Title (Units): **COMP2046 Problem Solving Using Object Oriented Approach (2,2,1)**

Course Aims: To study the object-oriented programming principles and techniques. Upon completion, students should be able to solve practical problems using an object-oriented programming language.

Prerequisite: COMP1005 Essence of Computing or COMP1007 Introduction to Python and Its Applications

Anti-requisite: COMP2026 Problem Solving Using Object Oriented Programming

Co-requisite: COMP2045 Programming and Problem Solving

Course Intended Learning Outcomes (CILOs):
Upon successful completion of this course, students should be able to:

No.	**Course Intended Learning Outcomes (CILOs)**
	Knowledge
1	Describe the fundamentals and concepts of object-oriented programming
2	Apply object-oriented programming concepts to construct computer programs
	Professional Skill
3	Formulate complex problems as modules so as to be solved systematically
	Attitude
4	Integrate robustness, reusability, and portability into software development

Calendar Description: This course practices the object-oriented programming concepts, principles, and techniques, including classes, objects, inheritance, and polymorphism, via solving practical problems.

Teaching and Learning Activities (TLAs):

CILOs	**Type of TLA**
1-4	Students will learn the concepts and the elements of an object-oriented programming language and the object-oriented principles and implementations via lectures.
1-4	Laboratories and machine problems are designed for students to incorporate object-oriented techniques into their programs.

Assessment:

No.	**Assessment Methods**	**Weighting**	**CILOs to be addressed**	**Description of Assessment Tasks**
1	Coding Assessment	39%	1 - 4	Laboratories coding exercises and take-home coding assignments are designed to measure how well the students have learned the concepts of an object-oriented language and apply them for problem solving. A number of machine problems will be given to students to train them to develop programs via object-oriented approach.
2	Quizzes and Tests	21%	1 - 2	Written and machine-assisted quizzes are designed at different stages of the course to assess students' ability to describe the fundamental concepts of object-oriented language and apply them to construct computer programme.
3	Examination	40%	1 - 4	Final examination questions are designed to see how far students have achieved their intended learning outcomes. Questions will primarily assess the student's ability in object-oriented approach for problem solving.

Assessment Rubrics:

	Excellent (A)	**Good (B)**	**Satisfactory (C)**	**Marginal Pass (D)**	**Fail (F)**
Principles of object-oriented programming	The student acquires excellent knowledge in the principles of object-oriented languages, namely, data encapsulation, inheritance, and polymorphism.	The student acquires sufficient knowledge in the principles of object-oriented languages, namely, data encapsulation, inheritance, and polymorphism.	The student acquires average knowledge in the principles of object-oriented languages, namely, data encapsulation, inheritance, and polymorphism.	The student is able to describe the meanings of data encapsulation, inheritance, and polymorphism, and to give simple examples on them.	The student is unable to describe the meanings of data encapsulation, inheritance, and polymorphism, and to give simple examples on them.
Applying object-oriented techniques to software packages	The student is able to extensively apply object-oriented techniques to write software applications with multiple classes, e.g., enforcing data hiding as much as possible via class privacy.	The student is able to sufficiently apply object-oriented techniques to write software applications with multiple classes, e.g., enforcing data hiding via class privacy.	The student is able to apply object-oriented techniques in some key elements of software applications with multiple classes, e.g., enforcing data hiding via class privacy.	The student can apply some object-oriented techniques to write software applications with multiple classes, e.g., enforcing data hiding via class privacy.	The student cannot apply object-oriented techniques to write software applications with multiple classes, e.g., enforcing data hiding via class privacy.
Design and implement object-oriented software for problem solving	The student demonstrates a strong ability in designing and implementing programs to solve moderately complex problems.	The student demonstrates a considerable ability in designing and implementing programs to solve moderately complex problems.	The student demonstrates an average ability in designing and implementing programs to solve moderately complex problems.	The student demonstrates some ability in designing and implementing programs to solve moderately complex problems.	The student does not demonstrate any ability in designing and implementing programs to solve moderately complex problems.

Course Content and CILOs Mapping:

Content		**CILO No.**
I	Object-Oriented Programming: Basic Elements	1-2
II	Advanced Concepts and Features	3-4

References:

- C. S. Horstmann and G. Cornell, Core Java 2 (Volume I-Fundamentals), Prentice Hall, 9th Edition, 2015.
- H. M. Deitel and P. J. Deitel, Java How to Program, Prentice Hall, 11th Edition, 2017.
- A. Kak, Programming with Objects: A Comparative Presentation of Object Oriented Programming with C++ and Java, Wiley-IEEE Press, 2003.
- D. Liang, Introduction to Java Programming, Prentice Hall, 9th Edition, 2014.
- G. Booch, R. A. Maksimchuk, M. W. Engel, and B J. Young, Object-oriented Analysis and Design with Applications, Addison-Wesley, 3rd Edition, 2007.

Course Content:

<u>Topic</u>

I. Object-Oriented Programming: Basic Elements

   A. Classes and objects
   B. References and dynamic memory
   C. Static and final
   D. Classification, generalization and specialization
   E. Constructing object-oriented program for problem solving

II. Advanced Concepts and Features

   A. Inheritance
   B. Polymorphism
   C. Interfaces and abstract classes