# CSE/ISE 337: Scripting Languages Stony Brook University Programming Assignment #5 Spring 2025

**Assignment Due: May 12, 2025 11:59 PM** 

# **Learning Outcomes**

- After completion of this programming project, you should be able to:
  - develop a database system with Python using sqlite3 o GUI application with Python using Tkinter
- The code given with the assignment is not working. It's a template that needs to be extended to develop Apps for each problem.
- Learn Basic Web Scraping

# Problem 1 [20 points] (Utsha)

In this exercise, you will enhance the Movie List program by improving its delete command and by adding a min command that lets the user view movies with run times that are less than a specific number of minutes.

# Open and test the program:

- 1. Review the starter code for the problem, problem1.tar
- 2. Review the code and note how the ui module uses the db module and the Movie class from the Objects module. Then, run the program.

# Improving the del command:

- 3. In the db module, add a get\_movie() function that gets a Movie object for the specified movie ID.
- 4. In the ui module, modify the delete\_movie() function so it gets a Movie object for the specified ID and asks whether you are sure you want to delete the movie as shown above. This code should only delete the movie if the user enters "y" to confirm the operation.

5. In the db.py, modify the add\_movie() function such that no duplicate movies can be made. A movie is a duplicate if it has the same name, year, minutes, and category ID.

### Add the minutes command

- 6. In the db module, add a get\_movies\_by\_minutes() function that gets a list of Movie objects that have a running time that's less than the number of minutes passed to it as an argument.
- 7. In the ui module, add a display\_movies\_by\_minutes() function that calls the get\_int() function to get the maximum number of minutes from the user and displays all selected movies. This should sort the movies by minutes in descending order.
- 8. Modify the main() function and the display\_menu() function so they provide for the min command. You need to submit code as Problem1.zip (Attached "Movies.db" Database for reference. As this assignment modifies the database while testing your code. **So don't include the database while submitting**. Testing is done on the Original Dataset.

```
Command: del
Movie ID: 14
Are you sure you want to delete 'Juno'? (y/n): y
'Juno' was deleted from database.
Command: min
Maximum number of minutes: 100
MOVIES - LESS THAN 100 MINUTES
ID Name
                                            Year
                                                   Mins
                                                         Category
    Ice Age
                                                   81
                                            2002
                                                         Animation
    Toy Story
                                            1995
                                                   81
                                                         Animation
    Spirit: Stallion of the Cimarron
                                            2002
                                                   83
                                                         Animation
    The Lion King
                                            1994
                                                   88
                                                         Animation
    Aladdin
                                                   90
                                            1992
                                                         Animation
    Shrek
                                            2001
                                                   90
                                                         Animation
    A Quiet Place
                                            2018
                                                   90
                                                         Horror
    Monty Python and the Holy Grail
                                            1975
                                                   91
                                                         Comedy
    Monty Python Life of Brian
                                            1979
                                                   94
                                                         Comedy
11 Eastern Promises
                                            2007
                                                   96
                                                         Comedy
```

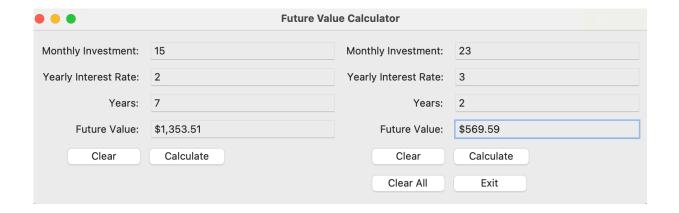
- Homework 5 - General Purpose Computing

# **Point Distribution:**

- Working commands :
  - get\_movie(): 4 points
  - delete\_movie(): 4 points
  - add\_movie(): 4 points
  - get\_movies\_by\_minutes(): 4 points
  - display\_movies\_by\_minutes(): 4 points

# Problem 2 [20 points] (Utsha)

In this exercise, you'll create a Future Value program that allows you to make two side-by-side calculations in the same window. When you're done, the GUI should look like this:



# Requirements:

- Inputs: Each calculation panel should include the following inputs:
  - o Monthly Investment: The fixed amount invested each month
  - Yearly Interest Rate: The annual interest rate.
  - Years: The total number of years the investment will grow.
- Outputs: Each calculation panel should display:
  - Future Value: The total value of the investment at the end of the given period, including compounded interest.
- Buttons:
  - Each panel must have Clear and Calculate buttons:
    - Clear: Resets all fields in the respective panel.
    - Calculate: Computes the future value based on the provided inputs for that panel.
  - A separate Exit button to close the application.
  - A separate Clear All button that clears all input values.
- Functionality:
  - o Ensure that the computed future value is displayed in a read-only field to prevent user modification.
  - o Input values for each panel should not affect the calculations in the another panel.
- Error Handling:
  - Display appropriate error messages if the user provides invalid inputs

(e.g., non-numeric values or negative numbers).

