B.TECH (CS, Computer Engineering and CSE) SIXTH SEMESTER SYLLABUS

Software Engineering (BCS601)		
Course Outcome (CO)		**Bloom's Knowledge Level (KL)**
At the end of course , the student will be able to		
CO 1	Explain various software characteristics and analyze different software Development Models	K_1, K_2
CO 2	Demonstrate the contents of a SRS and apply basic software quality assurance practices to ensure that design, development meet or exceed applicable standards	K_1, K_2
CO 3	Compare and contrast various methods for software design.	K_2, K_3
CO 4	Formulate testing strategy for software systems, employ techniques such as unit testing, Test driven development and functional testing	K_3
CO 5	Manage software development process independently as well as in teams and make use of Various software management tools for development, maintenance and analysis.	K_5

DETAILED SYLLABUS		**3-1-0**
Unit	**Topic**	**Proposed Lecture**
I	**Introduction:** Introduction to Software Engineering, Software Components, Software Characteristics, Software Crisis, Software Engineering Processes, Similarity and Differences from Conventional Engineering Processes, Software Quality Attributes. Software Development Life Cycle (SDLC) Models: Water Fall Model, Prototype Model, Spiral Model, Evolutionary Development Models, Iterative Enhancement Models.	**08**
II	**Software Requirement Specifications (SRS):** Requirement Engineering Process: Elicitation, Analysis, Documentation, Review and Management of User Needs, Feasibility Study, Information Modelling, Data Flow Diagrams, Entity Relationship Diagrams, Decision Tables, SRS Document, IEEE Standards for SRS. Software Quality Assurance (SQA): Verification and Validation, SQA Plans, Software Quality Frameworks, ISO 9000 Models, SEI-CMM Model.	**08**
III	**Software Design:** Basic Concept of Software Design, Architectural Design, Low Level Design: Modularization, Design Structure Charts, Pseudo Codes, Flow Charts, Coupling and Cohesion Measures, Design Strategies: Function Oriented Design, Object Oriented Design, Top-Down and Bottom-Up Design. Software Measurement and Metrics: Various Size Oriented Measures: Halestead's Software Science, Function Point (FP) Based Measures, Cyclomatic Complexity Measures: Control Flow Graphs.	**08**
IV	**Software Testing:** Testing Objectives, Unit Testing, Integration Testing, Acceptance Testing, Regression Testing, Testing for Functionality and Testing for Performance, TopDown and Bottom-Up Testing Strategies: Test Drivers and Test Stubs, Structural Testing (White Box Testing), Functional Testing (Black Box Testing), Test Data Suit Preparation, Alpha and Beta Testing of Products. Static Testing Strategies: Formal Technical Reviews (Peer Reviews), Walk Through, Code Inspection, Compliance with Design and Coding Standards.	**08**
V	**Software Maintenance and Software Project Management:** Software as an Evolutionary Entity, Need for Maintenance, Categories of Maintenance: Preventive, Corrective and Perfective Maintenance, Cost of Maintenance, Software Re- Engineering, Reverse Engineering. Software Configuration Management Activities, Change Control Process, Software Version Control, An Overview of CASE Tools. Estimation of Various Parameters such as Cost, Efforts, Schedule/Duration, Constructive Cost Models (COCOMO), Resource Allocation Models, Software Risk Analysis and Management.	**08**