| MODULE 8 | OBJECT ORIENTED PROGRAMMING | |
|---|---|---|
| CREDIT POINTS | 10 | |
| STATUS | Core | |
| ASSESSMENT | Continuous Assessment | 40% |
| | Examination | 60% |
| TOTAL CONTACT HOURS: 60 | | |
| Lecture: 24 | Practical: 36 | |
| Tutorial: | Other: | |
| TOTAL STUDENT EFFORT: 150 | | |

**Aims**

This module introduces you to the fundamental concepts of object oriented program design and how to use modelling for constructing complex software systems.

**Learning Outcomes**

Upon successful completion of this module, you should be able to:

1. develop the analytical skills necessary to apply abstract concepts in an object oriented manner
2. express system solutions in a formal manner and implement the derived formalisation
3. develop confidence in and awareness of the capabilities of object oriented development
4. produce correct software designs
5. analyse a problem, produce high quality software designs using Universal Modelling Language (UML) notation and relate the software designs to the implementation
6. identify problems associated with traditional methods of software specification, and explain how formal methods overcome these problems
7. develop high quality software that is reliable, reusable and maintainable

**Indicative Content**

| Topic | Description |
|---|---|
| Introduction and motivation | Review of procedural paradigm, control structures and data types with emphasis on structured data types and array processing. Case studies of software engineering failures. |
| The object paradigm | Classification: Objects and Object Types (Classes); Abstraction;<br>Encapsulation: Data and Behaviour;<br>Information Hiding: Access Specifiers;<br>Inheritance and Polymorphism;<br>Aggregation and Association;<br>Software Reuse; |
| Unified modelling language | Rationale and history of UML;<br>Use Case Analysis;<br>Structural View: Class and Object diagrams;<br>Behaviour View: Sequence, Collaboration, Statechart, Activity diagrams;<br>Environment View; |
| Object oriented programming | Implementation of classes and objects;<br>Static and Dynamic Objects;<br>Testing and debugging in Java;<br>Use of commercial libraries;<br>Sample programs; |
| Object oriented design | Implementation Options;<br>Object Oriented Methodologies;<br>Use of iterative development;<br>Introduction to Patterns and Anti-Patterns |
| Issues in OOD | Pros and cons of the OO approach;<br>Aspect-Oriented programming;<br>Object-Oriented Databases;<br>Remote/Distributed Objects |