Project 7 (50 points)
Due FRIDAY, November 18 by midnight

Assignment Description: Design and develop a MVC solution to the previous project 5 program. **(Reminder: Project 5 Solution posted on KSOL)**

Implementation Requirements:

Your project will contain **three** classes: ***Mortgage (Model), IO (View)*** and ***Proj7 (Controller)***.

Mortgage is basically finished for you. It contains private instance variables for the three values needed to calculate a mortgage, one to hold the monthly payment, and one to hold the total amount paid on the loan. It also contains two constructors – a no argument used to initialize the values for a promotional loan and a 3-argument to instantiate a unique loan. It also defines the following methods:

// Calculate and set the monthly payment for the loan
public void setMonthlyPayment()

// Calculate and set the total payment for the loan
public void setTotalPayment()

// Display formatted output
public void displayInfo()

- Modify the ***displayInfo*** so that it *returns a string* used to display (rather than printing within the method) and rename: **public String toString() …** Do ***not*** print within the *toString* method

---

The **IO** class will contain **only** the *Input-Output* portion of the program. I will leave it up to you how this class is designed, but I would suggest that you follow the *40-20-40* rule discussed in class and sketch out a design of your class (i.e. what methods it will contain, what each method will do, the method signature of each, etc.) before you actually start coding. You will not submit this design but it will help the coding process go quicker in the long run. **This class will include all input data validation using the criteria listed below**. In addition, your program should also handle a *character* being entered for all numeric input or a *double* being entered for an *int* (see sample run on next page)

1) **Valid menu choices are 1-3**
2) **Valid interest rates are between 1%-9% (inclusive)**
3) **Valid terms are 5-50 years (inclusive)**
4) **Valid amounts are $50,000 to 1 million (inclusive)**

The driver/controller class (***Proj7***) will contain only a main method with **NO print statements, data validation** or **input statements** *(i.e. s.nextLine)*. Communicate between the *Model* and *View* classes by creating objects in the *Proj7* class. You must call the constructors/methods defined in the *Mortgage* class on both types of loans (i.e. call *setMonthlyPayment* to determine the monthly payment for a promotional or unique loan). Format all currency values with '$'-signs, commas, and two values after the decimal.

A ***possible*** execution of your program might look like the following…

```
Please choose from the following choices below:
        1) Promotional Loan ($100,000 @ 5.5% for 15 years)
        2) Unique Loan (enter in loan values)
        3) Quit (Exit the program)

        Please enter your selection (1-3): a
Input must be an integer.  Try again: a
Input must be an integer.  Try again: 1

PROMOTIONAL LOAN...:
The monthly payment is $817.08
The total payment is $147,075.02


Please choose from the following choices below:
        1) Promotional Loan ($100,000 @ 5.5% for 15 years)
        2) Unique Loan (enter in loan values)
        3) Quit (Exit the program)

        Please enter your selection (1-3): 22
                Invalid Choice. Please select 1, 2, or 3: 2

Please enter in the following information...
  Enter yearly interest rate (Ex: 8.25): a
Input must be numeric only.  Try again...
  Enter yearly interest rate (Ex: 8.25): .5
        Valid Interest Rates are 1% - 9%
        Please re-enter valid yearly interest rate (Ex: 8.25): a
Input must be numeric only.  Try again...
  Enter yearly interest rate (Ex: 8.25): 5.5

  Enter number of years for the loan (5-50): a
Input must be an integer.  Try again...
  Enter number of years for the loan (5-50): 2
        Valid Loan Terms are 5-50
        Please re-enter valid number of years: 25

  Enter loan amount without $ or commas (Ex:120000): a
Input must be numeric only.  Try again...
  Enter loan amount without $ or commas (Ex:120000): a
Input must be numeric only.  Try again...
  Enter loan amount without $ or commas (Ex:120000): a
Input must be numeric only.  Try again...
  Enter loan amount without $ or commas (Ex:120000): 123000


UNIQUE LOAN...:
The monthly payment is $755.33
The total payment is $226,598.28


Please choose from the following choices below:
        1) Promotional Loan ($100,000 @ 5.5% for 15 years)
        2) Unique Loan (enter in loan values)
        3) Quit (Exit the program)

        Please enter your selection (1-3): 3

PROGRAM COMPLETE...
```

Documentation:
Put a description of the project at the top of the file **and at the top of each method**. Please use this template for the top of the file:

```
/**
 * (description of the project)
 *
 * @author (your name)
 * @version (which number project this is)
 */
```

Please use this template for the top of each method:

```
/**
 * (description of the method)
 *
 * @param (describe first parameter)
 * @param (describe second parameter)
 * (list all parameters, one per line)
 * @return (describe what is being returned)
 */
```

Submission:
To submit your project, first create a folder called ***proj7*** and move or copy your completed *Proj7.java, IO.java* **and** *Mortgage.java* files into that folder. Then, right-click on that folder and select "*Send To → Compressed (zipped) folder*". This will create the file ***proj7.zip***.

Go to "*Files and Content->Modules->Submit Projects Here*" on K-State Online. Select your lab time and upload the ***proj7.zip*** file. **Put your name and Project 7 in the description box.**

Grading:

Programs that do not compile will receive a grade of 0.

Requirement	Points
Proj7 class contains only a main method. Contains NO print statements, data validation or input statements (i.e. s.nextLine)	**20**
• Properly creates objects of the *Mortgage* and *IO* classes.	
• Properly *calculate monthly payment* method for promotional loans	
• Properly calls *calculate monthly payment* method for unique loans	
• Properly calls the other methods defined in the *Mortgage* and *IO* classes.	
Mortgage class – modify *displayInfo* method so it simply returns a string (***public String toString***) – does ***not*** print within this method	**3**
IO class contains **only** the *Input-Output* portion of the program. All data validation is done inside this class.	**20**
• Properly validates the range of values for the menu choice, loan amount, interest rate, and length of the loan	
• Properly handles *character* input on all numeric input and doubles on integer input	
Prints result by calling the *toString()* method – **value** format exactly matches example (‘$’-sign, Comma, two decimal places on all values)	**3**
Loops until user chooses to quit	**2**
Documentation/naming/submission	**2**
Total	**50**