MUTHAYAMMAL ENGINEERING COLLEGE

(An Autonomous Institution)

(Approved by AICTE, New Delhi, Accredited by NAAC & Affiliated to Anna University)

Rasipuram - 637 408, Namakkal Dist., Tamil Nadu.

MUST KNOW CONCEPTS	MKC
MCA	**2021-2022**

Course Code & Course Name :21CAB10 & Software Engineering

Year/Sem/Sec : I / II / -

S.No.	Term	Notation (Symbol)	Concept / Definition / Meaning / Units / Equation / Expression	Units
Unit-I : Introduction				
1.	Software Engineering	--	The application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software.	I
2.	Quality	--	Software quality measures how well the software is designed (quality of design), and how well the software conforms to that design (quality of conformance).	I
3.	Process	--	A software process is the set of activities and associated outcome that produce a software product. Software engineers mostly carry out these activities.	I
4.	Methods	--	Software development methodology is a process or series of processes used in software development.	I
5.	Waterfall life cycle model	--	The waterfall Model illustrates the software development process in a linear sequential flow.	I
6.	Feasibility Study	--	Feasibility Study in Software Engineering is a study to evaluate feasibility of proposed project or system.	I

7.	Design	--	Software design is the software requirements analysis (SRA). SRA is a part of the software development process that lists specifications used in software engineering.	I
8.	Coding	--	The coding is the process of transforming the design of a system into a computer language format.	I
9.	Unit Testing	--	Unit Testing of software product is carried out during the development of an application.In SDLC or V Model, Unit testing is first level of testing done before integration testing.	I
10.	Maintenance	--	Software maintenance is a part of Software Development Life Cycle. Its main purpose is to modify and update software application after delivery to correct faults and to improve performance.	I
11.	Analysis	--	Requirement Analysis, also known as Requirement Engineering, is the process of defining user expectations for a new software being built or modified. In software engineering.	I
12.	Prototyping Model	--	Prototyping Model is a software development model in which prototype is built, tested, and reworked until an acceptable prototype is achieved.	I
13.	Incremental Model	--	A process of software development where requirements are broken down into multiple standalone modules of software development cycle.	I
14.	Spiral Model	--	The spiral model is similar to the incremental development for a system, with more emphasis placed on risk analysis.	I
15.	RAD Model	--	RAD model is based on prototyping and iterative model with no (or less) specific planning. In general, RAD approach to software development means putting lesser emphasis on planning tasks and more emphasis on development and coming up with a prototype.	I

16.	Win-Win Spiral Model	--	The Win-Win spiral software engineering methodology expands the Boehm-Spiral methodology by adding a priority setting step, the Win-Win process, at the beginning of each spiral cycle.	I
17.	4GT	--	The term fourth generation techniques (4GT) encompasses a broad array of software tools that have one thing in common: each enables the software engineer to specify some characteristic of software at a high level.	I
18.	software process	--	Software process is defined as the structured set of activities that are required to develop the software system.	I
19.	System Engineering	--	System Engineering means designing, implementing, deploying and operating systems which include hardware,software and people	I
20.	Scheduling Techniques	--	Techniques such as PERT (Program Evaluation and Review Technique), CPM (Critical Path Method) and GANTT are the most used to plan into details a project, prevent uncertainties and avoid risk.	I
21.	PERT	--	Project Evaluation and Review Technique (PERT) is a procedure through which activities of a project are represented in its appropriate sequence and timing.	I
22.	CPM	--	Critical Path Method (CPM) is an algorithm for planning, managing and analyzing the timing of a project.	I
23.	.Software Risk	--	Software risk encompasses the probability of occurrence for uncertain events and their potential for loss within an organization.	I
24.	Requirements.	--	In product development and process optimization, a requirement is a singular documented physical or functional need that a particular design, product or process aims to satisfy	I
25.	System Requirements.	--	System requirements are all of the requirements at the system level that describe the functions which the system as a whole should fulfill to satisfy the	I

