Regis University CC&IS
CS210 Introduction to Programming
Java Programming Assignment 5: Objects and Decisions

Your last Alice program implemented decision making.

This assignment will apply the same concepts to Java, starting with the code from your last Java program (Java Assn 4), which implemented objects. The Java Assn 4 program ran all of the code sequentially.

Decision statements will now be used to run some of the statements conditionally, by implementing at least one of each of the following conditional programming constructs: **if** statement or **if/else** statement, ***multiple alternative*** **if** statement, ***nested*** **if** statement, and **switch** statement.

> WARNING: Again, this assignment will be ***more challenging*** than the previous assignments, so make sure to ***start early*** and ***allocate enough time*** to complete it, including time you might need to seek any necessary help.

Problem Summary

Suppose you wanted to further modify your **Mortgage Calculator**, to make it more versatile. Starting with Java programming assignment 4, you could add the following features:

- Let the user decide on the down payment percentage amount to use.
- Calculate and display mortgage payments for both a 20-year and a 30-year loan, to allow the user to compare them.
- Have the program use the mortgage company's guidelines to set the interest rate for a mortgage.
  - The mortgage company sets interest rates using the following chart, based on the length of a loan and the loan type.
  - Loans up to $417,000 are called "conforming" loans. Loans over that amount are considered "jumbo" loans, and have different interest rates.

Interest Rate Chart	**Conforming Loans**	**Jumbo Loans**
30-year fixed	4.5%	4.125%
20-year fixed	3.85%	3.5%

Overview of Program

This program will still contain two classes, in two separate files within your project:

- A modified **MortgageLoan** class to define the data fields and methods for a MortgageLoan object, containing:
  - ***Seven*** data field definitions for a MortgageLoan object
  - ***Two* constructor** methods to create a MortgageLoan object (one constructor without parameters and one constructor with parameters).
  - ***Six* setter** methods to set the values for six of the data fields
  - An instance method to compute the monthly property tax.
  - An instance method to compute the monthly insurance premium.
  - An instance method to compute the monthly principle and interest loan payment.
  - An instance method to display one loan's information.
  - An instance method to calculate and display a loan's payment information.

- The main **MortageCalculator** class, modified to define only ***two*** methods, containing:
  - A static method to display a description of what the program will do.
  - A **main** method to display the program description, to read the inputs from the user, to create ***two*** **MortgageLoan** objects, and to call the other methods to display loan and payment information.

NOTE: For an example of a Java program containing decisions, see Online Content section 11.16.

Program Requirements

Modify the program you wrote for Java Assignment 4, as follows:

1. ***Within*** the **MortgageLoan** class:

- Add the following additional **private** data field:
  - ***char*** loan type (will store 'C' for conforming or 'J' for jumbo)
- Add a setter for the loan type data field. This method will:
  - Define a local constant and set it to hold the upper limit for conforming loan amounts.
  - Set the loan type data field to 'C' for Conforming.

    Then use an **if** statement to decide whether to re-set the data field to 'J' for Jumbo, depending upon the value stored in the loan amount data field.
- Modify the setter for the loan identifier data field.
  - The setter will now only have ***one*** parameter, the last name (will no longer use the zip code).
  - The loan identifier field will be created using the first 6 letters (uppercased) of the last name.
    - If the last name has less than 6 letters, the code should append Xs to the end of the name, so that there will be 6 letters total.
  - Modify the body of the setter method to:
    - Declare an integer constant to hold the loan identifier's String length of 6.
    - Declare a second String constant to hold six Xs ("XXXXXX").
    - Use these constants and the last name to create the loan identifier, WITHOUT using brute force.
  - NOTE: You will need to check the length of the last name BEFORE trying to extract letters and then use a **decision** statement to decide which letters to extract. Otherwise the program may crash when there are less than 6 letters in the last name.
- Modify the setter for the loan annual interest rate
  - The user will no longer enter the annual interest rate. Instead, the setter will use a **nested if** statement to set the annual interest rate data field, according to the chart on the previous page.
    - Check the loan type data field.
    - If the loan is a conforming loan, decide which rate to use based on the rates for conforming loans and the loan length data field.
    - Otherwise, decide which rate to use based rates for jumbo loans and the loan length data field.

