Paper code : ARD 202	L	P	Credit
Subject : Software Engineering	4	0	4

Marking Scheme:
Teachers Continuous Evaluation: As per university examination norms from time to time.
End Term Theory Examination: As per university examination norms from time to time.

INSTRUCTIONS TO PAPER SETTERS: Maximum Marks : AS per University norms

- There should be 9 questions in the end term examination question paper
- Question No. 1 should be compulsory and cover the entire syllabus. This question should have objective or short answer type questions.
- Apart from Question No. 1, the rest of the paper shall consist of four units as per the syllabus. Every unit should have two questions. However, students may be asked to attempt only 1 question from each unit.
- The questions are to be framed keeping in view the learning outcomes of course/paper. The standard/ level of the questions to be asked should be at the level of the prescribed textbooks.
- The requirement of (scientific) calculators/ log-tables/ data-tables may be specified if required

Course Outcomes:
CO1: Student will be able to understand the concepts of Software Engineering.**[K1, K2, K3]**
CO2: Capability to perform requirement analysis and project planning of software systems. **[K2, K3]**
CO3: Student would be able to meet and understand the design and reliability of software systems.**[K1, K2, K4]**
CO4: Student would be able software testing techniques and software maintenance. **[K2, K3,K4]**

CO/PO	PO01	PO02	PO03	PO04	PO05	PO06	PO07	PO08	PO09	PO10	PO11	PO12
CO1	3	3	3	3	2	-	-	-	1	1	1	2
CO2	3	3	3	3	2	-	-	-	1	2	1	2
CO3	3	2	3	3	2	-	-	-	1	1	1	3
CO4	3	3	3	2	3	2	-	-	1	1	1	3

Course Content	**No of lectures**
Unit I Introduction: Software Engineering Paradigms. Software processes and its models (waterfall, Increment Process Models, Prototype Model, RAD, Spiral Model, Rational Unified Process) Agile Development model, plan driven vs agile model of development, agile methods and development techniques (user stories, refactoring, test first development, pair programming, agile project management (SCRUM agile method).	**[12]**

Unit II Software Requirement Analysis and Specification: Software Requirement Process, Functional and non-functional requirements, Quantifiable and Quality Requirements, System and software Requirements , requirement elicitation methods, requirement analysis and validation, requirement review or requirement change, SRS document. System modelling: Interaction models: Use case diagram, sequence diagrams, Structural models: class diagrams, generalization, aggregation, Behavioural models: ER diagrams, Data flow diagrams, data dictionaries. Software Project Planning: Software Size, LOC and Function point, cost and effort estimation, COCOMO Model.	**[12]**
Unit III Software Metrics: Project Metrics, Product Metrics and Process Metrics. Information flow Model Software Design: Architectural views and patterns, Modularity (cohesion and coupling), Information hiding, Functional independence, Function Oriented Design, Object Oriented Design, User Interface Design. Software Reliability : Failure and Faults , Reliability Models: Basic Model, Logarithmic Model	**[12]**
Unit IV Software Testing: Software process, Functional testing: Boundary value analysis, Equivalence class testing, Decision table testing, Cause effect graphing, Structural testing: Path testing, Data flow and mutation testing, unit testing, integration and system testing, User testing (alpha, beta and acceptance testing), Regression Testing, Stress Testing , Debugging, Testing Tools & Standards. Software Quality: McCall's Quality Factors, ISO 9126 Quality Factors, Quality Control, Quality Assurance, Software maintenance: Maintenance prediction, Re-Engineering, Refactoring. Software Configuration Management: Change Requirements , Version Control, Change Management. Case study of software Engineering	**[12]**

Text Books:

[T1] Pressman, R. S. (2005). *Software engineering*: a practitioner's approach. Palgrave macmillan.
[T2] Aggarwal, K. K. (2005). *Software engineering*. New Age International.
[T3] Ian Sommerville, "Software Engineering", 10th edition, Pearson, 2018.

Reference Books:

[R1] Sommerville, I. (2011). *Software Engineering*, 9/E. Pearson Education India.
[R2] Jalote, P. (2012). *An integrated approach to software engineering*. Springer Science & Business Media.
[R3] Bruegge, B., & Dutoit, A. H. (2009). *Object–oriented software engineering. using uml, patterns, and java*. Learning, 5(6), 7..
[R4] Blaha, M., & Rumbaugh, J. (2005). O*bject-oriented modeling and design with UML*. Pearson Education India.