

## CS130 Software Engineering: Design Review Activity: Mortgage Calculator

Study the following source code, Mortgage Calculator. Critique the design with respect to its use of information hiding principle.

**P** = principal, the initial amount of the loan

**I** = the annual interest rate (from 1 to 100 percent)

**L** = length, the length (in years) of the loan, or at least the length over which the loan is amortized.

The following assumes a typical conventional loan where the interest is compounded monthly. First, I will define two more variables to make the calculations easier:

**J** = monthly interest in decimal form = **I** / (**12** x **100**)

**N** = number of months over which loan is amortized = **L** x **12**

Okay now for the big monthly payment (**M**) formula, it is:

$$M = P \times \frac{J}{1 - (1 + J)^{-N}}$$

```
public class MortgageCalculator {
    double payment, principal = 200000;
    // Principle amount of loan is $200,000
    double annualInterest = 0.0575;
    // Interest rate is currently 5.75%
    int years = 30; //Term of the loan is 30 years

    public static void main (String[] args){
        MortgageCalculator calculator = new MortgageCalculator();
        if (args.length == 3) {

            double principal = Double.parseDouble(args[0]);
            double annualInterest = Double.parseDouble(args[1]);
            int years = Integer.parseInt(args[2]);
            calculator.principal= principal;
            calculator.annualInterest= annualInterest;
            calculator.years= years;
            calculator.print(principal, annualInterest, years);
        }
    }
}
```

```

        public static double calculatePayment(double principal, double annRate,
int years){
// using monthly compounding method internally, but that's not communicated in the
API.
        double monthlyInt = annRate / 12;
        double monthlyPayment = (principal * monthlyInt)
            / (1 - Math.pow(1/ (1 + monthlyInt), years * 12));
        //Shows 1 monthly payment multiplied by 12 to make one complete
year.
        return format(monthlyPayment, 2); // round upto cents.
    }
    public static double format(double amount, int mortgage) {
// the public API description is confusing
        double temp = amount;
        temp = temp * Math.pow(10, mortgage);
        temp = Math.round(temp);
        temp = temp/Math.pow(10, mortgage);
        return temp;
    }
    public void print(double pr, double annRate, int years){
// the API does not say anything about monthly payment but uses monthly payment
calculation
        double mpayment = calculatePayment(pr, annRate, years);
        System.out.println("The principal is $" + (int)pr);
        //Shows the principle amount in $ value.
        System.out.println("The annual interest rate is " + format(annRate
* 100, 2) + "%");
        System.out.println("The term is " + years + " years");
        //Term is normally in years.
        System.out.println("Your monthly payment is $" + mpayment);
        //Shows output of monthly payment.
    }
}

```

- (1) What kinds of secrets are currently hidden by MortgageCalculator?
- (2) What changes could you anticipate? List new features that you may want to add to the current mortgage calculator.
- (3) Which methods and fields need to be updated to accommodate such changes?
- (4) Critique the current code in terms of capability to support independent development.

(5) Critique the current code in terms of readability/comprehensibility.