MCT619
Object-Oriented Programming
Module Handbook

Master of Science in Software Engineering &
Database Technologies (MScSED)
# Table of Contents

1 **Module Details** 2
   1.1 Module Description 2
   1.2 Prerequisites 2
   1.3 Module Objectives 2
   1.4 Required Text 3
   1.5 Module Assignments 3
   1.6 Module Grading 4

2 **Module Outline** 5

3 **Module Syllabus** 7
   3.1 Workshop One (Overview of Java Programming) 7
   3.2 Workshop Two (Object-Based Programming) 8
   3.3 Workshop Three (Applets and Graphical User Interfaces) 9
   3.4 Workshop Four (Classes as Building-Blocks) 10
   3.5 Workshop Five (Advanced OOP Concepts) 11
   3.6 Workshop Six (Arrays, Collections Framework, Files & Streams) 12
   3.7 Workshop Seven (Multi-Threaded Programming) 13

4 **Appendices** 14
   4.1 References 14
   4.2 Information Sources 14
1 Module Details

1.1 Module Description

MCT619, Object-Oriented Programming, provides detailed coverage of Object-Oriented (OO) programming principles. It focuses on programming in Java, an OO language that is modern, vendor-independent, and widely used in industry.

Recognising that programming requires skill as well as knowledge, this module places emphasis on the practical aspects of developing significant Java programs using a professional development environment. Students also gain practical experience of program design, testing, and debugging.

Specifically in this module, students learn how to model objects in software, define classes describing categories of objects, and make appropriate use of concepts such as inheritance, composition, encapsulation, polymorphism, abstract classes, and interfaces. As well as learning basic Java syntax and how to express OO concepts in Java, practical topics such as applets, graphics, data storage, multi-threaded programming, and exception handling are addressed.

1.2 Prerequisites

Students must have a good working knowledge of a procedural programming language such as C, to the standard attained through successful completion of MCT612 Application Programming.

1.3 Module Objectives

At the end of the module, students are expected to be able to competently:
• Describe in detail Object-Oriented Programming concepts (e.g. classes, inheritance, composition, modularity, polymorphism)
• Analyse and interpret complex Object-Oriented programs written in Java
• Develop Object-Oriented solutions to programming problems, implement them in Java, test and debug them
• Write code to demonstrate knowledge of array structures, files, applets, graphical programs, multi-threaded programming, and exceptions
• Evaluate and justify decisions made in programs (e.g. selection of data types, choice of decision and repetition structures, use of inheritance or composition)

1.4 Required Text


1.5 Module Assignments

Details of module assignments and a sample final examination are detailed fully in the *MCT619 Facilitator Guide*. Student assessment will take the form of:

• Weekly programming assignments to be completed in the Virtual Lab
• Participation in the weekly forum discussions
• A final exam consisting of Java program development tasks and theoretical questions

1.6 Module Grading
<table>
<thead>
<tr>
<th>Assessment</th>
<th>Weight</th>
<th>Workshop</th>
</tr>
</thead>
<tbody>
<tr>
<td>Forum participation</td>
<td>25%</td>
<td>1 to 7</td>
</tr>
<tr>
<td>Weekly assignments/labs</td>
<td>35%</td>
<td>1 to 7</td>
</tr>
<tr>
<td>Final exam</td>
<td>40%</td>
<td>8</td>
</tr>
<tr>
<td>Total</td>
<td>100%</td>
<td></td>
</tr>
</tbody>
</table>
2 Module Outline

