Activity 3 (Day 2) Hands-on refactoring

Refactoring task 1

Example code:

```
// part of an Adder class
public int addAllSquaresOfOddsUp(int[] numbers) {
    int sum = 0;
    for (int i = 0; i < numbers.length; i++) {
        int number = numbers[i];
        if (number % 2 == 1) {
            sum += number * number;
        }
    }
    return sum;
}
// ... elsewhere ...
// adder is an instance of the Adder class
int total = adder.addAllSquaresOfOddsUp(theNumbers);
```

Extract method tasks Each of the following tasks asks you to extract some part of the presented code into a method.There are 5 questions to answer each time:

- Is it even possible to do this? Is the code in question changing more than 1 local variable? Do we need to change its form a bit (i.e. not simply a method call, but maybe assigning to a variable the result of a method call)?
- What would be a good name for this method?
- What would be the return type of this method?
- What parameters would the method take? What would good names for them be?
- What would be its code?

1. Replace the part that says `number % 2 == 1` with a method call.
2. Replace the part that says `number * number` with a method call.
3. Replace the whole `sum += number * number;` part with a method call.
4. Replace the whole `if (...) { ... }` part with a method call.
5. Replace the whole body of the `for` loop with a method call.
6. Replace the whole body of the `addAllSquaresOfOddsUp` method with a method call.

Other tasks

1. If we wanted to *inline* the local variable number so that we don't have it any more, can it be done? How would the code change?
2. Explain why we cannot *inline* the local variable sum so that we don't have it any more.
3. If we wanted to provide the sum local variable as a parameter instead, how would the code change? Would it still work?
4. We want to replace the whole body of the addAllSquaresOfOddsUp method with: Creating a new object of a new class and providing the numbers array as a constructor argument, then calling an invoke method to perform the steps in the body of addAllSquaresOfOddsUp method and returning the result of that invoke call.
5. We want to *inline* the addAllSquaresOfOddsUp method into its call so that it is not getting called any more. How would the last line (adder.addAllSquaresOfOddsUp(theNumbers);) change? What are some things we will need to watch out for when we do this?

Refactoring task 2

Example code:

```
// Method in a Printer class
public void print(int amount, String currency) {
    System.out.println(String.format("%d %s", amount, currency));
}

// ... elsewhere ...
// printer is an instance of the Printer class
printer.print(100, "USD");
```

This is a good example of the "extract parameter object" refactoring. Thinking about it, the parameters amount and currency should always go together, as they in effect represent the amount of *money* we have. Extracting parameters that should go together is a common way to discover new classes.

1. Change the code to have the following features:
   - There is a new Money class, with fields amount and currency and a constructor that takes the same as parameters and assigns them to the corresponding fields. The fields are public (for now).
   - The print method takes a single parameter of type Money, called money. In the body of the method, references to amount and currency are replaced by money.amount and money.currency instead.
   - The call to printer.print at the end takes as input a call to the Money constructor, passing the amount and currency in to the constructor instead.
2. Make the amount and currency fields of Money private, and provide getters for them. Adjust the code for the print method accordingly.

3. Create a new static method in Money so that the argument to our call to print can instead be Money.usd(100).

4. The string to be printed right now likely looks like this: String.format("%d %s", money.getAmount(), money.
   Extract a format method from this, and adjust the code.

5. Move the format method so that it is an instance method of the Money class. Adjust the code to match (it should take no arguments as an instance method to Money).

6. Inline the amount and currency getter usages, so that the method format uses the fields directly.