**LEE KONG CHIAN FACULTY OF ENGINEERING AND SCIENCE**

**UNIVERSITI TUNKU ABDUL RAHMAN**

**UEEN 1043 Object-Oriented Concepts and Programming Techniques**

**Practical Assessment 2: Overloading, Inheritance and Polymorphism**

**Assessment**

You are given a CSV file named ***oop\_uni\_students.csv*** containing student details.  
Each row in the CSV file represents a student's information with the following columns:

Student ID,Level,Credits Completed,GPA,Thesis Title,Publications,Study Duration

1. Implement load\_data(): Load students (undergraduate and master levels) from the CSV file.
2. A base class Person is provided below. Create a Student (undergraduate) class and a MasterStudent class that inherit from Person.
   1. Student class should include Credits Completed and GPA as its private attributes. The graduation criteria for an undergraduate student: Credits Completed >= 120 and GPA >= 2.0.
   2. MasterStudent class should include Thesis Title, Publications, and Study Duration as its private attributes. The graduation criteria for a master student: >=1 Publications and >=2 years of Study Duration.
3. Implement get\_students\_ready\_for\_graduation(): Displays students (undergraduate and master levels) who qualify for graduation. This function should call graduation\_readiness() of the Student and MasterStudent class demonstrating the concept of polymorphism.
4. Implement show\_probation\_list(): Displays undergraduate students with GPA <2.0.
5. Implement get\_student\_by\_id(student\_id): Retrieves details of a specific student (undergraduate or master), return message to indicate invalid id. This function should call get\_sumary() of the Student or MasterStudent class demonstrating the concept of polymorphism.

Write a Python program with clear and concise comments to complete the following tasks. You should be able to run the provided examples (though not limited to them) after completing the implementation. Keep in mind to apply the concept of inheritance and polymorphism in OOP. Reuse the base class implementation as much as possible.

**#-----------------------------------------------------**

# Base Class: Person

class Person:

def \_\_init\_\_(self, person\_id):

self.\_person\_id = person\_id

def get\_summary(self):

return f"Person ID: {self.\_person\_id}"

#### Examples ####

uni\_students **=** load\_data**(**"oop\_uni\_students.csv"**)**

get\_students\_ready\_for\_graduation(uni\_students)

show\_probation\_list(uni\_students)

get\_student\_by\_id(uni\_students, 'S0001')

get\_student\_by\_id(uni\_students, 'S11') # invalid

get\_student\_by\_id(uni\_students, 'M0002')

Example Output

Students Ready for Graduation:

- S????: Ready for graduation.

- M????: Ready for graduation.

Probation List (GPA < 2.0):

- S????: GPA ?.??

Student Details:

Student ID: S????, Level: Undergraduate, GPA: ?.??

Credits Completed: ?

Graduation Status: Ready for graduation

Student Details:

Student ID: M????, Level: Master

Thesis Title: ?

Publications: ?

Years of Study: ?

Graduation Status: Not ready

**Submission (Group)**

* Submit the assessment as ONE(1) python script file via WBLE before the end of the practical session. Penalties will be imposed on any error, late submission, or plagiarism detected.
* State the program, name, id for each group member in the .py file.
* Name your script file in the following format:
  + **PA2\_<Group>.py** e.g. **PA2\_Group1.py**

**Marking Rubrics**

* 8-10 Program runs successfully with minor/no issues
* 5-7 Program runs but contains some errors
* 3-4 Program has errors with illogical implementation
* 0-2 Program has critical errors or is incomplete