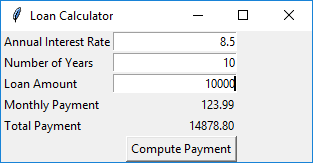
**CIS 221 PA6 - Loan Calculator Enhancement**

**Introduction**

You are a systems analyst working with a company that provides loans to customers. Your manager has asked you to enhance and correct an existing **Loan Calculator** program, which is designed to calculate monthly and total payments given the loan amount, the annual interest rate, and the duration of the loan. Although the current version of the program (hereby termed the **“As Is”** version) has some functionality, there are several missing pieces, and the user would like to see significant improvements. Also, the program has some bugs that need to be fixed. After a **requirements elicitation** session (see <https://en.wikipedia.org/wiki/Requirements_elicitation>), you have uncovered the following facts about the **As Is** version, and you are prepared to work with this existing program and create the enhanced version (hereby termed the **To Be** version).

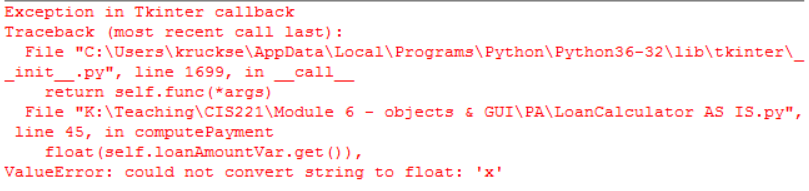
Note: the as-is program is **Listing 9.11 LoanCalculator.py**, which is described in section 9.7 of the textbook. The existing code is available in the PA6 Assignment – download the code for a starting point.

The **requirements** are discussed below.

**The *As Is* System**

In its current state, the program takes user input in the form of an annual interest rate, number of years, and total loan amount. It displays the calculated monthly payment, and the calculated total payment. This image shows an example.

This is fine as far as it goes. But it has a glitch. For example, if the user fails to enter proper numeric values, the program aborts with an ugly message like this:

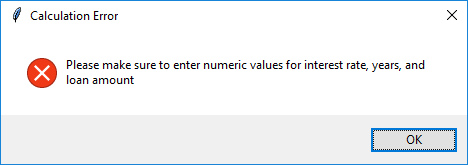


Users are very annoyed when they see this, and they have to restart the program in order to continue their work.

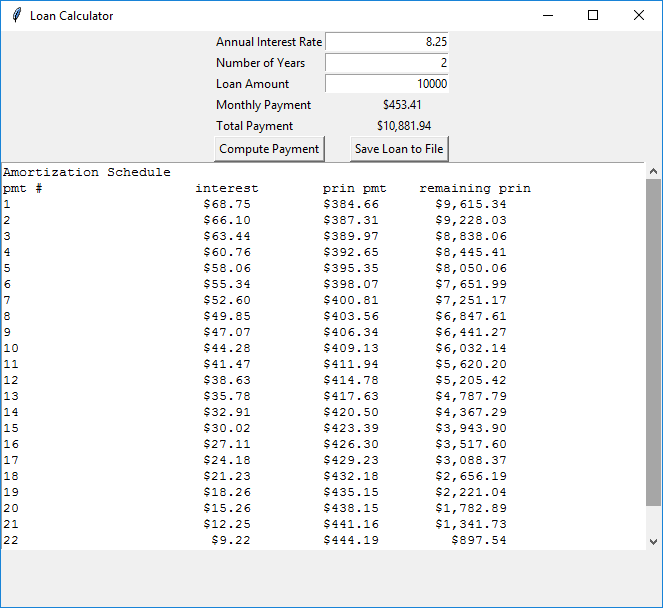
Also, the program is incomplete. There is more functionality that the users want, which is described below.

**The *To Be* System**

The users would like additional functionality to the program, and also some corrections.

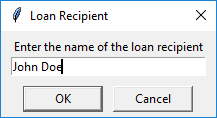
First, your program should handle exceptions more gracefully. Rather than the ugly situation described above, an error message should appear telling the user what correction to make. For example, if the user clicks the calculate button without having proper input data entered, you should display a dialog like this:

Second, the program should include an amortization schedule. This should be displayed in a Text widget. This screen shows an example of what would appear given user input:

