SUMMER – 19 EXAMINATION

Subject Name: Software Engineering Model Answer Subject Code:17513

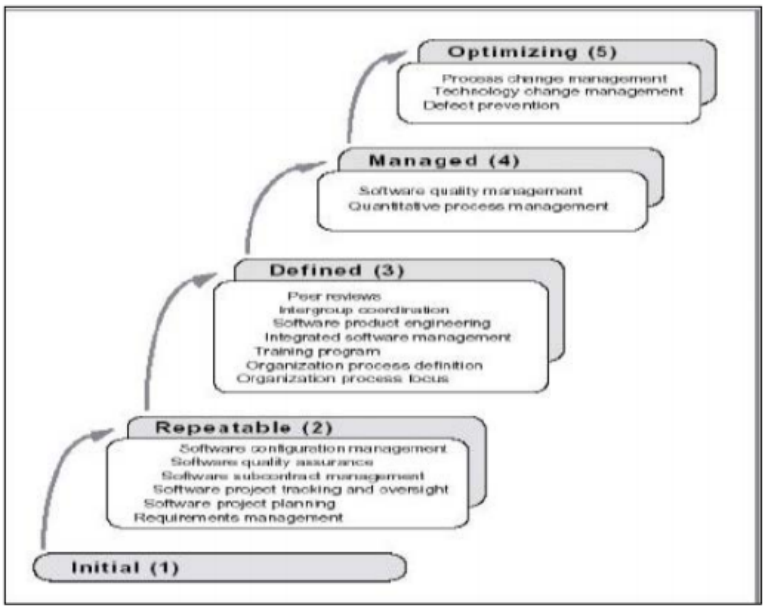
Important Instructions to examiners:

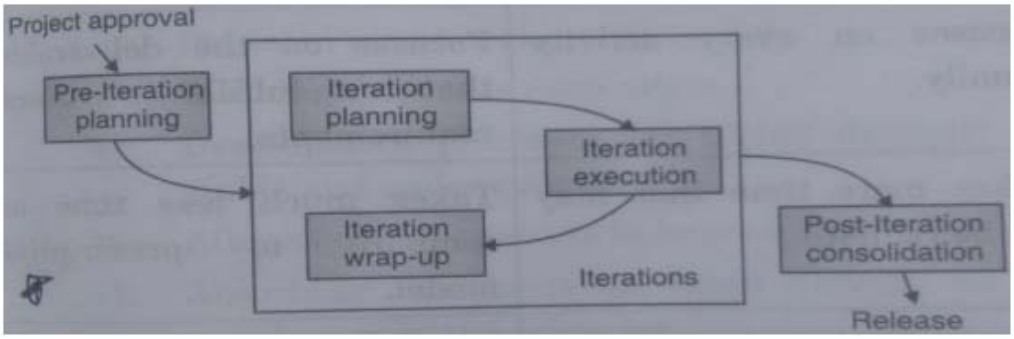
1) The answers should be examined by key words and not as word-to-word as given in the model answer scheme.
2) The model answer and the answer written by candidate may vary but the examiner may try to assess the understanding level of the candidate.
3) The language errors such as grammatical, spelling errors should not be given more Importance (Not applicable for subject English and Communication Skills.
4) While assessing figures, examiner may give credit for principal components indicated in the figure. The figures drawn by candidate and model answer may vary. The examiner may give credit for anyequivalent figure drawn.
5) Credits may be given step wise for numerical problems. In some cases, the assumed constant values may vary and there may be some difference in the candidate's answers and model answer.
6) In case of some questions credit may be given by judgement on part of examiner of relevant answer based on candidate's understanding.
7) For programming language papers, credit may be given to any other program based on equivalent concept.

Q. No.	Sub Q. N.	Answer	Marking Scheme
1	a	**Attempt any THREE of the following:**	12
	i	**Define Software Engineering and explain the reason what "Need of Software Engineering".**	4
	Ans	**Software Engineering:** The application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software; that is, the application of engineering to software. **Need of Software Engineering**: To produce reliable and trustworthy systems economically and quickly. The majority of costs are the costs of changing the software after it has gone into use.	Definition 2M; Need 2 M
	ii	**State the seven core principles of software engineering.**	4
	Ans	• The First Principle: The Reason It All Exists • The Second Principle: KISS (Keep It Simple, Stupid!) • The Third Principle: Maintain the Vision • The Fourth Principle: What You Produce, Others Will Consume • The Fifth Principle: Be Open to the Future • The Sixth Principle: Plan Ahead for Reuse • Seventh Principle: Think!	List of all 7 principles 4M
	iii	**Compare the difference between bottom-up integration and Top down Integration (four points)**	4
	Ans		