<table>
<thead>
<tr>
<th>Workshop</th>
<th>Sections</th>
<th>Assessment Summary</th>
</tr>
</thead>
<tbody>
<tr>
<td>1: Overview of Java Programming</td>
<td>Overview of Object Oriented Programming</td>
<td>Write a program for calculating the areas of triangles and rectangles. Then write a short report on experimenting with the debugger.</td>
</tr>
<tr>
<td></td>
<td>Beginning Java Programming</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Fundamentals of Java Syntax</td>
<td></td>
</tr>
<tr>
<td>2: Object-Based Programming</td>
<td>Control Structures and Methods</td>
<td>Create a simple class Rectangle with attributes length and width and then create a more sophisticated Rectangle class.</td>
</tr>
<tr>
<td></td>
<td>Object-Based Programming</td>
<td></td>
</tr>
<tr>
<td>3: Applets and Graphical User Interfaces</td>
<td>Applets and Graphics</td>
<td>Develop an applet that displays a dartboard pattern with equally spaced concentric circles and radial lines.</td>
</tr>
<tr>
<td></td>
<td>Graphical User Interfaces</td>
<td></td>
</tr>
<tr>
<td>Workshop</td>
<td>Sections</td>
<td>Assessment Summary</td>
</tr>
<tr>
<td>------------------------------------</td>
<td>-----------------------------------------------</td>
<td>-------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>4: Classes as Building-Blocks</td>
<td>Classes as Building Blocks 1</td>
<td>Develop a class called BankAccount and the develop a class called LoanAccount, based on BankAccount.</td>
</tr>
<tr>
<td></td>
<td>Classes as Building Blocks 2</td>
<td></td>
</tr>
<tr>
<td>5: Advanced OOP Concepts</td>
<td>Advanced Object-Oriented Programming</td>
<td>Modify the Payroll system to include an additional Employee subclass.</td>
</tr>
<tr>
<td></td>
<td>Advanced OOP: Example</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Advanced OOP: Exception Handling</td>
<td></td>
</tr>
<tr>
<td>6: Arrays, Collections Framework, Files &amp; Streams</td>
<td>Arrays and Collections</td>
<td>Write a Java program to read and process a text file.</td>
</tr>
<tr>
<td></td>
<td>Files and Streams</td>
<td></td>
</tr>
<tr>
<td>7: Multi-Threaded Programming</td>
<td>Multi-Threaded Programming 1</td>
<td>Write three versions of an applet that bounce balls inside of its graphics area.</td>
</tr>
<tr>
<td></td>
<td>Multi-Threaded Programming 2</td>
<td></td>
</tr>
</tbody>
</table>
# 3 Module Syllabus

## 3.1 Workshop One (Overview of Java Programming)

### Objectives

Upon completion of this workshop, students are expected to be able to:

- Explain how hardware independence is achieved
- Discuss essential differences between different editions of Java
- Select a development environment
- Read and analyse simple Java programs
- Write a Java application to perform basic computation
- Compile, run, test, and debug Java programs
- Compare and contrast Java and C programming
- Work with variables and constants in Java
- Convert between different data types
- Describe and distinguish between different categories of operator

### Topics

- Module Overview
- What's So Special About OOP?
- Different Editions
- How to Write, Compile & Run a Program
- Our First & Second Java Programs
- Choosing a Java Development Environment
- Debugging
- Comparison of Java & C Syntax
- Data Types, Variables & Operators

### Suggested Activities

- The facilitator and students introduce themselves
- The facilitator explains module objectives, grading criteria, assignments and administrative matters
- The facilitator and students discuss the basic module material (e.g. What are the key differences between Java and C?; Write a Java version of a C program that you’ve done before.)
- The facilitator reviews the concepts of the reading material
- The class reviews the assigned learning materials (multimedia learning objects, presentation slides, videos, etc.)
- The facilitator assists the students in getting started with the Java Development Environment
- The class work on this week’s assignment(s) in the Virtual Lab
3.2 Workshop Two (Object-Based Programming)

Objectives
Upon completion of this workshop, students are expected to be able to:

- Describe, select and use Java control structures
- Define and use methods
- Contrast object-oriented and procedural approaches to programming
- Explain the concepts of objects and classes
- Develop Java code to define classes, create objects, and work with them
- Discuss and use member visibility keywords and constructors
- Explain and demonstrate use of static members, getters and setters

Topics
Control Structures and Methods
The Object-Oriented Perspective
Modelling Objects & Classes in Software
Defining a Class
Creating Objects, Using Object References
Constructors
Visibility of Class Members
“Get” & “Set” Methods, Static Members

Suggested Activities
- The class discusses last week’s lab exercise
- The facilitator leads a review of this workshop’s learning materials (e.g. Real-world examples of classes, objects, attributes and behaviours; Select an example and develop, as a group, an appropriate Java representation of it; Technical details of syntax)
- The class work on this week’s assignment(s) in the Virtual Lab
### 3.3 Workshop Three (Applets and Graphical User Interfaces)

#### Objectives
Upon completion of this workshop, students are expected to be able to:

- Write and deploy basic Java applets
- Develop applets and applications with simple graphics
- Explain the principles of event-driven programming
- Develop applets and applications with GUIs
- Discuss the purpose of Java GUI Builders
- Describe and use JavaBeans in GUI development

#### Topics
- Applications & Applets
- Running an Applet & Basic HTML Required
- Simple Graphics Commands
- GUIs & GUI Components
- Event-Driven Programs in Java
- Using GUI Builders
- JavaBeans, BeanInfo, and JAR Files

#### Suggested Activities
- The class reviews last week’s assignment
- The facilitator leads a review of this workshop’s learning materials (e.g. Writing applets with graphics, Relate Java’s GUI elements to GUIs of well-known programs; Event-driven programming and how to do it in Java; Coding GUI’s ‘by hand’ versus using a GUI Builder; The motivation for JavaBeans)
- The class work on this week’s assignment(s) in the Virtual Lab
### 3.4 Workshop Four (Classes as Building-Blocks)

