GENG0033 Programming and Coursework

Laboratory Work 1 – Using Python (Functions)

Module Code:	GENG0033		
Module Title:	Programming and Coursework		
Module Leader:	Hasmath Farhana Thariq Ahmed		
Assessment Type:	Individual	**Weightage:**	12%
Submission Due Date:	09-January-2026 (3 PM Malaysia Time)		
Method of Submission:	Blackboard		
Assessment Tier	Tier 1: No GenAI use (for entire work) * Form: GenAI Declaration Form.docx		

This assessment relates to the following Module Learning Outcomes:

C. Subject Specific Practical Skills	C2. Demonstrate knowledge and understanding of digital literacies and use of common digital tools. C3. Able to apply problem-solving techniques to familiar and unfamiliar problems.

* This assessment is classified under Tier 1 (No GenAI use). Students must not use any Generative AI tools to generate code, text, or solutions for this assignment. **As Tier 1 applies, students do not need to submit the GenAI Declaration Form.**

Coursework Brief:

1. Task

You are required to develop an **expense tracker** using Python (version 3.x or above). This console-based program will help users monitor and manage their expenses across different categories. The system should allow users to choose from several tasks, such as:

- **Adding**, **editing**, and **deleting** entries for expense categories (e.g., food & drinks, commute, utilities, entertainment, shopping).
- **Tracking details for each expense**, including:
  - Expense ID (unique ID), expense date, category, amount, and type(debit/credit).
  - Additional notes or details (optional field for the user).
- **Summarising** the expenses, by showing the total amount spent daily, weekly, and monthly. It should also show which categories cause the highest spending based on the user's overall expenses.
- **Searching and filtering** the expense list based on category (e.g., food & drinks, commute, utilities, entertainment, shopping), date range (start date to end date), or amount spent (minimum to maximum). Only the expenses that meet the specified conditions should be displayed.

These functions are suggestions; feel free to expand upon them as needed. You may store data in a list, text/binary file, or a CSV file. The **pandas** library is recommended for data management tasks, such as reading and writing files (Example reference: https://realpython.com/pandas-read-write-files/#write-a-csv-file).

System Requirements:

To build a comprehensive program, please include the following elements:

- Selection Statements: Allow the user to choose which task to perform from the menu (e.g., add expense, edit expense, delete expense, view summaries, search/filter expenses, or exit the program).
- Repetitive Statements: Use loops so the program continues running until the user chooses to exit. This allows repeated data entry and repeated viewing of summaries or filtered results.
- Functions: Organise the code into functions (such as adding, editing, deleting, summarising, searching, loading, and saving expenses) to make the program modular, readable, and easy to manage.

Since you may not have covered Graphical User Interface (GUI) development, no interface is required for this assignment. Focus on creating a console-based application. Feel free to expand upon these requirements based on your ideas and preferences.

Additional Requirements:

- Ensure that your code is well-organized, readable, and avoids unnecessary redundancy.
- Demonstrate creativity by adding unique features or trying new approaches. This is an opportunity to showcase your programming and problem-solving skills.

2. Demonstration video

To support your program, record a demonstration video of your program for a maximum of 10 minutes. Run the program, key in input from the user and show the output. Imagine that you are promoting this system to a potential buyer, that's how you should explain your program. You may use Zoom or Teams to record your demonstration video.

3. Marking scheme

Criteria	Marks
Meeting system requirements	40
The application A code standard, sufficient comments, meaningful variables, functions	30
The initiative, innovation, creativity and effort, problem-solving	20
Demonstration video (10 minutes)	10
Total marks	100

You are expected to show how you developed your understanding and to demonstrate scientific curiosity and reflection in ensuring you fully understand your submission.

<40%	You have not succeeded in coding with Python programming
40-49%	You have got some of the Python code workings
50 -59%	You appear to have got your Python codes working but your code standard (comments, meaningful variables and functions) and report are too brief and unconvincing
60-69%	You have got the program working and some sensible, well-written notes in the report showing a good understanding of the given tasks.
70-80%	Your report demonstrates an excellent understanding of, and reflection on, the given tasks, and you have, where appropriate, completed beyond those required tasks.