- Modify the setter for the down payment percent
    - Add a ***char*** input parameter, to pass in the user's letter choice of down payment.
    - Use a **switch** statement to set the down payment percentage as follows:

        N (no down payment) – set the percentage to 0
        L (low down payment) – set the percentage to 10%
        S (standard down payment) – set the percentage to 20%
        H (high down payment) – set the percentage to 30%

      If the parameter contains any other letter, set the percentage to 0, and display an error message telling the user that since his/her choice was invalid, no down payment will be applied.

- Add a ***second*** **constructor** to instantiate a **MortgageLoan** type object.

  The new constructor with have ***four*** parameters:

    - the home buyer's last name (String)
    - the home value
    - the down payment percentage choice (char)
    - the loan length in years

  The **constructor** will:

    - Assign values to the home value and loan length data fields, using the constructor parameters for the values.
    - Call the modified setters for the loan identifier and the down payment percent, using the constructor parameter as the setter arguments.
    - Call the original setter for the loan amount (no arguments – uses data fields values)
    - Call the new setter for the loan type (no arguments – uses data field values).
    - Call the modified setter for the loan annual interest rate (no arguments – uses data field values.

  NOTE: The original **getters** for this class will no longer be necessary. All data fields will be accessed via the display instance methods.

- Modify the method that computes the insurance premium, so that it will decide what insurance rate to use.
    - Instead of using a ***constant*** insurance rate (delete the defined constant), use a **multiple alternative if** statement to decide what rate to use, based on the down payment percentage, and store the value into a variable.
        - For down payments of 10% or less, use a rate of 0.52%
        - For down payments over 10%, up to 20%, use the rate from assignment 3 (0.49%).
        - For down payments over 20%, use a rate of 0.47%.

- Convert the two ***static*** display methods from Java Assn 4 to be two ***instance*** methods within the **MortgageLoan** class, as described below.
    - Note that neither of these methods will require parameters, because the data fields can be accessed directly.

- The first instance method, **displayLoanInfo**, will:
  - Display only the loan identifier, the loan amount and loan type.
    - The loan amount should be displayed to 2 decimal places
    - Use an **if/else** statement to display the loan type.
      - Display "conforming" when the loan type is 'C'
      - Otherwise, display "jumbo"
    - All values should line up with each other on the right, as shown below:

```
Loan identifier                              JOHNSO
Loan amount                               450000.00
Loan type                                     jumbo
```

- The second instance method, **calcAndDisplayPaymentInfo**, will:
  - Call each of the calculation instance methods (to compute the monthly taxes, insurance, and loan principle & interest payments) using **`this`** so that the calculations methods calls will use the same object as the **calcAndDisplayPaymentInfo** method.
    - **Save the results** returned from the calls to each calculation method, and then compute the total monthly mortgage payment.
  - Modify the display of the loan length, annual interest, and monthly mortgage payment parts and total, to look like this:

```
  Loan length                            20 years
  Annual interest rate                     3.500%

  Monthly Taxes                            226.04
  Monthly Insurance                        217.00
  Monthly Principle & Interest            2609.82
                                        ---------
  Total Monthly Mortgage Payment          3052.86
```

    - Each of these lines should be indented by 2 spaces
    - The loan length should be displayed in whole years.
    - The annual interest rate should be displayed to 3 decimal places and be followed by a percent sign (%).
    - All dollar figures should be displayed to 2 decimal places, lined up on the decimal points.
    - All values should line up with each other on the right, and also line up with the loan identifier, amount, and type displayed from the **displayLoanInfo** method.

After creating the above two instance methods, be sure to delete the static display methods from the **`MortgagCalculator`** class.