			stakeholder needs and requirements stakeholder needs and requirements.	
Unit-II : SOFTWARE DESIGN				
26.	Software Design	--	Software design is a process to transform user requirements into some suitable form, which helps the programmer in software coding and implementation.	II
27.	Problem Partitioning	--	small problem, we can handle the entire problem at once but for the significant problem, divide the problems and conquer the problem it means to divide the problem into smaller pieces	II
28.	Abstraction	--	Abstraction is one of the fundamental concepts of software engineering. It is all about hiding complexity in building various parts of your application.	II
29.	Modularity.	--	Software modularity is the decomposition of a program into smaller programs with standardized interfaces.	II
30.	Information Hiding	--	Information hiding is the principle of segregation of the design decisions in a computer program that are most likely to change, thus protecting other parts of the program from extensive modification	II
31.	Cohesion	--	Cohesion measures the extent to which all elements of a module belong together. cohesion examines how the activities within a module are related to one another.	II
32.	Coupling	--	Coupling is the degree of interdependence between software modules; a measure of how closely connected two routines or modules are; the strength of the relationships between modules.	II
33.	Data Flow	--	Data flow is the movement of data through a system comprised of software, hardware or a combination of both.	II
34.	Data Dictionary	--	A Data Dictionary is a collection of names, definitions, and attributes about data elements that are being used or captured in a database, information system, or part of a research project.	II
35.	Data Structures	--	Data Structures are a specialized means of	II

			organizing and storing data in computers in such a way that we can perform operations on the stored data more efficiently.	
36.	Data Design	--	Data design is the first design activity, which results in less complex, modular and efficient program structure.	II
37.	Pseudo-Code	--	Pseudocode is an informal way of programming description that does not require any strict programming language syntax or underlying technology considerations.	II
38.	Distributed System Design	--	A distributed system is "a collection of independent computers that appears to the user as a single coherent system."	II
39.	Documentation	--	Documentation in software engineering is the umbrella term that encompasses all written documents and materials dealing with a software product's development and use.	II
40.	JSD	--	Jackson System Development (JSD) is a method of system development that covers the software life cycle either directly or by providing a framework into which more specialized techniques can fit.	II
41.	Software Reuse	--	Software reuse is the process of implementing or updating software systems using existing software assets.	II
42.	Design For Reuse	--	Design reuse is the process of building new software applications and tools by reusing previously developed designs.	II
43.	COTS	--	Commercial-off-the-shelf (COTS) software is a term for software products that are ready-made and available for purchase in the commercial market.	II
44.	Phases of JSD	--	• Entity/action step • Initial model step • Interactive function step • Information function step • System timing step • System implementation step	II
45.	Object Oriented Design	--	Object-oriented design (OOD) is the process of using an object-oriented methodology to design a computing	II

			system or application.	
46.	Classification of Modules	--	• Incremental Module. • Sequential Module. • Parallel Modules	II
47.	Stepwise Refinement	--	Stepwise refinement is the idea that software is developed by moving through the levels of abstraction, beginning at higher levels and, incrementally refining the software.	II
48.	Control Hierarchy	--	Control hierarchy, also called program structure, represents the organization of program components (modules) and implies a hierarchy of control.	II
49.	Fan-in and Fan-out	--	Fan-in refers to the maximum number of input signals that feed the input equations of a logic cell. Fan-out refers to the maximum number of output signals that are fed by the output equations of a logic cell.	II
50.	Archetype	--	An archetype is a generic model of some important component in a system. an archetype is a generic model of some important component in a system.	II
Unit-III : SOFTWARE TESTING AND MAINTENANCE				
51.	Software Testing	--	Software testing is the process of evaluating and verifying that a software product. The benefits of testing include preventing bugs, reducing development costs and improving performance.	III
52.	Fault	--	It is an incorrect step in any process and data definition in computer program which is responsible of the unintended behavior of any program in the computer.	III
53.	Unit Testing	--	Unit Testing is defined as a type of software testing where individual components of a software are tested. Unit Testing of software product is carried out during the development of an application.	III
54.	Verification Testing	--	Verification is the process of checking that a software achieves its goal without any bugs.	III
55.	Validation Testing	--	Validation testing in software engineering	III

			is in place to determine if the existing system complies with the system requirements and performs the dedicated functions for which it is designed along with meeting the goals and needs of the organisation.	
56.	Test Cases	--	A test case is exactly what it sounds like: a test scenario measuring functionality across a set of actions or conditions to verify the expected result.	III
57.	Integration Testing	--	Integration testing (sometimes called integration and testing, abbreviated I&T) is the phase in software testing in which individual software modules are combined and tested as a group.	III
58.	System Testing	--	System Testing is the level of software testing performed before Acceptance Testing and after Integration Testing.	III
59.	Alpha Testing	--	Alpha Testing is a type of software testing performed to identify bugs before releasing the software product to the real users or public.	III
60.	Beta Testing	--	Beta testing is a type of user acceptance testing where the product team gives a nearly finished product to a group of target users to evaluate product performance in the real world.	III
61.	White Box Testing	--	White box testing is an approach that allows testers to inspect and verify the inner workings of a software system—its code, infrastructure, and integrations with external systems.	III
62.	Black Box Testing	--	Black box testing refers to any type of software test that examines an application without knowledge of the internal design, structure, or implementation of the software project.	III
63.	Functional Testing	--	Functional Testing is a type of Software Testing in which the system is tested against the functional requirements and specifications.	III
64.	System Testing	--	System Testing is the level of software testing performed before Acceptance Testing and after Integration Testing.	III

