**Object Oriented Programming**

**Overview**

|  |  |
| --- | --- |
| Level | 1 (Semester 2) |
| Duration | 4 weeks |
| Lectures | 10x40 minutes/week for 2 weeks |
| Practicals/tutorials | 5x2 hours/week for 3 weeks |

**Learning Outcomes**

* 1. Have knowledge & understanding of the core concepts of, and common practices in, object-oriented programming.
  2. Have knowledge and understanding of fundamentals of object-oriented analysis and design, including a range of design patterns.
  3. Understand the need for principled approaches to programming problems, and the ability to think algorithmically: to take a problem and break it down into various components, translating these component problems into data structures and algorithms that can solve them.
  4. Be able to use existing programming tools, frameworks and systems to build, test and manage solutions.
  5. Be able to use UML notation to document software systems.

**Syllabus**

* Course Introduction & Python Primer
* Object-Oriented Programming: An Introduction
* Intro & OOP in Python - I
* OOP in Python - II
* Data Structures and Algorithms: Stack and Queue
* OOP in Python III & Modules
* Data Structures and Algorithms: Recursion
* Object-Oriented Analysis & Design
* File and Exception Handling
* Python Best Practice
* Object-Oriented Design & UML
* Object-Oriented Testing
* Data Structures and Algorithms: Linked List
* Design Patterns Intro & Iterator
* Data Structures and Algorithms: Trees
* Maps, Dictionaries, and Hash Tables
* Design Patterns II
* Design Patterns III
* Architectural Patterns
* Binary Search Tree Lecture
* Git (Versioning)
* Agile Development and Course Wrap Up Lecture