<table>
<tr><td>1.</td><td>Name of Course</td><td colspan="6">Object Oriented Programming</td></tr>
<tr><td>2.</td><td>Course Code</td><td colspan="6">CCPS1563</td></tr>
<tr><td>3.</td><td>Name(s) of academic staff</td><td colspan="6"></td></tr>
<tr><td>4.</td><td>Rationale for the inclusion of the course/module in the programme</td><td colspan="6">Faculty<br>The study of Object-oriented programming and the development of programming skills are central to any undergraduate course in computing.<br>Object-oriented programming (OOP) is claimed to give more flexibility, easing changes to programs, and is widely popular in large scale software engineering. OOP is easier to learn for those new to computer programming than previous approaches.<br>Object-oriented programming builds upon the fundamental skills acquired in the module CCP0001 Computer Programming. This module is necessary for students to proceed to other modules in the program.</td></tr>
<tr><td>5.</td><td>Semester and Year offered</td><td colspan="6">2/2</td></tr>
<tr><td rowspan="3">6.</td><td rowspan="2">Total Student Learning Time (SLT)</td><td colspan="4">Face to Face</td><td colspan="2">Total Guided and Independent Learning</td></tr>
<tr><td rowspan="2">L = Lecture<br>T = Tutorial<br>P = Practical<br>O= Others</td><td>L</td><td>T</td><td>P</td><td>O</td><td rowspan="2">Independent = 84<br>Total =140</td></tr>
<tr><td>28</td><td></td><td>28</td><td></td></tr>
<tr><td>7.</td><td>Credit Value</td><td colspan="6">3</td></tr>
<tr><td>8.</td><td>Prerequisite (if any)</td><td colspan="6">CCP0001 Computer Programming</td></tr>
<tr><td>9.</td><td colspan="7">Objectives:<br>The objectives of this course are to introduce students to the key features of object-oriented technology as well as an industry standard methodology (UML) for Object-Oriented Analysis and Design. Students will be expected to analyze and design an Object-Oriented system in UML and implement it using C++</td></tr>
<tr><td>10.</td><td colspan="7">Learning outcomes:<br>By the end of the subject, students should be able to:<ul><li>Apply the key features of object-oriented technology as well as standard methodology (UML) for Object-Oriented Analysis and Design.</li><li>Demonstrate the basic notions and techniques for algorithm development</li><li>Implement algorithms using C++ correctly and effectively.</li></ul></td></tr>
<tr><td>11.</td><td colspan="7">Transferable Skills:<ul><li>Appreciate the importance of OOP principles.</li><li>Adopt a methodical approach to the creation and implementation of software development.</li><li>Be proficient in analyzing and designing an Object-Oriented system in UML and implement it using C++</li></ul></td></tr>
</table>

12. **Teaching-learning and assessment strategy**

A variety of teaching and learning strategies are used throughout the course, including:

- Classroom lessons: Lectures and Power Point presentations
- Laboratory sessions: Practice exercises
- brainstorming;
- student-Lecturer discussion
- collaborative and co-operative learning;
- Independent study.

Assessment strategies include the following:

- Ongoing quizzes
- Midterm tests
- Performance Assessment ( project, Assigned exercises)
- Lecturer Observation

13. **Synopsis:**

The major areas of study include: Basic Concepts of Object Oriented Technology, State, Behaviour, and Identity of Objects, Principles in Object-Oriented Programming, Object-Oriented Analysis and Design, Programming in C++ as Object-Oriented Language, and Case Studies

14. **Mode of Delivery:**

- Classroom lessons: Lectures and Power Point presentations
- Laboratory sessions: Practice exercises

15. **Assessment Methods and Types:**

The assessment for this course will be based on the following:

**Coursework** **50%**

- Quizzes and assignments 15%
- Project 15%
- Mid-Semester Exam 20%

**Final Exam** **50%**

100%

16. **Mapping of the course/module to the Programme Aims**

| A1 | A2 | A3 | A4 | A5 | A6 | A7 | A8 | A9 |
|---|---|---|---|---|---|---|---|---|
| 4 | 2 | 2 | 1 | 3 | 1 | 1 | 2 | 0 |

17. **Mapping of the course/module to the Programme Learning Outcomes**

| LO1 | LO2 | LO3 | LO4 | LO5 | LO6 | LO7 | LO8 | LO9 | LO10 | LO11 | LO12 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 4 | 1 | 0 | 0 | 3 | 0 | 1 | 2 | 0 | 1 | 0 | 0 |

18. **Content outline of the course/module and the SLT per topic**

| | Details | SLT | | | |
|---|---|---|---|---|---|
| | | L | P | Indep. | Total |
| **Topic 1** | **Overview of the subject**<br>• Historical background<br>• Classes and Objects,<br>• Abstraction<br>• Encapsulation | 2 | 2 | 6 | 10 |

| | | | | | | |
|---|---|---|---|---|---|---|
| | **Topic 2** | **Introduction to Object-Orientation**<br>• Inheritance,<br>• Polymorphism<br>• Message Passing<br>• OOAD Methodologies<br>• Introduction to UML | 4 | 4 | 12 | 20 |
| | **Topic 3** | **Object-Oriented Analysis**<br>• Syntax and Semantic<br>• examples of Use Case Diagrams, | 4 | 4 | 12 | 20 |
| | **Topic 4** | **Object-Oriented Analysis**<br>• Package Diagrams<br>• Class Diagrams<br>• Collaboration Diagrams<br>• Sequence Diagrams<br>• State Diagrams<br>• Activity Diagrams | 6 | 6 | 18 | 30 |
| | **Topic 5** | **Object-Oriented Design**<br>• Syntax and Semantic<br>• Deployment Diagrams.<br>• Implementation in an Object-Oriented Language | 4 | 4 | 12 | 24 |
| | **Topic 6** | **Case Studies**<br>Analyzing, Designing a Business Information System using UML and Implementing it Using C++ or Java. | 8 | 8 | 24 | 40 |
| | | Total | 28 | 28 | 84 | 140 |
| | | Laboratory Details<br>Exercises based on topics covered in each lecture. Experimental work must include the following<br>• Introduction to java environment.<br>• Classes ,Objects and Abstraction.<br>• Syntax and Semantic<br>• Message Passing<br>• Implementation in an Object-Oriented Language (C++, Java)<br>• Utility Classes (C++, Java)<br>• develop basic programs which contain a set of GUI components<br>• Assignments involve the modeling of simple real-world systems using UML diagrams.<br>• Designing a Business Information System using UML and implementing it Using C++ or Java. | | | | |
| 19. | | **Main references supporting the course:**<br>• Joyce Farrell, **Object-Oriented Programming** Using C++ ( 2008) | | | | |
| | | **Additional references supporting the course:**<br>1. Grady Booch, James Rumbaugh, and Ivar Jacobson, **Unified Modeling Language User Guide**, The (2nd Edition) (Addison-Wesley Object Technology Series) ( 2005)<br>2. *John W. Satzinger, Robert B. Jackson, and Stephen D. Burd,* ***Object-Oriented Analysis and Design with the Unified Process*** *( 2004)* | | | | |
| 20. | | **Other additional information**<br>All materials will be available to the students online. | | | | |