65.	Reliability Testing	--	Reliability Testing is a software testing process that checks whether the software can perform a failure-free operation for a specified time period in a particular environment.	III
66.	Acceptance Testing	--	Acceptance Testing is a method of software testing where a system is tested for acceptability.	III
67.	Testing Tools	--	Tools from a software testing context can be defined as a product that supports one or more test activities right from planning, requirements, creating a build, test execution, defect logging and test analysis.	III
68.	Smoke Testing	--	Smoke Testing is a software testing process that determines whether the deployed software build is stable. Smoke testing is also known as "Build Verification Testing" or "Confidence Testing."	III
69.	Maintenance	--	Software maintenance in software engineering is the modification of a software product after delivery to correct faults, to improve performance or other attributes.	III
70.	Types of Maintenance	--	• Corrective Software Maintenance. • Adaptive Software Maintenance. • Perfective Software Maintenance. • Preventive Software Maintenance.	III
71.	Security Testing	--	Security Testing is a type of Software Testing that uncovers vulnerabilities of the system and determines that the data and resources of the system are protected from possible intruders.	III
72.	Performance Testing	--	Performance Testing is a type of software testing that ensures software applications to perform properly under their expected workload.	III
73.	Recovery Testing	--	In software testing, recovery testing is the activity of testing how well an application is able to recover from crashes, hardware failures and other similar problems.	III
74.	Thread Testing	--	Thread testing, a software testing technique used during early integration	III

			testing phase to verify the key functional capabilities that carry out specific task.	
75.	Equivalence Partitioning	--	Equivalence Class Partitioning are the most common technique in Black-box Testing Techniques for test case design.	III
Unit-IV : SOFTWARE METRICS				
76.	Software Measurement	--	Software measurement is a quantified attribute (see also: measurement) of a characteristic of a software product or the software process. It is a discipline within software engineering.	IV
77.	Direct Measures	--	Direct measures of software engineering process include cost and effort. Direct measures of the product include lines of code (LOC), execution speed, memory size, defects per reporting time period.	IV
78.	Indirect Measures	--	Indirect measures include functionality, quality, complexity, efficiency, reliability, and maintainability. indirect measures. measures of software engineering output and quality.	IV
79.	Software Metrics	--	Software metrics are valuable for many reasons, including measuring software performance, planning work items, measuring productivity, and many other uses.	IV
80.	Scope of Metrics	--	**Cost** – Estimate project cost includes its maintenance, research, and other typical expenditure associated with the project. **Quality assurance** – Different metrics are used to measure different aspects of software quality, especially code quality (line of code). **Size and Complexity** – It demonstrates the code size and complexity at the macro level of projects. **Functionality** – Software metrics follow a scheduled procedure of software projects that focus on functionality, a document produced, and estimated time utilization.	IV

81.	Product Metrics	--	Software product metrics are measures of software products such as source code and design documents. Software process metrics are measures of software development process.	IV
82.	Lines of Code	--	The phrase “lines of code” (LOC) is a metric generally used to evaluate a software program or code base according to its size.	IV
83.	Size Metrics	--	Size Oriented Metrics are also used for measuring and comparing productivity of programmers. It is a direct measure of a Software.	IV
84.	Cost Estimation	--	The cost estimate is the financial spend that is done on the efforts to develop and test software in Software Engineering.	IV
85.	COCOMO Model	--	Cocomo (Constructive Cost Model) is a regression model based on LOC, i.e number of Lines of Code. A project such as size, effort, cost, time and quality.	IV
86.	Cyclomatic Complexity	--	Cyclomatic complexity is a software metric used to indicate the complexity of a program. It is a quantitative measure of the number of linearly independent paths through a program's source code.	IV
87.	Software Quality Assurance	--	Software quality assurance (SQA) is a means and practice of monitoring the software engineering processes and methods used in a project to ensure proper quality of the software.	IV
88.	SQA Activities	--	Software Quality Assurance (SQA) is simply a way to assure quality in the software. It is the set of activities which ensure processes, procedures as well as standards are suitable for the project and implemented correctly.	IV
89.	Complexity Metrics	--	The performance of three different software complexity metrics; McCabe's cyclomatic complexity, Halstead's complexity measures and Douce's spatial complexity, by using data from an Open Source project Eclipse JDT.	IV
90.	Classification of Metrics	--	It can be classified into three categories: product metrics, process	IV