**Objectives**

Upon completion of this workshop, students are expected to be able to:

- Explain OOP concepts covered, including encapsulation, packages, composition and inheritance
- Develop Java code demonstrating thorough understanding of these concepts
- Explain the concepts of inheritance and composition
- Demonstrate an understanding of constructors, finalizers, and garbage collection

**Topics**

<table>
<thead>
<tr>
<th>Modularity &amp; Encapsulation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Working with Packages</td>
</tr>
<tr>
<td>Inheritance &amp; the Substitutability Principle</td>
</tr>
<tr>
<td>The 'Object' Class</td>
</tr>
<tr>
<td>Composition</td>
</tr>
<tr>
<td>More about Constructors &amp; Finalizers</td>
</tr>
</tbody>
</table>

**Suggested Activities**

- The class reviews last week’s assignment
- The facilitator leads a review of this workshop’s learning materials (e.g. Real-world examples of hierarchies that could be modelled by inheritance; Real-world examples of composition; Select an example and develop, as a group, an appropriate OO model of one of these real-world examples)
- The class work on this week’s assignment(s) in the Virtual Lab
3.5 Workshop Five (Advanced OOP Concepts)

Objectives
Upon completion of this workshop, students are expected to be able to:

- Explain OOP concepts covered, including polymorphism, abstract classes and interfaces
- Develop Java code demonstrating thorough understanding of these concepts
- Design object-oriented programs that apply the ideas developed in Workshops 2, 4 and 5
- Implement these designs in coherent, well-structured Java code
- Discuss exception handling and demonstrate how it is done

Topics
Polymorphism
Abstract Classes
Interface Definitions
A Worked Example
Exceptions & Exception Handling

Suggested Activities
- The class reviews last week’s assignment
- The facilitator leads a review of this workshop’s learning materials (e.g. The concepts of polymorphism, abstract classes and interfaces, and their implications in OOP; The distinctions between abstract classes and interfaces; For the worked example, consider & analyse design alternatives and how they might be implemented; Practical benefits of exception handling)
- The class work on this week’s assignment(s) in the Virtual Lab
3.6 Workshop Six (Arrays, Collections Framework, Files & Streams)

Objectives
Upon completion of this workshop, students are expected to be able to:

- Describe and program with 1-D and multi-dimensional arrays, including initialising, indexing, manipulating and examining
- Explain the structure of the Collections Framework
- Describe and program with the ArrayList collection class
- Describe and apply the general model of file input/output in Java, and its variations
- Write code to perform input & output with text and binary files, including random access and object serialization
- Demonstrate the use of the File class for file management

Topics
- Arrays
- Multi-Dimensional Arrays
- Overview of Collections Framework
- ArrayList Collection Class
- File Input and Output in Java
- Object Persistence & Serialisation
- Random Access Files
- File and Folder Management

Suggested Activities
- The class reviews last week’s assignment
- The facilitator leads a review of this workshop’s learning materials (e.g. Using standard arrays versus ArrayList; Relating Java’s file I/O mechanisms to those encountered in other programming languages; Relating the File class to operations that may be performed in an OS file manager (such as Windows’ My Computer interface))
- The class work on this week’s assignment(s) in the Virtual Lab
3.7 Workshop Seven (Multi-Threaded Programming)

Objectives
Upon completion of this workshop, students are expected to be able to:

- Distinguish between threads and processes and explain what threads are used for
- Discuss issues such as Operating System dependence, life cycle, priority and daemons
- Explain the Java language constructs for writing multi-threaded applications
- Analyse and write Java code to work with threads
- Discuss synchronisation, deadlocks and race conditions

Topics
Processes Versus Threads
Life Cycle of a Thread
Threads in Java
Examples of Multi-Threaded Programming
Synchronisation & Deadlocks
Producer/Consumer Model

Suggested Activities
- The class reviews last week’s assignment
- The facilitator reviews with the students this week’s reading material and workshop material (The necessity for multi-threaded programming; cycle of a thread and how this relates to Java code; multi-threading implementation in Java; problems that may arise with synchronization; the producer/consumer model.
- The class work on this week’s assignment(s) in the Virtual Lab
4 Appendices

4.1 References

Official Java language documentation and software downloads:

http://www.oracle.com/technetwork/java/index.html

4.2 Information Sources

Discussion Forum:  http://forums.oracle.com
Bruce Eckel “Thinking in Java”: http://www.mindview.net/Books/TIJ/
New to Java programming: http://www.ibm.com/developerworks/java/newto/