**COSC 2050 (JAVA Programming I)**

Assignment 2, (Due 09/12/2017, before class)

## Console

Welcome to the Loan Calculator

DATA ENTRY

Enter loan amount: ten

Error! Invalid decimal value. Try again.

Enter loan amount: -1

Error! Number must be greater than 0.0

Enter loan amount: 100000000000

Error! Number must be less than 1000000.0

Enter loan amount: 500000

Enter yearly interest rate: 5.6

Enter number of years: thirty

Error! Invalid integer value. Try again.

Enter number of years: -1

Error! Number must be greater than 0

Enter number of years: 100

Error! Number must be less than 100

Enter number of years: 30

FORMATTED RESULTS

Loan amount: $500,000.00

Yearly interest rate: 5.6%

Number of years: 30

Monthly payment: $2,870.39

Continue? (y/n):

Error! This entry is required. Try again.

Continue? (y/n): x

Error! Entry must be 'y' or 'n'. Try again.

Continue? (y/n): n

## Operation

* The Data Entry section prompts the user to enter values for the loan amount, yearly interest rate, and number of years. If the user doesn’t enter data that’s valid, this section displays an appropriate error message and prompts the user again.
* The Formatted Results section displays a formatted version of the user’s entries as well as the formatted result of the calculation.
* The application prompts the user to continue.
* Save your program as LoanCalculator.java.

## Specifications

* The formula for calculating monthly payment is:

double monthlyPayment =  
 loanAmount \* monthlyInterestRate/  
 (1 - 1/Math.pow(1 + monthlyInterestRate, months));

* The application should accept decimal entries for the loan amount and interest rate entries.
* The application should only accept integer values for the years field.
* The application should only accept integer and decimal values within the following ranges:

Greater Less  
 Than Than  
Loan amount: 0 1,000,000  
Yearly interest rate: 0 20  
Years: 0 100

* The application should only accept a value of “y” or “n” at the Continue prompt.
* If the user enters invalid data, the application should display an appropriate error message and prompt the user again until the user enters valid data.
* The code that’s used to validate data should be stored in separate methods. For example:

public static double getDoubleWithinRange(Scanner sc, String prompt,  
 double min, double max)

public static int getIntWithinRange(Scanner sc, String prompt,  
 int min, int max)