			metrics, and project metrics. Product metrics describe the characteristics of the product such as size, complexity, design features, performance, and quality level.	
91.	Function Point Metrics	--	Function points are a unit of measure used to define the value that the end user derives, or the functional business requirements the software is designed to	IV
92.	Halstead Theory of Software	--	Halstead's metrics are included in a number of current commercial tools that count software lines of code. n1* = Number of potential operators. n2* = Number of potential operands. Halstead refers to n1* and n2* as the minimum possible number of operators and operands for a module and a program respectively.	IV
93.	Product Complexity	--	Product complexity can be e.g. the number of products, the number of components they consist of or raw materials used.	IV
94.	Algorithm Method	--	An algorithm (pronounced AL-go-rith-um) is a procedure or formula for solving a problem, based on conducting a sequence of specified actions.	IV
95.	Dynamic Metrics	--	Dynamic metrics are the class of software metrics that capture the dynamic behaviour of the software system and are usually obtained from the execution traces of the code or from the executable	IV
96.	Static Metrics	--	Static metrics that are collected by measurements made from system representations such as design, programs, or documentation.	IV
97.	Reliability Metrics	--	Reliability metrics are used to quantitatively expressed the reliability of the software product.	IV
98.	Product Quality	--	Software quality is defined as a field of study and practice that describes the desirable attributes of software products.	IV
99.	Process Quality	--	Process quality refers to the degree to which an acceptable process, including measurements and criteria for quality, has been implemented and adhered to in order to produce the artifacts. Software development requires a complex web of	IV

			sequential and parallel steps.	
100.	Quality Standard	--	Quality standards are defined as documents that provide requirements, specifications, guidelines, or characteristics that can be used consistently to ensure that materials, products, processes, and services are fit for their purpose.	IV
Unit-V : SCM				
101.	SCM	--	Software Configuration Management (SCM or S/W CM) is the task of tracking and controlling changes in the software, part of the larger cross-disciplinary field of configuration management.	V
102.	Baseline	--	A baseline is a reference point in the software development life cycle marked by the completion and formal approval of a set of predefined work products.	V
103.	Need for SCM	--	Software Configuration Management(SCM) is a process to systematically manage, organize, and control the changes in the documents, codes, and other entities during the Software Development Life Cycle.	V
104.	SCM Process	--	It uses the tools which keep that the necessary change has been implemented adequately to the appropriate component. The SCM process defines a number of tasks: • Identification of objects in the software configuration • Version Control • Change Control • Configuration Audit	V
105.	Software Library	--	A software library is a suite of data and programming code that is used to develop software programs and applications. It is designed to assist both the programmer and the programming language compiler in building and executing software.	V

106.	Configuration Control	--	Configuration Control is the activity of managing the product (or project's deliverable) and related documents, throughout the life cycle of the product.	V
107.	SCM Repository	--	Software configuration management (SCM) is any kind of practice that tracks and provides control over changes to source code. Software developers sometimes use revision control software to maintain documentation and configuration files as well as source code.	V
108.	Risk Management	--	Risk management is the process of identifying, assessing and controlling threats to an organization's capital and earnings.	V
109.	Features of CASE Tools	--	• Standard Methodology. • Flexibility. • Strong Integration. • Integration with Testing Software. • Support for Reserve Engineering. • Online help.	V
110.	CASE Repository	--	A CASE Repository should be the representation, in data, of all relevant information about the system under development, in a consistent, complete form which is independent of its mode of entry and modification or subsequent use.	V
111.	Information Repository	--	In software development, a repository is a central file storage location. It is used by version control systems to store multiple versions of files.	V
112.	Data Dictionary	--	A data dictionary in Software Engineering means a file or a set of files that includes a database's metadata, like data ownership, relationships of the data to another object, and some other data.	V
113.	Web Engineering	--	Web Engineering is the application of systematic, disciplined and quantifiable approaches to development, operation, and maintenance of Web-based applications.	V
114.	Need for Web Engineering	--	Web Engineering is the application of systematic, disciplined and quantifiable approaches to development, operation, and maintenance of Web-based applications.	V

