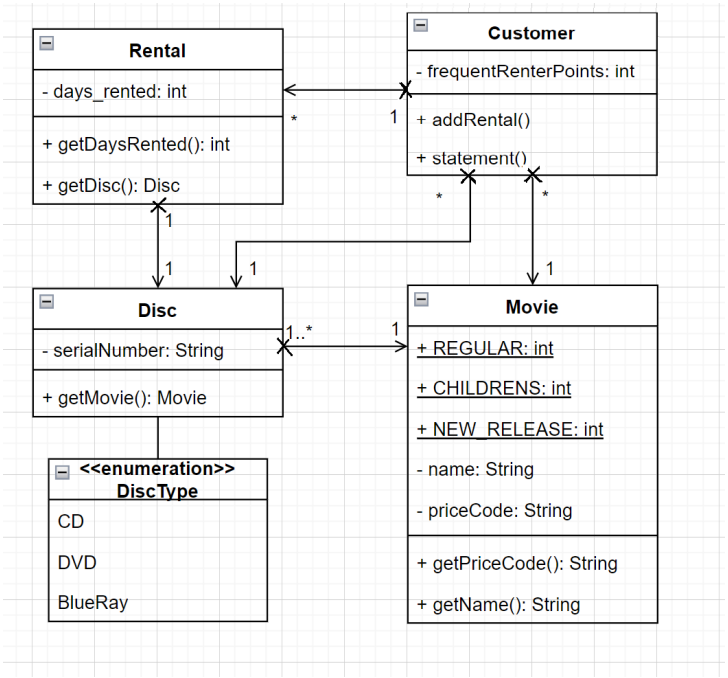
1.2 Task 1.2: Refactoring

Your colleague's tool described in the previous section is now on the market, and he tells you he must extend it for the numerous different coffee types known to international customers. Moreover, he must add estimations for calories and fat levels for each of the different coffee types.

1. Describe the type of refactoring that you would advise him to perform, and how it would work in practice.
2. Draw a UML class diagram to explain better your proposed changes to the design.

2 Task 2: Movie rental store [1]

You are provided with the following class diagram for a movie rental store's system, as well as with the code for the Customer class's statement() method, concerned with printing out a statement of a customer's charges at the store.




```
public String statement() {
  double totalAmount = 0;
  int thisFrequentRenterPoints = 0;
  String result = "Rental record for " + name + "\n";
  for (int i = 0; i < rentals.size(); i++) {
      double thisAmount = 0;
      Rental each = rentals.get(i);

      //determine amounts for each line
      switch (each.getDisc().getMovie().getPriceCode()) {
          case Movie.REGULAR:
              thisAmount += 2;
```

[1] inspired from Martin Fowler's refactoring example

```
                if (each.getDaysRented() > 2)
                    thisAmount += (each.getDaysRented() - 2) * 1.5;
                break;
            case Movie.NEW_RELEASE:
                thisAmount += each.getDaysRented() * 3;
                break;
            case Movie.CHILDRENS:
                thisAmount += 1.5;
                if (each.getDaysRented() > 3)
                    thisAmount += (each.getDaysRented() - 3) * 1.5;
                break;
        }
        totalAmount += thisAmount;

        // add frequent renter points for this rental
        thisFrequentRenterPoints ++;
        // add bonus for a two day new release rental
        if ((each.getDisc().getMovie().getPriceCode()== Movie.NEW_RELEASE) && each.getDaysRented() > 1)
           thisFrequentRenterPoints ++;
        //add new frequent renter points to the customer's total
        frequentRenterPoints + = thisFrequentRenterPoints;

        //show figures for this rental
        result += each.getDisc().getMovie().getName()+ ": " + thisAmount + "\n";

    }
    //add footer lines
    result +=  "You owe " + totalAmount + "\n";
    result += "On this rental you earned " + thisFrequentRenterPoints + " frequent renter points";
    return result;

}
```

Your company is required to address a new requirement: customers would also like to be able to print out the same statement in HTML format. The following is an example of how such a statement should be printed for a customer named Daniel Solza:

```
<H1>Rental record for <EM>Daniel Solza</EM></H1><P>
Lord of the Rings 1: 8<br>
Alien 3: 5<br>
Dune: 12<br>
<p>You owe <em>25</em><p>
On this rental you earned <em>25</em> frequent renter points<p>
```

Focusing on the method, but also the design of this system:

1. Can you notice any *code smells* in the method's code, or in how the rest of the code would look based on the class diagram? Why are they a problem? A useful set of code smells is described at this link.
2. What would be a naive solution to adding functionality for printing the same statement in HTML format? Why is it not a good solution?
3. What refactorings would you do to improve the design of this system, and make it easier to add the new functionality? Get inspiration from Martin Fowler's catalogue of refactorings available here.
4. Make your changes in the class diagram and the code related to the functionality of printing the two statements.