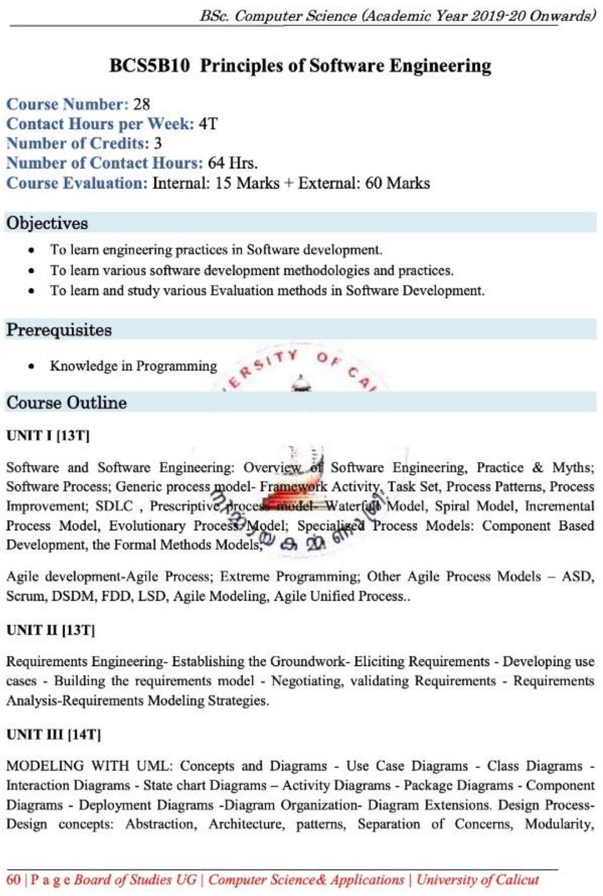
Syllabus

Python Programming



BCS5B10 Principles of Software Engineering

Course Number: 28
Contact Hours per Week: 4T
Number of Credits: 3
Number of Contact Hours: 64 Hrs.
Course Evaluation: Internal: 15 Marks + External: 60 Marks

Objectives

- To learn engineering practices in Software development.
- To learn various software development methodologies and practices.
- To learn and study various Evaluation methods in Software Development.

Prerequisites

- Knowledge in Programming

Course Outline

UNIT I [13T]

Software and Software Engineering: Overview of Software Engineering, Practice & Myths; Software Process; Generic process model- Framework Activity, Task Set, Process Patterns, Process Improvement; SDLC , Prescriptive process model- Waterfall Model, Spiral Model, Incremental Process Model, Evolutionary Process Model; Specialized Process Models: Component Based Development, the Formal Methods Models;

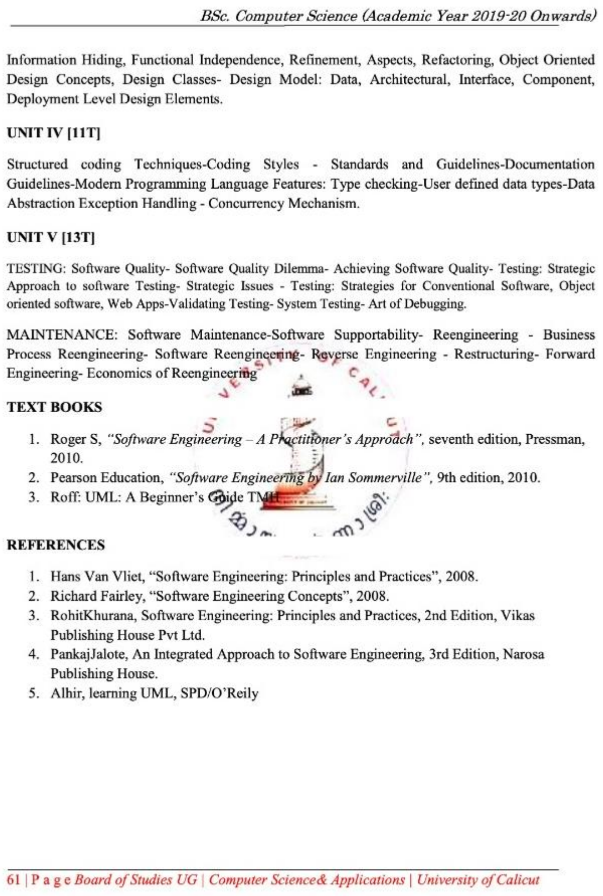
Agile development-Agile Process; Extreme Programming; Other Agile Process Models – ASD, Scrum, DSDM, FDD, LSD, Agile Modeling, Agile Unified Process..

UNIT II [13T]

Requirements Engineering- Establishing the Groundwork- Eliciting Requirements - Developing use cases - Building the requirements model - Negotiating, validating Requirements - Requirements Analysis-Requirements Modeling Strategies.

UNIT III [14T]

MODELING WITH UML: Concepts and Diagrams - Use Case Diagrams - Class Diagrams - Interaction Diagrams - State chart Diagrams – Activity Diagrams - Package Diagrams - Component Diagrams - Deployment Diagrams -Diagram Organization- Diagram Extensions. Design Process-Design concepts: Abstraction, Architecture, patterns, Separation of Concerns, Modularity,



Information Hiding, Functional Independence, Refinement, Aspects, Refactoring, Object Oriented Design Concepts, Design Classes- Design Model: Data, Architectural, Interface, Component, Deployment Level Design Elements.

UNIT IV [11T]

Structured coding Techniques-Coding Styles - Standards and Guidelines-Documentation Guidelines-Modern Programming Language Features: Type checking-User defined data types-Data Abstraction Exception Handling - Concurrency Mechanism.

UNIT V [13T]

TESTING: Software Quality- Software Quality Dilemma- Achieving Software Quality- Testing: Strategic Approach to software Testing- Strategic Issues - Testing: Strategies for Conventional Software, Object oriented software, Web Apps-Validating Testing- System Testing- Art of Debugging.

MAINTENANCE: Software Maintenance-Software Supportability- Reengineering - Business Process Reengineering- Software Reengineering- Reverse Engineering - Restructuring- Forward Engineering- Economics of Reengineering

TEXT BOOKS

1. Roger S, *"Software Engineering – A Practitioner's Approach"*, seventh edition, Pressman, 2010.
2. Pearson Education, *"Software Engineering by Ian Sommerville"*, 9th edition, 2010.
3. Roff: UML: A Beginner's Guide TMH

REFERENCES

1. Hans Van Vliet, "Software Engineering: Principles and Practices", 2008.
2. Richard Fairley, "Software Engineering Concepts", 2008.
3. RohitKhurana, Software Engineering: Principles and Practices, 2nd Edition, Vikas Publishing House Pvt Ltd.
4. PankajJalote, An Integrated Approach to Software Engineering, 3rd Edition, Narosa Publishing House.
5. Alhir, learning UML, SPD/O'Reily