115.	HCL	--	Human-Computer Interaction (HCI) are both relatively new disciplines of computer science.	V
116.	HTML	--	Hypertext Markup Language, a formatting system for displaying material retrieved over the Internet. HTML markup tags specify document elements such as headings, paragraphs, and tables.	V
117.	Taxonomy	--	A taxonomy is a "knowledge organization system," a set of words that have been organized to control the use of terms used in a subject field into a "vocabulary" to facilitate the storing and retrieving of items from a repository.	V
118.	Advantages of CASE	--	• Increased efficiency • Cost reduction • Enhanced system and process reliability • efficient change management • Faster restoration of your service if a process failure occurs	V
119.	Version Control	--	Version control is the control of deliverables whereas configuration management is managing the entire process leading to produce the deliverables. Configuration management involves change management, project management.	V
120.	Layers of SCM Process	--	The five tasks of the SCM process are configuration identification, change control, version control, configuration auditing, and reporting.	V
121.	CSR	--	Configuration Status Reporting the recording and reporting of information needed for configuration management including the status of configuration items (CIs), proposed changes and the implementation status of approved changes.	V
122.	Configuration Audit	--	Configuration auditing is conducted by auditors by checking that defined processes are being followed and ensuring that the SCM goals are satisfied.	V

123.	Data Integrity	--	Data integrity is a fundamental component of information security. In its broadest use, “data integrity” refers to the accuracy and consistency of data stored in a database, data warehouse, data mart or other construct.	V
124.	Tool Integration	--	Software Configuration Management (SCM) Tools handle the task of tracking and controlling changes in the software.	V
125.	Data Integration	--	This function data present in databases is integrated	V
Placement Questions				
126.	Concept Of Modularization.Concept Of Modularization.	--	Modularization is used to divide software into multiple components or modules. Each module is worked upon by an independent development and testing team.	
127.	The Various Phases Of SDLC	--	• Requirement Analysis • Design • Coding • Testing • Maintenance	
128.	Project Management Tools.	--	• Gantt Chart • Checklists • Status Reports • Histograms • Microsoft Project	
129.	Functional Requirements	--	Functional requirements are the features that a developed software product is expected to perform.	
130.	Baseline	--	A baseline is a milestone on the project which is usually defined by the project manager.	
131.	Coding	--	This is the phase where the code for the system to be developed is written.Unit Testing and Integration Testing must be performed by the developers at this stage before deploying the code for testing.	
132.	V-Model	--	V-Model stands for the verification and validation model. V-model is an addition to the waterfall model, in the sense that V-model is also a sequential model.	

133.	Software Project Manager	--	A software project manager is a person who undertakes the responsibility of carrying out the software project.	
134.	Function Points	--	Function points are the various features provided by the software product. It is considered as a unit of measurement for software size.	
135.	Measure Project Execution	--	Measure project execution by means of Activity Monitoring, Status Reports and Milestone Checklists.	
136.	Functional Requirements	--	Functional requirements are functional features and specifications expected by users from the proposed software product.	
137.	Cohesion	--	Cohesion is a measure that defines the degree of intra-dependability among the elements of the module.	
138.	Formula To Calculate Cyclomatic Complexity	--	Cyclomatic complexity uses graph theory's formula: V(G) = e – n + 2	
139.	Software Analysis & Design Tools	--	software analysis & design tools are Data flow Diagrams (DFD), Structured Charts, Data Dictionary, UML (Unified Modeling Languages) diagrams, ER (Entity Relationship) Diagrams etc.	
140.	Data Dictionary	--	A data dictionary is also known as metadata. Data Dictionary is utilized to capture the information related to naming conventions of objects and files utilized in the software project.	
141.	Corrective	--	This type of maintenance is used to remove the errors spotted by business users.	
142.	Preventive	--	This maintenance activity is performed to avoid any issues in future implementations.	
143.	Unit Testing	--	A programmatic test that tests the internal working of a unit of code, such as a method or a function.	
144.	Test Environment	--	A test environment consists of a server/computer on which a tester runs their tests.	
145.	Beta Testing	--	The software to the customers after alpha testing, the software's actual users	

			perform the beta testing in a real production environment.	
146.	Performance Testing	--	It is a type of non-functional software testing technique that is used to determine the system parameters like speed, scalability, and stability under different workload conditions.	
147.	Test Stubs	--	Test stubs are used in a top-down testing approach and allow testing of the upper levels of the code when the lower levels of the code are not developed.	
148.	Path Testing	--	In this type of testing, the control flow graph of a program is specially designed to identify a set of linearly independent paths of execution.	
149.	Categories Of Debugging	--	• Brute force debugging • Backtracking • Cause elimination • Program slicing • Fault tree analysis	
150.	Types Of Integration Testing	--	• Big bang testing • Bottom-Up Testing • Top-Down Testing	

Mrs. R.Pavithra

Faculty Prepared **Signature**

HoD