# Review the starter code (Problem2.tar) for the application. You will submit your final code as Problem2.zi

## **Point Distribution:**

- The student's implementation sufficiently resembles the sample shown above. Does not have to be exact: 5 points

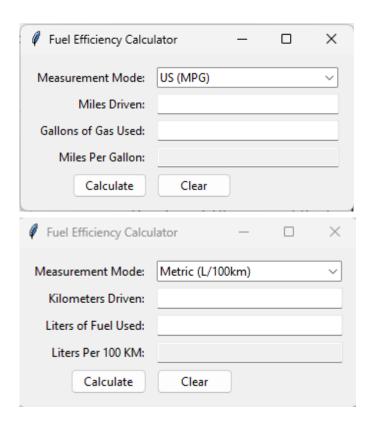
Clear button : 3 pointsClear All button : 3 pointsCalculate button : 3 points

- Exit button: 3 points

- The side-by-side calculators work separately : 3 points

# Problem 3 [25] (Kushal)

Develop a GUI version of the Fuel Efficiency program. When you're done, the GUI should look like one of the following based on the Measurement Mode:



The program should compute the Miles Per Gallon (MPG) or Liters

Per 100 KM(L/100km) using the formulas:

$$MPG = \frac{Miles\ Driven}{Gallons\ of\ Gas\ Used}$$

or

$$L/100 \text{ km} = \left(\frac{\text{Liters of Fuel Used}}{\text{Kilometers Driven}}\right) \times 100$$

# **Requirements:**

- Inputs:
  - A distance text field to accept "Miles Driven" or "Kilometers Driven" as input.
  - O A fuel consumption text field to accept "Gallons of Gas Used" or "Liters of Fuel Used" as input.
  - A toggle/drop-down mode field to select "US(MPG)" or "Metric(L/100km)".
- Outputs:
  - O A read-only result text field that displays either (1) the computed Miles Per Gallon rounded to two decimal places, or (2) the computed Liters Per 100 KM rounded to two decimal places, depending on US mode or Metric mode, respectively.
- Button:
  - A button labeled "Calculate" which computes the MPG or L/100km based on the provided inputs when clicked.
  - O A button labeled "Clear" which clears all input and output fields and resets them to default (default for Mode will be "US(MPG)").
- Functionality:

Initial Start of Program + On Changing Modes:

- O The Mode should be set to a default value of "US(MPG)", with input and output labels reflecting US mode appropriately, with distance and fuel inputs being "Miles Driven" and "Gallons of Gas Used" and output being "Miles Per Gallon".
- When the user toggles/changes the mode (to "Metric(L/100km)" for example), the input and output labels should update to reflect the

new (Metric) Mode, with distance and fuel inputs being "Kilometers Driven" and "Liters of Fuel Used" and output being "Liters Per 100 KM".

# When the "Calculate" button is clicked:

- The program should read the values entered for the distance and fuel consumption input fields.
- It should calculate the MPG or L/100km and display the result in the read-only output field.
- Error Handling:
  - Ensure the program gracefully handles invalid or empty input values (e.g., non-numeric input or division by zero).

Review the starter code for the problem, Problem3.tar. You will submit the final code as Problem3.zip.

# Problem 4 [25 points] (Web Scraping: Extracting Headlines) (Kushal)

Develop a Python script to scrape the **top headlines** from the BBC News website (https://www.bbc.com).

# **Requirements:**

## Inputs:

- Use Python's requests library to fetch the content of the webpage.
- Use BeautifulSoup to parse the webpage's HTML.

# **Outputs:**

- Extract and display the following for each headline:
  - The **headline text**.
  - The corresponding link (absolute URL).
  - The absolute date (metadata for date last updated) [Date format to display as (with description for each field in parentheses) = Month(string

with first letter of abbreviated month name capital) day-of-month(1-31) year(YYYY)].

- O If the date last updated is displayed relatively (as within minutes/hours/days), use the current date to figure out the absolute date the headline was last updated.
- O Otherwise if the date last updated is already displayed absolutely [following the website's given format = day-of-month(1-31) Month(string with first letter of abbreviated month name capital) year(YYYY)], simply parse each of these fields and display them using the format defined in this bullet's parent bullet for **absolute date**.
- The tag (metadata for tag or genre).
- Display the extracted headlines in a clean, numbered format.

# **Functionality:**

Your program should:

- 1. Fetch the BBC homepage (https://www.bbc.com).
- 2. Extract and display a list of top headlines and their corresponding links, dates last updated, and tags.
- 3. Ensure that links are displayed as complete URLs (e.g., convert relative links to absolute URLs).
- 4. Ensure that the dates last updated are displayed as absolute dates according to the defined format (e.g, convert relative dates to absolute dates).

# **Submission Requirements:**

- 1. Submit the completed scraper\_helper.py file with the missing functions implemented.
- 2. Submit the completed scrape\_headlines.py file with the logic to fetch, extract, and display headlines.

# **Error Handling:**

- Gracefully handle invalid webpage requests (e.g., server errors or connection issues).
- Ensure the program doesn't crash if no headlines are found.

Review the starter code for this problem, Problem4\_Starter.zip. You will submit your solution as Problem4.zip.