			Bottom-up integration	**Top down integration**		4 points of comparison 1M each
			Bottom up integration begins with sub modules and atomic checking	This is incremental approach begins with construction of the software architecture.		
			Low- level components are combined into clusters that perform a specific software sub function	Modules are integrated by moving downward through the control hierarchy, beginning with the main control module (main Program).		
			Drivers are required for test cases	Drivers are not required for test cases		
			Different clusters are formed for the testing	Depth-first integration integrates all components on a major control path of the program structure.		
	iv	**What is SQA? And define any four activities of SQA.**				**4**
	Ans	Software quality assurance is composed of a variety of tasks associated with two different constituencies - the software engineers who do technical work and an SQA group that has responsibility for quality assurance planning, oversight, record keeping, analysis, and reporting. Software engineers address quality (and perform quality assurance and quality control activities) by applying solid technical methods and measures, conducting formal technical reviews, and performing well-planned software testing. Activities of SQA: 1) **Prepare an SQA plan for a project:** The plan is developed during project planning and is reviewed by all interested parties. Quality assurance activities performed by the software engineering team and the SQA group are governed by the plan. The plan identifies • evaluations to be performed. • audits and reviews to be performed. • standards that are applicable to the project. • procedures for error reporting and tracking. • documents to be produced by the SQA group. • amount of feedback provided to the software project team. 2) **Participate in the development of the project's software process description:** The software team selects a process for the work to be performed. The SQA group reviews the process description for compliance with organizational policy, internal software standards, externally imposed standards (e.g., ISO-9001), and other parts of the software project plan. 3) **Review software engineering activities to verify compliance with the**				SQA definition 2M; Any 4 activities 2M; description of activities is optional

		defined software process: The SQA group identifies, documents, and tracks deviations from the process and verifies that corrections have been made. 4) **Audits designated software work products to verify compliance with those defined as part of the software process:** The SQA group reviews selected work products; identifies, documents, and tracks deviations; verifies that corrections have been made; and periodically reports the results of its work to the project manager. 5) **Ensure that deviations in software work and work products are documented and handled according to a documented procedure:** Deviations may be encountered in the project plan, process description, applicable standards, or technical work products. 6) **Records any noncompliance and reports to senior management:** Noncompliance items are tracked until they are resolved.	
	b	**Attempt any <u>ONE</u> of the following:**	**6**
	i	**Define role of the Repository. List the six functions and explain any two of them in detail.**	**6**
	Ans	**Role of The Repository:** The SCM repository is the set of mechanisms and data structures that allow software team to manage change in an effective manner. It provides the obvious functions of a modern database management system by ensuring data integrity, sharing, and integration. In addition, the SCM repository provides a hub for the integration of software tools, is central to the flow of the software process, and can enforce uniform structure and format for software engineering work products. **Functions** 1. Data integrity: Various entries in the SCM repository must be validated. This can be done by some data integrity function. 2. Information sharing: This function takes care of sharing of information among various objects, tools, multiple developers so that resources and data can be equally distributed. 3. Methodology enforcement: This function defines entity relationship model. In the sense, this function elaborates various objects and relationship among them in the repository. It also defines the steps that must be conducted to build the contents of repository. 4. Tool integration: In SCM repository many tools are used to access the data model. In order to control the access of data tool integration functionality is used. 5. Data integration: Using this function data present in databases is integrated.	Definition 2M; List of six functions 2 M; Description of any 2 – 2 M

		6. Document standardization: There are some important objects in the database using which software documents can be created. Using some standardization functions SCM repository documents can be maintained.	
	ii	**Enlist the five maturity levels of capability maturity model integration (CMMI) and explain it.**	**6**
	Ans	 **Level 1: Initial.** The software process is characterized as ad hoc and occasionally even chaotic. Few processes are defined, and success depends on individual effort. **Level 2: Repeatable.** Basic project management processes are established to track cost, schedule, and functionality. The necessary process discipline is in place to repeat earlier successes on projects with similar applications. **Level 3: Defined.** The software process for both management and engineering activities is documented, standardized, and integrated into an organization wide software process. All projects use a documented and approved version of the organization's process for developing and supporting software. This level includes all characteristics defined for level 2 **Level 4: Managed.** Detailed measures of the software process and product quality are collected. Both the software process and products are quantitatively understood and controlled using detailed measures. This level includes all characteristics defined for level 3 **Level 5: Optimizing.** Continuous process improvement is enabled by quantitative feedback from the process and from testing innovative ideas and technologies. This level includes all characteristics defined for level 4.	Diagram 2 M; Description 4M
2		**Attempt any <u>FOUR</u> of the following:**	**16**
	a	**Draw the diagram of Agile software development and give its drawbacks (four points).**	**4**

