**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 3: Encapsulation and Data Handling in Mystery Adventure Game**

**Lab Objective:** In this lab, you will practice implementing the principles of encapsulation and data handling within the mystery adventure game. You will create a structured class hierarchy, encapsulate data using access control, and utilize methods to interact with and modify the game's internal state.

**Lab Duration:** 2 hours.

**Lab Instructions**: Complete the following tasks, then answer the lab quiz which requires you to upload your Python file.

**Lab Tasks:**

1. Review and Understand the Existing Code:

* Open the updated Python code from Week 1 (either your or mine solution, it is your choice) and ensure you understand its structure and key methods.
* Use the Pycharm Debugger to step through the programme and understand the flow of the code.

1. Create a Structured Class Hierarchy:

* Create a new class CrimeScene that encapsulates the crime scene details and methods for interaction:
  + The class should instantiate location (a Python string type), clues (a Python list type) and investigated (a Python boolean type) in its \_\_init\_\_ method.
  + HINT: for more on how to use the Python list data type, see your Python cheat sheet.
  + Only the location is public, the other attributes should be private using the double underscore notation.
  + Your \_\_init\_\_ method sets up your attributes. Your \_\_init\_\_ method should also accept an argument in addition to self: the location of the crime scene that you then assign to the self variable of the same name.
  + The class should contain an add\_clue(self, clue) method that adds a clue to its clues attribute, a list.
  + The class should contain a review\_clues method that returns the private clues attribute. You may include checks here to safeguard that not just anyone at any time can access see all the clues that have been gathered already.

1. Implement Encapsulation with Access Control:

* Implement getter and setter methods to access and modify private attributes using the decorator notation in Python.
  + **Think first! Should all your private variables really be readable and/or writable?**

1. Enhance Crime Scene Interaction:

* Modify the game's interaction logic to involve the new CrimeScene class. For this, you set up an object of CrimeScene in the \_\_init\_\_ method of the Game class. Remember that you need to provide the location variable to the creation of your CrimeScene class.
* Allow players to interact with the crime scene, examine clues, and gather information using the methods you've created, via the CrimeScene class. For this, review your code. Where could you add clues using the add\_clues method that CrimeScene provides.

1. Enhance the game play:

* Add another option to your interactive menu:
* The “i” for investigating a crime scene should check if the scene has already been investigated and print out that the scene has already been investigated if this is the case. If the scene has not been investigated yet, be creative. What are we revealing in our investigation?

Press 'q' to quit, 'c' to continue, 'i' to investigate the crime scene or 'r' to review your clues:

This option should also set the investigated boolean attribute in your new CrimeScene class to true.

Below you find an example script output of a game being played.

1. Test Your Modifications:

* Run the modified code and ensure that encapsulation is effectively maintained and data handling is controlled through methods.

1. 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.

**Example Script of the Game being Played:**

Welcome to 'The Poirot Mystery'.

You are about to embark on a thrilling adventure as a detective.

Your expertise is needed to solve a complex case and unveil the truth.

Press 'q' to quit or 's' to start: s

Enter your detective's name: Ali

Welcome, Detective Ali!

You find yourself in the opulent drawing room of a grand mansion.

As the famous detective, you're here to solve the mysterious case of...

'The Missing Diamond Necklace'.

Put your detective skills to the test and unveil the truth!

You notice two doors in the room. Enter '1' to investigate the left door, '2' for the right door: 1

As you approach the left door, you hear a faint whisper... The plot thickens!

Press 'q' to quit, 'c' to continue, 'i' to investigate the crime scene or 'r' to review your clues. r

Your clues so far:

['faint whisper']

Press 'q' to quit, 'c' to continue, 'i' to investigate the crime scene or 'r' to review your clues. c

You continue your investigation, determined to solve the mystery...

Press 'q' to quit, 'c' to continue, 'i' to investigate the crime scene or 'r' to review your clues. i

You step into the dimly lit crime scene. Broken glass lies near the window, and a table is overturned.

There's a distinct smell of perfume lingering in the air. The mystery deepens.

Press 'q' to quit, 'c' to continue 'r' to review your clues. r

Your clues so far:

['faint whisper', 'broken glass near window', 'an overturned table at crime scene', 'smell of perfume']

Press 'q' to quit, 'c' to continue 'r' to review your clues. q

Process finished with exit code 0