2. Within the original class `MortgageCalculator` class that contains the **main** method:

- Modify the **main** method to:
  - Still display the program description, as in Java Assignment 4.
  - Modify what data is read from the user.
    - Read the home buyer's last name, as in Java Assignment 4.
    - ***Remove*** the code that reads the zip code.
    - Read the home value, as in Java Assignment 4.
    - ***Remove*** the code that reads the annual percentage rate from the user.
    - Read the down payment choice (as shown in the **Sample Inputs** below), as follows:
      - Display a menu to the user, with **letters** as choices.
      - Prompt for the user's choice, and read the letter chosen by the user.

Sample Inputs

```
Please enter the home buyer's last name: Poe
Please enter the home value: 250000

Down Payment Options:
  N - Nothing down
  L - Low down payment
  S - Standard down payment
  H - High down payment
Please enter your choice: S
```

  - Use the new constructor to create TWO objects of the **MortgageLoan** class type.

    Both objects will have the ***same*** buyer's last name, home value, and down payment choice. But they will have different loan lengths.
    - The first **MortgageLoan** object will have a loan length of 20 years.
    - The second **MortgageLoan** object will have a loan length of 30 years.
  - Display a couple of lines to separate the input from the output.
  - Using ***one of*** the objects, call the **displayLoanInfo** instance method to get and display the loan identifier, the loan amount, and loan type.
  - Display a "Loan Option 1:" header. Then call the **calcAndDisplayPaymentInfo** instance method with the first MortgageLoan object.
  - Display a "Loan Option 2:" header. Then call the **calcAndDisplayPaymentInfo** instance method with the second MortgageLoan object.

(see sample output on next page)

WARNING:
As with your previous Java programs, the objects, classes, and methods must be implemented exactly as specified above, or you will lose a significant number of points.

Sample Output

```
Loan identifier                              POEXXX
Loan amount                               200000.00
Loan type                                conforming

Loan Option 1:
  Loan length                              20 years
  Annual interest rate                       3.850%

  Monthly Taxes                              114.48
  Monthly Insurance                          102.00
  Monthly Principle & Interest              1196.21
                                         ----------
  Total Monthly Mortgage Payment            1412.69

Loan Option 2:
  Loan length                              30 years
  Annual interest rate                       4.500%

  Monthly Taxes                              114.48
  Monthly Insurance                          102.00
  Monthly Principle & Interest              1013.37
                                         ----------
  Total Monthly Mortgage Payment            1229.85
```

Coding Standards

The program must also follow the **CS210 Coding Standards** from Content section 6.10.

You must ***include*** the following comments:

- Comments at the ***top of the file, above*** the **main** class, describing what the class does
  - Include **tags** with the author's name (i.e. your full name) and the version of the code (e.g. @version 1.0, Java Assn 3)
- Comments at the ***top of each method***, ***above*** the method header, describing what the method does (***only*** this method – do not refer to actions of any other methods)
  - Include **tags** with names and descriptions of ***each*** parameter and return value.

Delete any default comments supplied by the IDE that you did not use.

Debugging and Testing

Run, test, and debug your Java program, until you confirm that all of the decision control structures work. Be sure to test your program with different inputs to make sure it provides correct results.

Submission

This programming assignment is due by midnight of the date listed in the **Course Assignments by Week**.

Again, you will submit a single **zip** file containing all of the files in your project.

- First export your project from NetBeans:
  - Highlight the project name.
  - Click on **File** from the top menu, and select **Export Project**.
  - Select **To ZIP**
  - Name your export file in the following format:
    `<lastname>Assn<x>.zip`

    For example:
    `SmithAssn5.zip`

  NOTE: Save this zip file to some other directory, not your project directory.

- Then submit your **.zip** file to the **Java Prog Assn 5** assignment submission folder (located under the **Assignments/Dropbox** tab in the online course).

  - Warning: Only NetBeans export files will be accepted.
    Do not use any other kind of archive or zip utility.

Grading

Your program will be graded using the **rubric** that is linked on the same assignment page from which this program requirements file was downloaded.

WARNING:
Programs submitted more than 5 days past the due date will ***not*** *be accepted,*
and will receive a grade of 0.