SMU
School of
Computing and
Information Systems

Course Code:	**IS442**
Course Name:	**Object Oriented Programming**
When was the course design document last verified by the Course Manager:	**03 June 2025**

NOTE: The information given in this document is for reference only; the updates given during the class sessions and/or eLearn will supersede the information given in this document.

IMPORTANT NOTICE

Please note that the below information is for Term 2, AY2024-25 and earlier. For AY2025-26 and onwards, there will be a new public course catalogue. Please watch out for email updates from RO/in BOSS.

1. Synopsis

This course focuses on fundamental concepts of developing programs using the object-oriented approach. There will be an emphasis on writing clean and efficient code, and the ability to use an appropriate data structure or algorithm to solve problems.

This course assumes no prior knowledge of Java.

2. Prerequisites/Co-requisites

Prerequisite(s):	IS111 Introduction to Programming/ SMT111 Programming for Smart City Solutions/ CS101 Programming Fundamentals I/ COR-IS1704 Computational Thinking and Programming
Mutually Exclusive:	CS102 Programming Fundamentals II

(Please check Course Catalogue in BOSS for updated information!)

3. Course Areas

Information Systems Major
Business Options
Econ Major-Related/Econ Options
Social Sciences/PLE Major-related
IS Major: Software Development Track
IS Major: Digital Business Solutioning Track

(Please check Course Catalogue in BOSS for updated information!)

4. Course Objectives

Upon completion of the course, students will be able to:

- **Apply** the key object-oriented programming and design techniques of abstraction, encapsulation, inheritance and polymorphism to a given scenario.
- **Sketch** UML class diagrams and sequence diagrams.
- **Create** and **debug** programs using the Java programming language.
- **Apply** good programming practices and design concepts to develop software.
- **Integrate** object-oriented thinking into application of problem-solving skills.
- **Appreciate** the role of algorithms and data structures in problem solving.

5. Competencies

1. **Demonstrate** understanding of *abstraction*:
    a. **Contrast** concepts of a *class* and an *object*.

   b. **Write** class definitions that represent abstract data types.
   c. **Write** class definitions that properly handle a life-cycle of its instances: objects.
   d. **Write** programs that safely utilizes data members (fields, attributes) and member functions (methods) associated with individual objects, or a class itself (static members).
2. **Demonstrate** understanding of *encapsulation*:
   a. **Write** programs that properly use encapsulation to control visibility of *public*, *private* and *protected* members by applying correct access modifiers.
   b. **Write** programs with packages that organize source code definitions.
3. **Demonstrate** understanding of *inheritance*:
   a. **Explain** the benefits of inheritance.
   b. **Write** a class which extends a parent class.
   c. **Contrast** inheritance of implementation (extends) and inheritance of interface (implements).
   d. **Write** programs that utilize inheritance to eliminate repetition by reusing member definitions from an existing data type to define new data types.
4. **Demonstrate** understanding of *polymorphism*:
   a. **Write** a child class that overloads methods from a parent class (static polymorphism).
   b. **Write** a child class that overrides methods from a parent class (dynamic polymorphism).
   c. **Write** programs that utilize polymorphism to allow for data type generalization, specialization or interface implementation.
5. **Demonstrate** understanding of *object-oriented design*:
   a. **Sketch** UML class diagrams and sequence diagrams.
   b. **Translate** UML class diagrams and sequence diagrams into code.
   c. **Explain** concepts of cohesion and coupling.
   d. **Explain** concepts of an exception, and exception propagation.
6. **Demonstrate** ability to write programs in the *Java* programming language:
   a. **Write** simple batch scripts to compile and run programs.
   b. **Explain** the workings of *CLASSPATH* and **write** programs that use external libraries.
   c. **Apply** fundamental and complex data types (arrays, lists, sets, maps) to solve basic problems.
   d. **Write** programs that properly handle exceptions to safely perform I/O operations.
   e. **Write** class definitions for custom exceptions.

6. Teaching Staff

Faculty: ZHANG Zhiyuan, ZHU Bin

7. Course Assessments

Assessment	Weightage (%)
Class Participation	10%
Quizzes	20%
Final Exam (aka Lab Test in previous terms)	40%
Group Project	30%
Total	**100%**

Note: Weightage may vary depending on the year the course is offered and the faculty/instructor teaching the course.

