

ENGR 102 – Spring 2021
Sections 501 and 502
Dr. Fullerton
Lab Assignment #9b

Deliverables:

There are three deliverables for this individual assignment. Please submit the files to the Submission Box labeled “Lab Assignment 9b Individual Activities”. The files should be named as listed below:

- Lab9b_Act1.py
- Lab9b_Act2.py
- Lab9b_Act3.py

These activities are meant to help give you practice reading and writing files, as well as processing larger amounts of data, which is one of the common tasks that computer programs are written for.

Activity #1:

Create a text file named Celsius.txt using a word processor. The file should contain a list of 25 temperature values in degrees Celsius (one per line) in the form of floating-point numbers. Write a Python program `Lab9b_Act1.py` that will read the temperature values from the file Celsius.txt and will create a file Fahrenheit.txt that contains the same temperature values (one per line, in the same order) in degrees Fahrenheit.

Activity #2:

Note: In Lab Assignment #9 Team, we practiced using Python to write data to a file with each data item separated from its neighbors using a comma. We then practiced using Python to read the data in that file and then do something useful with it. This activity provides even more practice with what we will now call a CSV (Comma Separated Value) file.

One of the most common ways that data can be stored to be later loaded in a spreadsheet or other similar table is with a CSV (Comma Separated Value) file. These files are often given a .csv extension. A csv file is a way of representing a table in a file. Each line represents a row of the table, and the cells in each column are separated by commas. CSV files can usually be read into spreadsheet programs (such as Excel), and most spreadsheets can output their data in a CSV format (sometimes called “comma delimited” format). You are going to practice writing and reading CSV data directly.

Write a program `Lab9b_Act2.py` that will save, to a file, a list of amortized values for a loan. Specifically:

- a. Ask the user for a file name for saving the data.
- b. Ask the user for the amount of the loan (P), the number of months (N) over which the loan will be repaid, and the annual interest rate (i). (Note that i should be a decimal number, not a percentage: 0.025 not 2.5%)
- c. Calculate the monthly payment (M) as shown below. Note: $J = i/12$

$$M = P \left[\frac{J}{1 - (1 + J)^{-N}} \right]$$

- d. Check your calculations: For $P = \$100000$, $N = 60$, $i = 0.025$ the monthly payment should be $M = \$1774.74$

- e. For each month, calculate the accrued interest by multiplying the beginning balance for that month by J . Then calculate the ending balance for that month by subtracting the payment from the beginning balance and adding the accrued interest to that figure.
- f. For each month, write to the output file the month number, the total amount of interest accrued so far, and the amount remaining on the loan, separated by commas.
 - i. Start with month 0, when there is no payment and no interest, with month 1 being the first payment and first interest accumulation
 - ii. The loan will eventually be paid off, so write out values until the balance is \$0.00.
- g. Be sure that you write out column headers for the table. It is permissible to include the header strings as comma-separated data on the first line of the file.

Note: If you write your .csv file correctly, you should be able to open it in a spreadsheet program that can read .csv files. You can also check the values themselves using the PMT() function in Excel. (See the posted file `LoanExample.xlsx`.)

Activity #3:

Practice opening and reading ~.csv files using Python. Write a Python code `Lab9b_Act3.py` that opens the output file from Activity #2 above, reads the values, and prints a nicely formatted table to the console.