# Part 7: Extracting Methods

## Solution Code

The refactoring is straightforward and you shouldn't have to do anything beyond selecting the correct statements. Note if you selected the wrong statements, then this can go very wrong!

But hopefully, you didn't and could get it done without any issues! If not, just have another go by reusing the code from the previous module as your starting point.

The inlining call just tightens things up a bit and makes it neater.

Notice that we now have methods which do specific things and so our code is more readable inside the `main` method itself. Also, it's very clear what each method does by the way it's named. That's another key aspect of clean code - make your code as readable as possible by using sensible names and extracting methods wherever you can. The reader can then read the program at a higher level of abstraction by not getting lost in the details if everything's just in one big 'this does everything' method.

### Code Listings

#### App.java

```
package com.javaeasily.demos;

public class App {
    public static void main(String[] args) {
        System.out.println("Loan Calculator".toUpperCase());
        System.out.println();

        int amount = 100;
        int years = 5;
        double interestRate = 10;

        if (amount > 0 && years > 0 && interestRate > 0.0) {
            printInputs(amount, years, interestRate);
            printResult(calculateRepaymentAmount(amount, years, interestRate));
        } else {
            System.out.println("Invalid values - cannot calculate repayment amount
.");
        }
    }

    private static void printInputs(int amount, int years, double interestRate) {
        System.out.println("Calculating loan based on:");
        System.out.println("Principal:       " + amount);
        System.out.println("Loan Term:       " + years + " year" + ((years > 0) ?
"s" : ""));
        System.out.println("Interest Rate:   " + interestRate + "%");
    }

    private static double calculateRepaymentAmount(int amount, int years, double i
nterestRate) {
        double interestRateMultiplier = 1 + interestRate / 100;

        double currentAmountPayable = amount;
        int currentYear = 1;
        while (currentYear <= years) {
            currentAmountPayable = currentAmountPayable * interestRateMultiplier;
            currentYear++;
        }
        return currentAmountPayable;
    }

    private static void printResult(double currentAmountPayable) {
        String totalAmountDue = Double.toString(currentAmountPayable);
        int indexOfDecimalPoint = totalAmountDue.indexOf(".");
        String totalAmountDueFormatted = totalAmountDue.substring(0, indexOfDecima
lPoint+3);
        System.out.println("Total Amount Due: " + totalAmountDueFormatted);
    }

}
```