	Ans	 **Drawback of Agile model:** • In case of some software deliverables, especially the large ones, it is difficult to assess the effort required at the beginning of the software development life cycle. • There is lack of emphasis on necessary designing and documentation. • The project can easily get taken off track if the customer representative is not clear what final outcome that they want. • Only senior programmers are capable of taking the kind of decisions required during the development process. Hence it has no place for newbie programmers, unless combined with experienced resources.	Diagram 2M; four drawbacks 2M
	b	**Explain principles of communications which ensures smooth and proper communication among project participants.**	**4**
	Ans	**Principle 1. Listen.** Try to focus on the speaker's words, rather than formulating your response to those words. Ask for clarification if something is unclear, but avoid constant interruptions. Never become contentious in your words or actions (e.g., rolling your eyes or shaking your head) as a person is talking. **Principle 2. Prepare before you communicate.** Spend the time to understand the problem before you meet with others. If necessary, do some research to understand business domain. If you have responsibility for conducting a meeting, prepare an agenda in advance of the meeting. **Principle 3. Someone should facilitate the activity.** Every communication meeting should have a leader (a facilitator) to keep the conversation moving in a productive direction, to mediate any conflict that does occur, and to ensure that other principles are followed. **Principle 4. Face-to-face communication is best.** Face to face communication is always makes sense. It usually works better when some other representation of the relevant information is present. For example, a participant may create a drawing document that serves as a focus for discussion. **Principle 5. Take notes and document decisions.** Things have a way of falling into the cracks. Someone participating in the communication should serve as a "recorder" and write down all important points and decisions. **Principle 6. Strive for collaboration.** Collaboration occurs when the collective knowledge of members of the team is used to describe product or system functions or features. Each small collaboration serves to build trust	Any four principles with description 1M each

		among team members and creates a common goal for the team. **Principle 7. Stay focused;** modularize your discussion. The more people involved in any communication, the more likely that discussion will bounce from one topic to the next. The facilitator should keep the conversations modular; leaving one topic only after it has been resolved **Principle 8. If something is unclear, draw a picture:** Verbal communication goes only so far. A sketch or drawing can often provide clarity when words fail to do the job. **Principle 9. (a) Once you agree to something, move on. (b) If you can't agree to something, move on. (c) If a feature or function is unclear and cannot be clarified at the moment, move on.** Communication, like any software engineering activity, takes time. Rather than iterating endlessly, the people who participate should recognize that many topics require discussion and that "moving on" is sometimes the best way to achieve communication agility. **Principle 10. Negotiation is not a contest or a game. It works best when both parties win.** There are many instances in which you and other stakeholders must negotiate functions and features, priorities, and delivery dates. If the team has collaborated well, all parties have a common goal. Still, negotiation will demand compromise from all parties.	
	c	**Describe the following requirements engineering tasks:** **i. Specification** **ii. Requirement Management**	**4**
	Ans	i) Specification In the context of computer-based systems (and software), the term specification means different things to different people. A specification can be a written document, a set of graphical models, a formal mathematical model, a collection of usage scenarios, a prototype, or any combination of these. ii) Requirement Management Requirements for computer-based systems change, and the desire to change requirements persists throughout the life of the system. Requirements management is a set of activities that help the project team identify, control, and track requirements and changes to requirements at any time as the project proceeds. Many of these activities are identical to the software configuration management (SCM) techniques.	Specification 2M; Requirement Management 2M
	d	**List types of testing and explain it in brief.**	**4**
	Ans	**1. Unit Testing** It focuses on smallest unit of software design. In this we test an individual unit or group of inter related units. It is often done by programmer by using sample input and observing its corresponding outputs. **2. Integration Testing** The objective is to take unit tested components and build a program structure that has been dictated by design.Integration testing is testing in which a group of components are combined to produce output. **3. Regression Testing**	Any 4 types of testing with description 1M each

Every time new module is added leads to changes in program. This type of testing make sure that whole component works properly even after adding components to the complete program.

