DAYANANDA SAGAR COLLEGE OF ENGINEERING

(An Autonomous Institution affiliated to Visvesvaraya Technological University (VTU), Belagavi
Approved by AICTE and UGC, Accredited by NAAC with 'A' Grade & ISO 9001:2015 Certified Institution)
OUTCOME BASED EDUCATION (OBE) and CHOICE BASED CREDIT SYSTEM (CBCS)
SCHEME OF TEACHING AND EXAMINATIONS 2022

DEPARTMENT OF INFORMATION SCIENCE AND ENGINEERING

Software Engineering & Software Testing

Course code:	**IPCC22IS63**	**Credits:**	**04**
Prerequisites:	**Programming Languages**	**CIE Marks:**	**50**
Teaching Hours / Week (L : T : P)	**3 : 0 : 2**	**SEE Marks:**	**50**
Total Hours:	**52 [L: Lecture 40 + P: Practical 12]**	**Exam Hours:**	**03**

Course Objectives:	
1.	To understand the significance of software engineering principles and methodologies for the development of software projects.
2.	To analyze and apply the different process models for various types of software applications.
3.	To create qualitative software projects within project constraints in order to attain sustainability in software projects
4.	To analyze and apply the different testing methods for various types of software applications

Course Outcomes: At the end of the course, student will be able to:	
CO1	Apply software engineering principles and methodologies for projects which will be developed and tested.
CO2	Design and apply appropriate process models depending on the application requirement.
CO3	Develop qualitative projects while adhering to project restrictions and raise awareness of the need to create test cases based on the specifications provided.
CO4	Design test cases using various techniques & apply appropriate technique for designing control flow graphs.

Mapping of Course outcomes to Program outcomes:

	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PSO1	PSO2	PSO3
CO1	3	2						1				2			
CO2	3	2						2				1		1	
CO3	3	2						2				1		3	2
CO4	3				2			2				2			

Module	Module Contents	Hours
1.	**Introduction**: software engineering and its significance, Professional and ethical responsibility, System engineering, Organizations, people and computer systems, Legacy systems. Software Requirements: **Software requirements:** Functional and Non-functional requirements, User requirements, System requirements, Interface specification, The software requirements document, Software Requirements Specifications, Request for Proposal **Requirements engineering process:** Feasibility studies, Requirements elicitation and analysis, Requirements validation, Requirements management	8
2.	**Software Process Models:** The waterfall model, Evolutionary development model, spiral model, component-based software engineering, Process iteration, process activities, Rational Unified Process Agile methods, Extreme programming, Rapid application development and software prototyping	8
3.	**Design and Development:** Architectural design decisions, system organization, modular decomposition styles, control styles **Introduction to software testing:** Basics of Software Testing: Basic definitions, Software Quality, Requirements, Behavior and Correctness, Correctness versus Reliability, Testing and Debugging, Test cases, Insights from a Venn diagram, Identifying test cases, Test-generation Strategies, Test Life Cycle, Levels of testing, Testing and Verification, Static Testing	8
4.	**Functional Testing:** Boundary value analysis, Robustness testing, Worst-case testing, Robust Worst testing for triangle problem, Nextdate problem and commission problem, Equivalence classes, Equivalence test cases for the triangle problem, NextDate function, and the commission problem, Guidelines and observations. Decision tables, Test cases for the triangle problem, NextDate function and commission problem, Guidelines and observations.	8
5.	**Structural Testing:** Overview, Statement testing, Branch testing, Condition testing. Path testing: DD paths, Test coverage metrics, Basis path testing, guidelines and observations. **Data–Flow testing:** Definition-Use testing, Slice based testing. Guidelines and observations, Cyclomatic complexity and Examples.	8

Expt. No	Contents of the experiment	Hour s
1.	Design and develop a program in a language of your choice to solve the triangle problem defined as follows: Accept three integers which are supposed to be the three sides of a triangle and determine if the three values represent an equilateral triangle, isosceles triangle, scalene triangle, or they do not form a triangle at all. Assume that the upper limit for the size of any side is 10. Derive and compare test cases for your program based on boundary-value analysis, decision-table approach, compare and execute the test cases and discuss the results.	**2**
2.	Design, develop, code and run the program in any suitable language to implement the NextDate function. Analyze it from the perspective of boundary value testing, equivalence class testing and dataflow testing. Derive different test cases, compare them and execute these test cases and discuss the test results.	**2**
3.	Design, develop, code and run the program in any suitable language to solve the commission problem. Analyze it from the perspective of boundary-value analysis, equivalence class testing and dataflow testing. Derive different test cases, compare and execute these test cases and discuss the test results.	**2**
4.	Design, develop, code and run the program in any suitable language to implement an absolute letter grading procedure, making suitable assumptions. Determine the basis paths and using them derive different test cases, execute these test cases and discuss the test results.	**2**
5.	Design, develop, code and run the program in any suitable language to implement the binary search algorithm. Determine the basis paths and using them derive different test cases, execute these test cases and discuss the test results.	**2**
	Mini Project	
1.	Using any open-source software testing tool, generate a test plan document for any known application (Banking / Library Management system etc.)	

TEXT BOOKS: (Use latest edition textbooks)	
1.	Ian Somerville, Software Engineering, 10th Edition, Pearson Education, Pearson publications, 2017
2.	Paul C. Jorgensen: Software Testing, A Craftsman's Approach, 4th Edition, Auerbach Publications, 2017

REFERENCE BOOKS:	
1.	Software Engineering: A Practitioner's Approach by Rogers S Pressman, 9th edition, McGrawHill, 2023.
2.	Aditya P Mathur: Foundations of Software Testing, 2nd edition, Pearson Education, 2013
3.	Mauro Pezze, Michal Young: Software Testing and Analysis – Process, Principles and Techniques, Wiley India, 2019