**CMPU 2016 Object Oriented Programming**

TU857-2

2024-25, Semester 1: Python with Sunder Ali Khowaja

[SunderAli.Khowaja@tudublin.ie](mailto:SunderAli.Khowaja@tudublin.ie)

**Lab 8:** **Composition and Dependency Management**

Welcome to Week 8 of the Object-Oriented Programming course! In this lab, you will continue your exploration of composition and dependency management in object-oriented design. These principles play a crucial role in creating modular and maintainable code by structuring classes effectively and managing their relationships. You'll work with the existing game code from previous weeks and enhance it by incorporating these essential techniques, including the integration of a logger class.

**Lab Duration**: 2 hours.

**Lab Objectives:**

* Understand the concept of composition and its significance in promoting code reusability and maintainability.
* Learn to manage dependencies between classes to reduce tight coupling and enhance flexibility.
* Explore how composition and effective dependency management contribute to better software design.
* Apply composition principles to the existing game code and utilize a Logger class to facilitate logging.

**Instructions:**

1. Open the Python code file from the previous Week 5 (the version without operator overloading).
2. Review the existing codebase, including the Character, Suspect, Witness, and Game classes.
3. Implement composition by creating a new Loggable class to handle game logs. Create methods for logging messages and retrieving logs.
   * Create a variable that can hold the collection of logs, called logs. Think about if this variable should be public, private or protected and offer getter/setters accordingly via decorators.
   * Create a method called log that accepts a message of type string and adds it to the logs variable.
4. Refactor the code to integrate the Loggable class into the game by logging relevant events during the game's execution.
   * Set up an instance variable of type Loggable in the Game class’s \_\_init\_\_ method.
   * In the various methods within the Game class log events.
5. In the main function read out your logs:

if \_\_name\_\_ == "\_\_main\_\_":  
 game = Game()  
 game.run()  
  
 *# Using the logger*  
print("\nGame Logs:")  
 for log in game.log.logs:  
 print(log)

1. Challenge Task (optional – for bonus points): Enhance the game by logging interactions between characters, allowing players to review the history of interactions during game play.
2. Answer the quiz questions and upload your code. This lab is marked. A solution is available from tomorrow on. The solution code will be discussed in the next lecture. Your solution will likely differ from mine. If you fulfil the task description this is not a problem and is to be expected. If you have questions after reviewing the solution, please contact me asap via email.