4. Smoke Testing

This test is done to make sure that software under testing is ready or stable for further testing. It is called smoke test as testing initial pass is done to check if it did not catch the fire or smoked in the initial switch on.

5. Alpha Testing

This is a type of validation testing. It is a type of *acceptance testing* which is done before the product is released to customers. It is typically done by QA people.

6. Beta Testing

The beta test is conducted at one or more customer sites by the end-user of the software. This version is released for the limited number of users for testing in real time environment

7. System Testing

In this software is tested such that it works fine for different operating system.It is covered under the black box testing technique. In this we just focus on required input and output without focusing on internal working.In this we have security testing, recovery testing, stress testing and performance testing

8. Stress Testing

In this we give unfavorable conditions to the system and check how they perform in those conditions.

9. Performance Testing

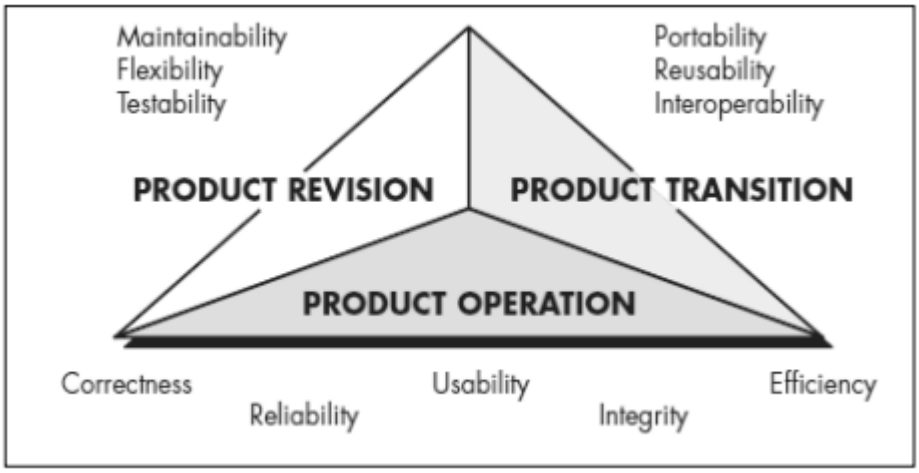
It is designed to test the run-time performance of software within the context of an integrated system.It is used to test speed and effectiveness of program.

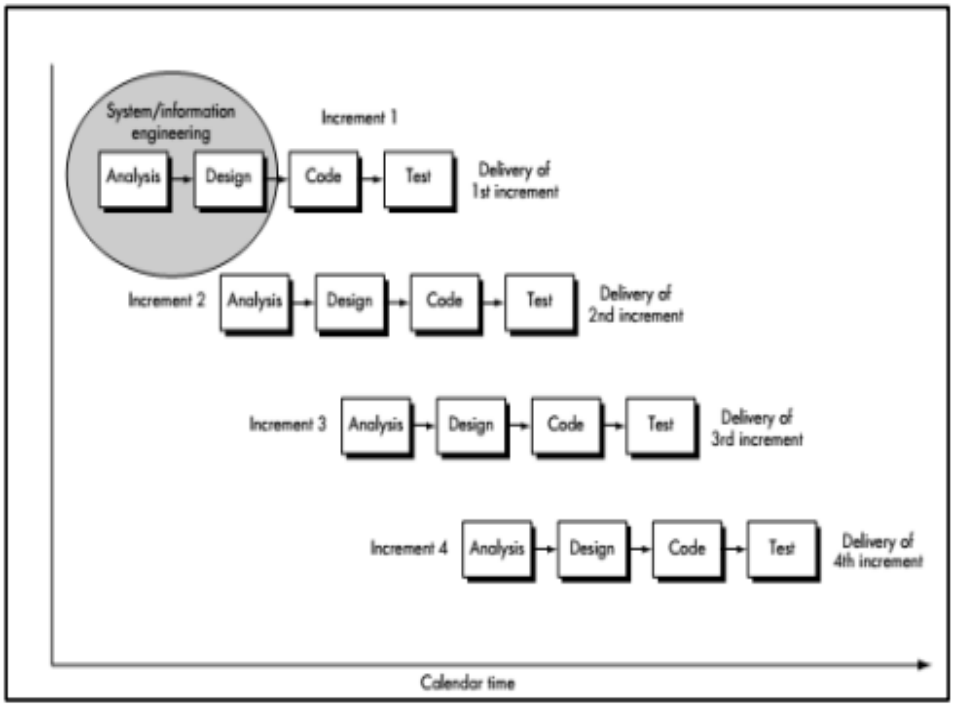
e **Give difference between version control and change control (Four points).** **4**

