Subject code and name: PRG2104 Object Oriented Programming (Updated 14 Mar 2016)

Name(s) of academic staff	Dr Chin Teck Min
Rationale for the inclusion of the subject in the programme	This course teaches the student: • Understand the advanced object concepts (polymorphism, multiple inheritance, code injection) • Design program using good and effective advanced object oriented programming design methods. • Build program with object oriented API (GUI, Collection) and other third party library effectively.
Semester and year offered	Semester 1, Year 2
Credit value	4
Prerequisite (if any)	PRG1203 Object Oriented Programming Fundamentals

Subject learning outcomes and mapping to programme learning outcomes (PO)

Subject learning outcomes	Programme learning outcomes								
	PO 1	PO 2	PO 3	PO 4	PO 5	PO 6	PO 7	PO 8	PO9
Upon conclusion of this subject, the student should be able to:									
a. Explain advanced object-oriented programming concepts - inheritance, polymorphism, GUI programming.	✓	✓							
b. Apply advanced OOP concepts in problem solving with maintainability and elegant.	✓	✓					✓		
c. Demonstrate how to use java collections framework / library components and exceptions.	✓	✓	✓				✓		
d. Write Computer programs that use: object-oriented programming concepts - inheritance, polymorphism, GUI programming - event-driven programming, layout managers.	✓						✓		

Transferable skills

- An ability to use skills and techniques applicable in the field of advanced object oriented programming
- Ability to make decisions related to advanced object oriented point of view
- An ability to function effectively in a programming team in a senior advanced object oriented programmer

Synopsis

This subject emphasizes the object-oriented concepts of advanced programming. The students will learn effective ways of writing rigorous OO programs. The students will be taught to further appreciate the object-oriented approach through the writing of a GUI application.

Mode of delivery (lecture, tutorial, workshop, seminar, etc.)

Lectures, tutorials, practicals

Assessment methods and types

Final Examination		50%
Coursework		50%
• Test	20%	
• Project (Group work)	30%	

Content outline of the subject and learning time per topic

Topic	Lecture	Tutorial	Practical	Self learning	Overall (hours)
Classes & objects review • Classes & Objects, Instantiation • Predefined packages • User-defined packages	1	1	2	5	9
Inheritance and polymorphism • Superclass and subclass • Super keywords and super reference • Method overriding	4	2	4	14	24
Abstract classes and methods • Creating abstract classes and methods • Abstract superclass	4	2	4	14	24
Interface • Defining an interface • Interface implementation	4	2	4	14	24
Graphical user interfaces • Introduction to GUI components • Event handling • Inner classes	4	2	4	14	24
Advanced object oriented design • Abstract classes and static classes • Multiple inheritance and interfacing • Collections framework • Exception handling	4	2	4	14	24

Problem Solving concepts using Advance Object Oriented programming structure * Use of Polymorphism * Use of separation of roles * Use of Third Party Library * Advanced UML Modelling - Structural Diagram, Behaviour Diagram * Use of Collection API	4	2	4	14	24
Assessment • 3 coursework components • 2-hour examination				7	7
Total student learning time (SLT)	25	13	26	89	160

Main references	Dathan, Brahma, Ramnath, Sarnath(2015), Object-Oriented Analysis, Design and Implementation:An Integrated Approach, Springer Paul Deitel, Harvey Deitel, (2017), Java How to Program, Early Objects, 11th Edition, Pearson. Vikash Sharma (2018), Learning Scala Programming, Packt Publishing
Additional references	C Wu, 2009. An Introduction to Object-Oriented Programming with Java. McGraw-Hill. Bernd Bruegge., Allen Dutoit, 2009. Object-Oriented Software Engineering using UML, Patterns and Java. 3^{rd} Edtiton, Pearson. Skrien,2009. Object Oriented Design Using Java. McGraw-Hill
Other additional information	Nil