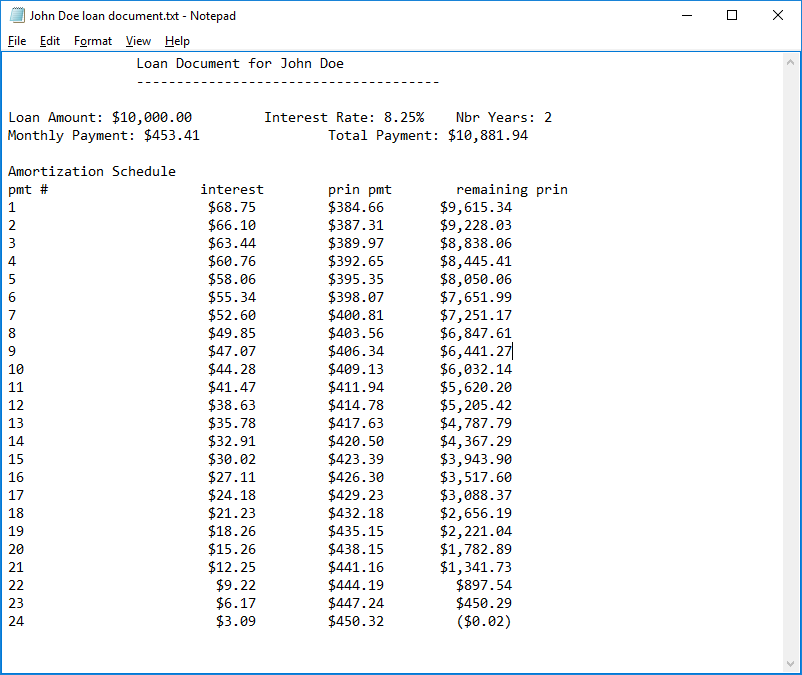


Note: for hints on how to calculate the amortization schedule, see problem 5.24 on page 162 of your textbook. Instead of printing as shown in the algorithm on page 162, you will either concatenate to a string or insert into the Text widget.

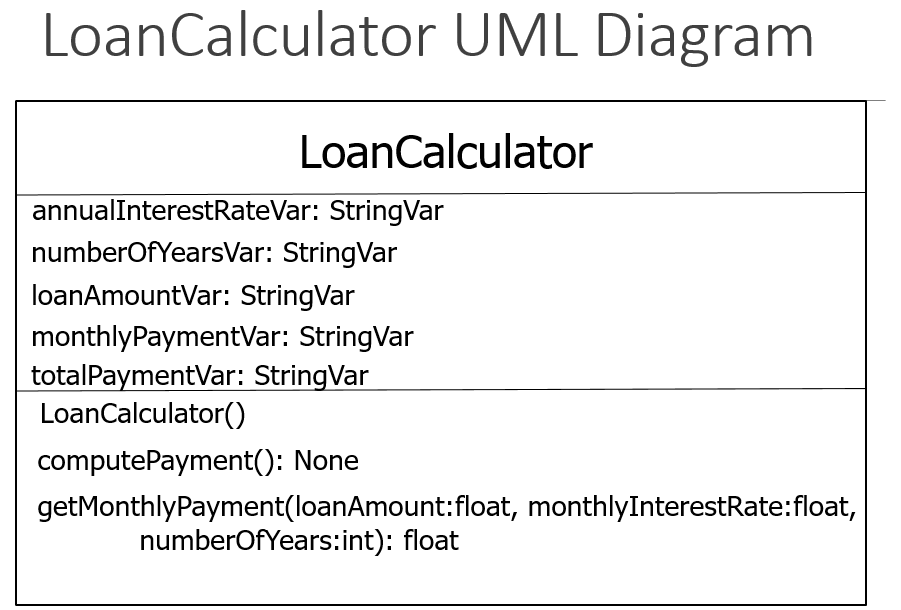
Note also that the interest, monthly payment, and total payment values are shown in currency format. In order to make these amounts show up with correct currency format, make use of Python’s locale module: <https://docs.python.org/2/library/locale.html>. Use Google to find out how you can do this.

Finally, we want to be able to save the loan report to a file (note the extra button in the figure shown above). If the user clicks the Save button, prompt the user to enter the loan recipient’s name using a dialog, as shown below:

When the user enters the name, write to a file named using the given name. The new file will include data shown as below:



In addition to the proper functioning program, please also include a **modified UML diagram**. The original UML diagram is shown below and is discussed in Ch 9 of the text. The modified UML diagram will include any additional methods you wrote to accomplish your programming task.



**Final Deliverables:**

Zip together the modified Python program and the modified UML diagram; upload the zip file to Canvas. Test the program to make sure what you’re turning in works correctly (download what you uploaded on a different computer and run the program).