

cs171: Introduction to Programming
Programming Assignment 8

Due Date: Monday, March 27

Points: 100

When you leave Denison, many of you will probably buy a car. You might be surprised at how much a monthly payment is even for economy cars. In this program we practice creating methods in the context of a program that computes monthly payments for a car loan.

The program is divided into three sets of requirements. Completing the first set earns 75 out of 100 possible points. Adding the second set of requirements earns an additional 10 points (85 out of 100). Completing the last set earns up to the remaining 15 points (up to 100 out of 100). Each set of requirements is more difficult than the previous. Completing the last set is "outstanding work" meriting an A grade.

- There are four parts to any loan:
 - The principal is the amount you borrow.
 - The interest rate determines how much the principal grows as the length of the loan progresses (this is how banks make money from a loan).
 - The term is the number of payments. For example, a term of 48 mean 48 monthly payments, one per month for four years.
 - The monthly payment is how much you need to pay the bank at the end of each month.

The interest rate is a bit tricky. Suppose you borrow money at 7.1% interest. This means your principal grows at 0.071 per year. But the rate is computed monthly, not annually. So we divide this rate by 12 to obtain a monthly interest rate of $0.071/12 \approx 0.005917$.

For example, suppose you borrow \$10,000 at 7.1% interest for 48 months. At the end of the first month, we compute the new principal as follows.

$$10000 + 10000 * 0.005917 = 10059.17$$

Thus your principal has grown by about \$59 dollars and 17 cents. You now write a check for \$430 and so your new principal at the end of the month is \$9,629.17. This continues on and on until the loan is paid off. If you compute the monthly payment correctly, then the balance will be 0 after the 48th payment. Notice that each month the interest is always computed and added before you subtract your payment.

PART 1 (75 points)

- Your program will have five methods including `main`. Three of the other methods will be called from `main` to enter values (principal, interest rate, and monthly payment). The last method will compute how much money you still owe at the end of the loan period.
- You will need to prompt the user to enter a principal. Write a method called `getPrincipal()` that prompts the user for a principal and reads in this value. Return the amount entered by the user as your return value. What type do you think your method should return? Be sure to include the word "principal" as part of your prompt.

Have two input parameters to your method. The first parameter is of type `double` and is a lower bound on the principal. The second parameter is of type `double` and is an upper bound on the principal. Be sure the value your user enters is within the lower and upper bound. If the user's value is outside this range, then keep prompting them over and over until they enter a value within range. Use a range of 10,000 to 20,000 dollars (these are your actual parameters).

- You will need to prompt the user to enter an interest rate. Write a method called `getRate()` that prompts the user for an interest rate and reads in this value. Return the amount entered by the user as your return value. What type do you think your method should return? Be sure to include the words "interest rate" as part of your prompt.

Have two input parameters to your method. The first parameter is of type `double` and is a lower bound on the rate. The second parameter is of type `double` and is an upper bound on the rate. Be sure the value your user enters is within the lower and upper bound. If the user's value is outside this range, then keep prompting them over and over until they enter a value within range. Use a range of 7% to 12% as actual parameters.

- You will need to prompt the user to enter a monthly payment amount. Write a method called `getPayment()` that prompts the user for a monthly payment and reads in this value. Return the amount entered by the user as your return value. What type do you think your method should return? Be sure to include the word "payment" as part of your prompt.

Have two input parameters to your method. The first parameter is of type `double` and is a lower bound on the payment. The second parameter is of type `double` and is an upper bound on the payment. Be sure the value your user enters is within the lower and upper bound. If the user's value is outside this range, then keep prompting them over and over until they enter a value within range. Use a range of 100 to 900 dollars.

- We will always use a term of 48 (assume a four-year loan). Do not provide the option for the user to change this value.
- Write a method called `computeResidual()`. This purpose of this method is to calculate the balance left at the end of the loan (after all 48 payments have been paid). Use a loop to adjust the principal month by month. Remember, first add the interest and then subtract the payment for each month. Return the residual loan amount. Note: the residual may be negative if the monthly payment is too high.

PART 2 (10 pts)

- Notice that you have three methods which are essentially the same. They only differ in their message (words of principal, rate, and payment) and in the values for their upper and lower bounds. It is good programming practice to reduce the number of such redundant methods. See if you can combine these three methods into one method that is flexible enough to prompt the user for all three values. Then call it three times from the main program. Hint: think about adding a `String` parameter as a third input to the method to help with the different prompts. Choose a name for your method that is appropriate. (Note: you should comment out the three old methods so that I can see you finished PART1).

PART 3 (15 pts)

- It is quite likely that the payment value entered by the user is too small or too large. If the payment is too small there will be a positive residual at the end

of the loan (still some principal left to be paid). If the payment is too large, there will be a negative principal (you paid too much). In this third part, you will need to search for a monthly payment that is appropriate for the loan.

In the perfect case, you would like your principal to be exactly \$0.00 after the 48th payment. If your monthly payment is limited to dollars and cents (not fractional cents), it is likely that this will not work out exactly; you will either have a little left over or a little too much taken away. The goal is to find a payment such that 47 of the payments are the same and the last is the same or slightly less than the monthly payment. We will call this monthly payment the optimal monthly payment: it is the smallest monthly payment that leaves a negative (or zero) residual.

For example, assume we borrow \$10,000 for 48 months at an annual rate of 7%. The optimal monthly payment will be \$237.47 for 47 months with a final payment of \$237.05 (slightly less than the optimal monthly payment). If we were to only pay \$237.46 per month, then we would still have a final balance that we owe (a very small one).

In this third part of the assignment, start with the payment entered by the user and then search to find the optimal payment. You can do this in many ways. A very simple way is to increase or decrease the payment little by little until you find the optimal value. This will make many calls to the `computeResidual()` method. A better way makes fewer calls. See if you can find a procedure to search for the optimal payment using as few calls to `computeResidual` as possible. The number of points you receive is dependent on the efficiency of your procedure.

Have your program output the optimal monthly payment and the final payment. For example: \$237.47 for 47 months, with \$237.05 due on the final month.

Name your program `LoanCalculator.java` and email me the source code before the start of class on Monday, March 20. In the comment block at the top, add a comment to let me know what PARTs you have implemented. For example, "I implemented PART 1 and 2".