8. Course Assessment Details

Class participation (5%)

- Sub-components:
  - Attendance.
  - Submitting in-class exercises.
  - Recording in-class participation.
  - Recording reflections.
- Students who want to be rewarded for their in-class participation in a week should record the summary of their activity and submit it with reflections via eLearn (a link can be found at the Contents page for that week) before the end of the week (Sunday, 11:59pm).
- Lecturers reserve the right to give discretionary marks to students whose contributions are deemed particularly insightful and valuable.

Quizzes (20%)

- There will be 4 short take-home quizzes held during the course (each 4 marks).
- Each quiz is expected to take approximately 15-20 minutes.

Final Exam (aka Lab Test) (45%)

- There will be **ONE** longer in-class lab test held at the end of the course (week 13 / week 14).
- The lab test is expected to take approximately 2 hours.
- Past papers and further details will be shared during the course.
- The format is same with lab test in previous terms

Group Project (30%)

- There will be **ONE** group project as an important component of the course assessment criteria.
- Teams of 5-7 students will have to implement a system that addresses a real-world problem statement.
- Students will be graded based on the quality of submitted deliverables, as well as presentations.

9. Tooling

1. Java SE Development Kit (JDK) 22 (*LTS; long-term support*)
   a. Oracle: https://jdk.java.net/22/
   b. OpenJDK (open source): https://openjdk.org/projects/jdk/22/
2. Astah UML (http://astah.net/student-license-request)
3. VSCode: https://code.visualstudio.com/download

10. Lesson Plan

Session	Topics
1	Introduction to the course
2	objects and classes
3	UML Basics
4	Inheritance I
5	Inheritance II
6	Exceptions
7	Use Case and Sequence Diagram
8	Recess
9	Collections
10	Class variables and Methods
11	Arrays
12	Arrays
13	**Lab Test**

11. Reference Books

1. Martin Fowler, "*UML Distilled. A Brief Guide to the Standard Object Modeling Language*", 3rd Ed., ISBN: 978-0321193681.
2. Craig Larman, "*Applying UML and Patterns: An Introduction to Object-Oriented Analysis* [...]", 3rd Ed., ISBN: 978-0131489066.
3. Robert C. Martin, Micah Martin, "*Agile Principles, Patterns, and Practices in C#*", 1st Ed., ISBN: 978-0131857254.
4. Jeanne Boyarsky, Scott Selikoff, "*OCA Java SE 8 Programmer I Study Guide: Exam 1Z0-808*", 1st Ed., ISBN: 978-1259587535.
5. Jeanne Boyarsky, Scott Selikoff, "*OCP Java SE 11 Programmer I Study Guide: Exam 1Z0-815*", 1st Ed., ISBN: 978-1119584704.

12. University Policies

Academic Integrity

All acts of academic dishonesty (including, but not limited to, plagiarism, cheating, fabrication, facilitation of acts of academic dishonesty by others, unauthorized possession of exam questions, or tampering with the academic work of other students) are serious offences.
All work (whether oral or written) submitted for purposes of assessment must be the student's own work. Penalties for violation of the policy range from zero marks for the component assessment to expulsion, depending on the nature of the offense.
When in doubt, students should consult the instructors of the course. Details on the SMU Code of Academic Integrity may be accessed at https://smu.sharepoint.com/sites/oasis/SitePages/DOS-WKLSWC/UCSC.aspx.

Copyright Notice

Please note that all course materials are meant for personal use only, namely, for the purposes of teaching, studying and research. You are strictly not permitted to make copies of or print additional copies or distribute such copies of the course materials or any parts thereof, for commercial gain or exchange.
For the full copyright notice, please visit: https://smu.sg/Copyright-notice or *OASIS -> CAMPUS LIFE & EXCHANGE -> CONDUCT & DISCIPLINE -> UNIVERSITY COUNCIL OF STUDENT DISCIPLINE*

Accessibility
SMU strives to make learning experiences accessible for all. If you anticipate or experience physical or academic barriers due to disability, please let me know immediately. You are also welcome to contact the university's student accessibility support team if you have questions or concerns about academic provisions: accessibility@smu.edu.sg. Please be aware that the accessible tables in our seminar room should remain available for students who require them.

Digital Readiness for Teaching and Learning (DRTL)
As part of emergency preparedness, instructors may conduct lessons online via the Zoom platform during the term, to prepare students for online learning. During an actual emergency, students will be notified to access the Zoom platform for their online lessons. The class schedule will mirror the current face-to-face class timetable unless otherwise stated.