Ans

4 points of differences – 1M each

version control	change control
Version Control Combines procedures and tools to manage different version of configuration objects that are created during the software process	Change control is manual step in software lifecycle. It combines human procedures and automated tools.
A number of version control systems establish a set – a collection of all changes (to some baseline configuration) that are required to create a specific	The result of the evaluation are presented as a change report, which is used by the change control authority(CCA)

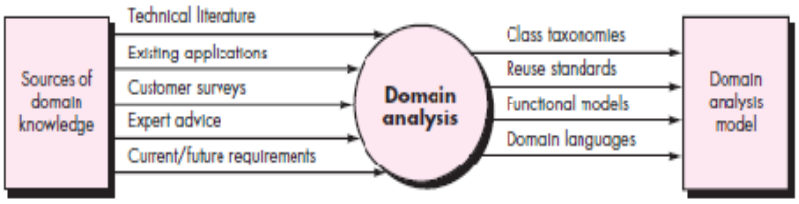
			version of the software.			
			"Changes set" captures all changes to all files in the configuration along with reason for changes and details of who made the changes and when.	An engineering change order (ECO) is generated for each approved change. The ECO describes the change order to be made, the constraints that must be respected, and the criteria for view and audit.		
			A number of named change set can be identified for an application or system. This enables a software engineer to construct a version of the software by specifying the changes set (by name) that must be applied to the baseline configuration.	The objects are then "checked in" to the database and appropriate version control mechanism is used to create the next version of the software.		
	f	**Explain 8 McCall's quality factor with the help of diagram.**				**4**
	Ans	 1) Correctness. The extent to which a program satisfies its specification and fulfills the customer's mission objectives. 2) Reliability. The extent to which a program can be expected to perform its intended function with required precision. 3) Efficiency. The amount of computing resources and code required by a program to perform its function. 4) Integrity. Extent to which access to software or data by unauthorized persons can be controlled. 5) Usability. Effort required to learn, operate, prepare input for, and				Any 8 quality factor ½ M each

		interpret output of a program 6) Maintainability. Effort required to locate and fix an error in a program. 7) Flexibility. Effort required to modify an operational program. 8) Testability. Effort required to test a program to ensure that it performs its intended function. 9) Portability. Effort required to transfer the program from one hardware and/or software system environment to another. 10) Reusability. Extent to which a program [or parts of a program] can be reused in other applications—related to the packaging and scope of the functions that the program performs 11) Interoperability. Effort required to couple one system to another.	
3		**Attempt any <u>FOUR</u> of the following:**	**16**
	a	**Draw the diagram of Incremental process model and give its demerits(four points)**	**4**
	Ans	**Incremental Process Model:**  **Demerits of Incremental Process Model**: i. Increments need be relatively small ii. Mapping requirements to increments may not be easy. iii. Common software facilities may be difficult to identify. iv. It requires a good planning designing. v. Rectifying a problem in one unit requires correction in all the units and consumes a lot of time.	Diagram-2 M, Any four Demerits-1/2 M each
	b	**Why good planning practice is needed for software development.**	**4**

	Ans	Any complicated journey can be simplified if a map exists. A Software project is a complicated journey, and the planning activity creates a “map” that helps guide the team as it makes the journey. The map is called a software project plan which defines the software engineering work by describing the technical tasks to be conducted, the risks that are likely, the resources that will be required, the work products to be produced, and a work schedule. The planning activity encompasses a set of management and technical practices that enable the software team to define a road map as it travels toward its strategic goal and tactical objectives.	Explanation-4M
	c	**Give difference between modality and cardinality (four points).**	**4**
	Ans		4 Points :1 M each **Note**: Any other relevant points shall be considered

