CS10 Python Programming Homework 5
20 points (OOP)

1. You must turn in your program listing and output for each program set. Each program set must have your student name, student ID, and program set number/description. Late homework will not be accepted for whatever reasons you may have.

*********_**for this homework, you are to submit your Program sets to Canvas**_ under Homework 5 link*********

   a. Name your files : HW5_PS1_ lastname_firstinitial.py for Program Set 1 and so on. PS means program set. Example if your name is Joe Smith then the filename will be HW5_PS1_smith_j.py
   b. You must submit your homework by the deadline at the specified upload link on Canvas under homework 5. Late Homework submission will not be accepted. Homework submitted via email attachment, comment in Canvas, Canvas message, or by any other method is not accepted and will be given a zero for no submission.
   c. if you do not follow instructions on file naming provided in this section you will receive a zero for the question you did not correctly name the file.
   d. It is your responsibility to check if your homework is properly submitted to Canvas.

2. Please format your output properly, for example all dollar amounts should be printed with 2 decimal places *when specified*. Make sure that your output values are correct (check the calculations).

3. Use only the 'tools' in the topics we covered from lesson 1 to lesson 68 only.

4. Each student is expected to do their own work. **IF IDENTICAL PROGRAMS ARE SUBMITTED, EACH IDENTICAL PROGRAM WILL RECEIVE A SCORE OF ZERO.**

Grading:

Each program set must run correctly both syntactically, logically, and display the correct output exactly as specified. **If the program set does not run and does not display the output exactly as specified, shown in my sample test run, a zero will be given**. If the program executes properly with proper syntax, logic, and displays the correct output with proper formatting as specified in the program set, you will receive the full points for that question. Then points will be deducted for not having proper:

a. Comments (1 pt. deducted for each occurrence)
   - Your name, description at the beginning of each program set. Short description of the what each section of your codes does.

b. Consistency/Readability (2 pts deducted for each occurrence)
   - Spacing(separate each section of codes with a blank line) but separate each function with 2 blank lines
   - Indentation
   - Style (proper naming of variables no a, b, c – use descriptive and mnemonics)
   - include **docstrings** for every function

c. Required elements (2 pts deducted for each occurrence)
   - Use only 'tools' in the topics that have been covered in class.
   - proper formatting for output when specified

d. Output
   - if no output(test runs) are provided for the uploaded file, a zero will be given for that program set.
   - Test runs must to be displayed at the end of the program listing(codes), if it not displayed at the end of the program listing it is equivalent to no output.
   - must use test run when provided in the Program set question. The minimum test cases you must provide is 5 or more. If you provide less than 5 test cases per Program Set then that program set will receive a zero grade.

Points will be deducted from grading items a. to d. above until your Program Set reaches zero points.

Program set 1 – OOP (20 points)

Write a program to calculate and display the loan for buying a car.

1. Create a class call Loan.
   Data fields in the Loan class include:
   1. Annual Interest Rate (Float)
   2. Number of years of loan (int)
   3. Loan Amount (Float)
   4. Borrower's Name (string)

2. Create the initializer or constructor for the class with the above data fields. Make the data fields **private**.

3. Create accessors (getter) for all the data fields(there should be 4 getters).

4. Create mutators (setters) for all the data fields(there should be 4 setters.

5. Create a class method - getMonthlyPayment where

   *monthlyPayment = loanAmount * monthlyInterestRate / (1 - (1 / (1 + monthlyInterestRate) ** (numberOfYears * 12)))*

   note: that the monthlyInterestRate = annualInterestRate / 1200 (this line of code must be before the monthlyPayment calculation formula)

6. Create a class method - getTotalPayment where

   *totalPayment = getMonthlyPayment() * numberOfYears * 12*

7. Write a test program (main function) to allow the user to enter the following:
   1. Annual Interest Rate
   2. Number of Years of Loan
   3. Loan Amount
   4. Borrower's Name
   5. Allow the user to change the loan amount and reprint the new loan information.

   ****use same object instantiated to change the loan, not allowed to create a new object.**

Note : you are not allowed to use the break or continue statement for this program set 1. Using a break or continue statement will result in a zero score for this program set.

Sample Test Run (use this sample test run data and provide 4 more test runs using your own data). user input in red

Test run 1

Enter yearly interest rate: 2.5
Enter number of years as an integer: 5
Enter loan amount: 1000.00
Enter a borrower's name: John Jones
The loan is for John Jones
The monthly payment is 17.75
The total payment is 1,064.84

Do you want to change the loan amount? Y for yes or enter to quit y
Enter new loan amount 5000
The loan is for John Jones
The monthly payment is 88.74
The total payment is 5,324.21

Do you want to change the loan amount? Y for yes or enter to quit Enter

>>>

Test run 2, 3, 4 and 5 using your own input data

Hints:

Use the circle program with ***private*** data fields(members) to follow along. The circle's constructor or initializer has only one data member radius but the loan has 4, so you must put all 4 data members in your initializer.

```
def __init__(self, annualInterestRate = 2.5,
             numberOfYears = 1, loanAmount = 1000, borrower = " "):
```

you should also have 4 accessors(getters) and also 4 mutators(setters) for each of the data member.

You also need to write methods getMonthlyPayment() and getTotalPayment().