Subject Description Form

Subject Code	EIE3320 (for BEng in EIE, BSc in IMT and HD in EIE)
Subject Title	Object-Oriented Design and Programming
Credit Value	3
Level	3
Pre-requisite	For BEng in EIE and BSc in IMT: ENG2002 Computer Programming For HD in EIE: EIE2264 Computer Programming/EIE2111 Computer Programming
Co-requisite/ Exclusion	Nil
Objectives	This subject will provide students with the principles of object-oriented software design and programming from the perspective of Java implementation and UML. Students are expected to learn the concepts of and practical approaches to object-oriented analysis, design and programming using UML and Java.
Intended Subject Learning Outcomes	**Upon completion of the subject, students will be able to:** Category A: Professional/academic knowledge and skills 1. Understand the principles of object oriented design. 2. Apply Java in object oriented software development. 3. Apply UML in object oriented software modeling. 4. Apply object oriented approach to developing computer software. Category B: Attributes for all-roundedness 5. Learn independently and be able to search for the information required in solving problems. 6. Present ideas and findings effectively. 7. Think critically. 8. Work in a team and collaborate effectively with others.
Subject Synopsis/ Indicative Syllabus	**Syllabus:** 1. Introduction to Software Engineering Software products; software processes; software process models; 2. Java Programming Basic Java technologies; Java platform; Java language basic: variables, operators, expressions, statements, blocks, control flow, methods, arrays. 3. Object-Oriented Programming with Java Objects and classes; class definition; fields, constructors and methods; object interaction; grouping objects; array and collections; designing classes; inheritance and polymorphism; managing inheritance: creating subclasses and super-classes, hiding member variables, overriding methods. Interfaces and packages. 4. Data Structures with Java Implementation-dependent structures such as array and linked list; Implementation-independent structures such as stack, queue, list, map, tree, graph; Fundamental algorithms such as searching and sorting. 5. Unified Modelling Language (UML) Purposes of modelling. Structural Modelling: classes, relationships, class Diagrams, interfaces, packages, and object diagrams. Behavioural modelling interactions and use case diagrams. Architectural modelling:

components, deployment, and collaborations. Mapping UML diagrams to Java Code.

Laboratory Experiment:

Students will be requested to use integrated development environment (IDE) to write and debug Java programs during tutorial and lab sessions.

Teaching/ Learning Methodology

Teaching and Learning Method	Intended Subject Learning Outcome	Remarks
Lectures	1, 2, 3	fundamental principles and key concepts of the subject are delivered to students
Quizzes/Tests	1, 2, 3	students' knowledge on understanding of certain topics can be easily estimated, and the corresponding teaching time will be adjusted accordingly
Assignments	2,4,5,7	Programming exercises are used to reinforce the knowledge taught in lectures.
Laboratory sessions	2,3,4,5,6,7,8	Students will need to design, develop, test, and document Java programs.

Assessment Methods in Alignment with Intended Subject Learning Outcomes

Specific Assessment Methods/ Task	% Weighting	Intended Subject Learning Outcomes to be Assessed (Please tick as appropriate)							
		1	2	3	4	5	6	7	8
1. Continuous Assessment (Total: 100%)									
• Assignments	8%		✓		✓	✓		✓	
• Lab reports	20%		✓	✓	✓	✓	✓	✓	✓
• Knowledge Tests/ Quizzes	32%	✓		✓					
• Practical Tests	40%		✓		✓				
Total	100%								

The continuous assessment consists of programming assignments, laboratory reports, knowledge tests/quizzes and practical tests.

Explanation of the appropriateness of the assessment methods in assessing the intended learning outcomes:

Specific Assessment Methods/Tasks	Remark
Knowledge Tests/Quizzes	Short questions will be used to test and enhance students' understanding about the topics covered in lectures.

		End-of-chapter problems will be used to evaluate students' ability in applying concepts and skills learnt in the classroom.
	Assignments	Students will be asked to write Java programs and test the programs. Students will need to think critically and creatively in order to come up with a good solution for an existing problem.
	Lab reports	Each group of students are required to produce a written report for the Laboratory sessions. Students will be assessed based on the quality of their programs and the clarity of their reports. Students will be asked to work as a team to develop a Java application. Each of them will be responsible for part of the software. They will also need to use UML diagram to illustrate the structure of their programs. Students will need to think critically and creatively in order to come up with a good solution for an existing problem.
	Practical Tests	Students will be given programming problems and asked to write Java programs to solve the problems.

Student Study Effort Expected	**Class contact (time-tabled):**	
	• Lecture	26 Hours
	• Tutorial/Laboratory/Practice Classes	13 hours
	Other student study effort:	
	• Lecture: preview/review of notes; homework/assignment; preparation for test/quizzes/examination	36 Hours
	• Tutorial/Laboratory/Practice Classes: preview of materials, revision and/or reports writing	30 Hours
	Total student study effort:	**105 Hours**

Reading List and References

Reference Books:

1. G. Booch, I. Jacobson and J. Rumbaugh, *The Unified Modeling Language User Guide*, 2nd ed., Addison-Wesley, 2005.
2. D.J. Barnes and M. Kolling, *Objects First with Java: A Practical Introduction using BlueJ*, 5th ed., Prentice-Hall, 2012.
3. Nell Dale, Daniel T. Joyce, and Chip Weems. *Object-Oriented Data Structures Using Java (4th. ed.)*. Jones and Bartlett Publishers, Inc., USA. 2018.
4. H.M. Deitel and P.J. Deitel, *Java: How To Program (Early Objects)*, 10th ed., Prentice-Hall, 2014.
5. J. Lewis and W. Loftus, Java Software Solutions, 8th Edition, Pearson, 2015.
6. J. Rumbaugh, I. Jacobson and G. Booch, *The Unified Modeling Language Reference Manual*, 2nd ed., Addison-Wesley, 2004.

Last Updated	July 2020
Prepared by	Dr Pauli Lai and Mr Richard Pang