Assessment rubric is also attached (at the end of the document) for reference.

Any work submitted after the deadline's time will be subject to the standard University late penalties unless an extension has been granted, in writing by the Senior Tutor, in advance of the deadline. Details on the University's late penalties can be found here:

- https://www.southampton.ac.uk/~assets/doc/quality-handbook/Late%20Submission.pdf

Submission Instructions:

You are required to submit the following items for this assessment:

1. Python File (.py)

- Submit **ONE** Python file containing your full Expense Tracker program.
- All functions (add, edit, delete, summary, filter, menu, load/save, etc.) must be included in this single file.
- The program must run directly from this file.

2. External Data Files (if used)

- If your program uses any external files (e.g., CSV or text files), you must upload these files separately.
- Ensure the filenames you use in your code match the files you upload.

3. Demonstration Video Link

- Record a demonstration video (maximum 10 minutes) showing your program running, with user inputs and outputs.
- Upload the video to your OneDrive.
- Set the link permissions to "View Only".
- Share the link with the following emails:
  H.F.Thariq-Ahmed@soton.ac.uk
  M.N.Zamri@soton.ac.uk
- Ensure the link is accessible (test it before submitting).

4. README File (optional but recommended)

- You may include a short **README** file to explain:
  - how to run your program
  - any special instructions
  - any files required (e.g., sample CSV)
- This helps the marker run your program smoothly, however, this file is optional and missing it will not affect your grading.

Important Note About File Uploads!

Please **do NOT zip your files**. Blackboard cannot preview ZIP folders.
Upload **each file separately** in the submission area, including:

1. your Python file **(Mandatory)**
2. any external data files (if used)
3. and your demonstration video link **(Mandatory)**
4. README file (optional)

University of Southampton MALAYSIA

Assessment Rubric*

Grading Criteria	Wtg	Exceptional 4	Acceptable 3	Amateur 2	Unsatisfactory 1	Incomplete 0	Mark (Wtg x Score)
Functionality	**10.0**	All required functionality works flawlessly, and the program handles edge cases effectively.	Most required functionality works without issues, but some edge cases may not be handled optimally.	Basic functionality is present, but there are notable issues and limitations.	Only a small portion of the required functionality works, and many issues exist.	The program does not function as expected, and most functionality is missing or broken.	**/40**
Code Quality	**7.5**	The code is exceptionally well-organized, and efficient, and adheres to best practices and coding standards.	The code is well-organized, efficient, and follows good coding practices.	The code is mostly organized and efficient but may contain minor issues.	The code is disorganized, inefficient, or contains significant issues.	The code is chaotic, inefficient, and riddled with issues.	**/30**
Creativity & Innovation	**5.0**	Introduces innovative solutions or creative elements that significantly enhance the project.	Includes some creative and innovative elements that add value to the project.	Lacks significant creativity or innovation but meets the basic requirements.	Lacks creativity and innovation, leading to a somewhat bland result.	No signs of creativity or innovation; the project is a basic, unoriginal implementation.	**/20**
Demonstration Video	**2.5**	Includes more than required in the presentation. Presents ideas in exceptional order. Organises time very well. The audience managed to really understand the program through the demo and liked the program	Includes everything required in the presentation. Presents ideas in order that makes sense. Organizes time well. No part of the presentation is rushed, too short or too long. The audience managed to understand the program through the demo and liked the program	Includes almost everything required in the presentation. Tries to present ideas in order but it doesn't always make sense. Presents for the right length of time, but some parts may be too long or too short. The audience managed to fairly understand the program through the demo	Does not include everything required in the presentation. Presents ideas in an order that does not make sense. Does not plan the timing of the presentation well. It is too short or too long. The audience did not manage to understand the program through the demo	No video attached	**/10**
						Total Marks	**/100**

***** The rubric outlines how marks are awarded. Attempting the task alone is not enough for full marks; you must meet the expected level of performance in each criterion.