**Homework 1**

**Code Smells**

In this homework, you are given a simple program to support the management of a movie rental place.

You are required to identify 10 instances of bad smells in that program.

Please copy the piece of code for each instance into a word file and provide your explanations on why that piece of code has bad smell.

1. Constants are used in this code instead of subclasses.  
  
public static final int CHILDRENS = 2;

public static final int REGULAR = 0;

public static final int NEW\_RELEASE = 1;

2. The statement() method is quite large and could be reduced in size. While it fits on one screen, it is too large for what it’s intended to do.

public String statement() {

…50 lines…

return result;

}

3. We could use subclasses again instead of switch statements to improve logic flow (i.e. split up functionality)

switch (each.getMovie().getPriceCode()) {

case Movie.REGULAR:

thisAmount += 2;

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;

}

4. Don’t use magic numbers; if you have to use a constant, define it elsewhere (i.e. not the method)

thisAmount += (each.getDaysRented() - 2) \* 1.5;

5. The “frequentRenterPoints” variable doesn’t do anything long-term.

int frequentRenterPoints = 0;

result += "You earned " + String.valueOf(frequentRenterPoints)

6. Using floating point numbers are bad when dealing with currency (or anything else that needs to be accurate)

double totalAmount = 0;

7. Price for a movie should be in the rental class, grouping functionality used together

public Movie(String title, int priceCode) {

\_title = title;

\_priceCode = priceCode;

}

public int getPriceCode() {

return \_priceCode;

}

8. The statement() method should leave all of the logic calculations to the other classes; it doesn’t need to be in the Customer class

case Movie.REGULAR:

thisAmount += 2;

if (each.getDaysRented() > 2) {

thisAmount += (each.getDaysRented() - 2) \* 1.5;

}

break;

9. There’s no way to set the movie or days rented in the Rental class without creating a new instance of it.

public int getDaysRented() {

return \_daysRented;

}

public Movie getMovie() {

return \_movie;

}

10. No way to remove a customer’s rental when they return a rental.

public void addRental(Rental arg) {

\_rentals.addElement(arg);

}