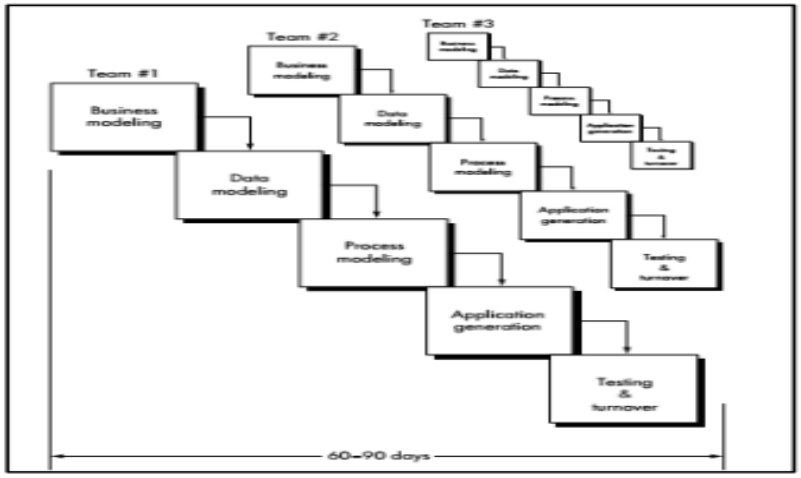
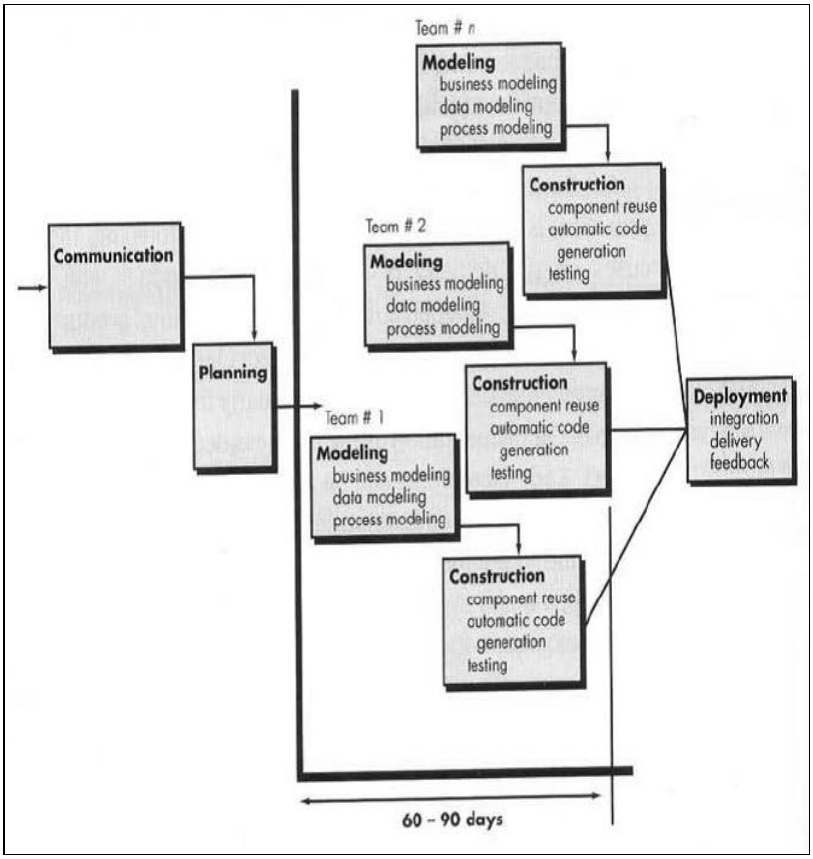
Modality	**Cardinality**
1. The Modality indicates whether or not a relationship between objects is mandatory	1.The cardinalityof an object-relationship pair specifies “the number of occurrences of one [object]that can be related to the number of occurrences of another [object]”
2.Expected values are 0 or 1	2.Expected values are 1:1,1:M,M:N
3. It provides indication of participation in the relationship by value 1. If no participation then value will be 0.	3. It does not provide an indication whether or not particular data object can participate in relationship.
4.It gives minimum number occurrences in relationship	4.It gives maximum number occurrences in relationship

	d	**Explain the situation where the condition alpha testing and beta testing is used in integration testing.**	**4**
	Ans	**Alpha Testing:** It is conducted by a team of highly skilled testers/customer/end user at development site. Alpha Testing is conducted in Control Environment as Developer is present. Error/Problem may be solved earlier as developer is present. **Beta Testing:** It is always conducted in Real Time environment by customers or end users at their own site. Beta Testing is conducted in Uncontrolled Environment as Developer is absent. Therefore, the beta test is a "live" application of the software in an environment that cannot be controlled by the	Alpha testing:2M, Beta Testing: 2 M

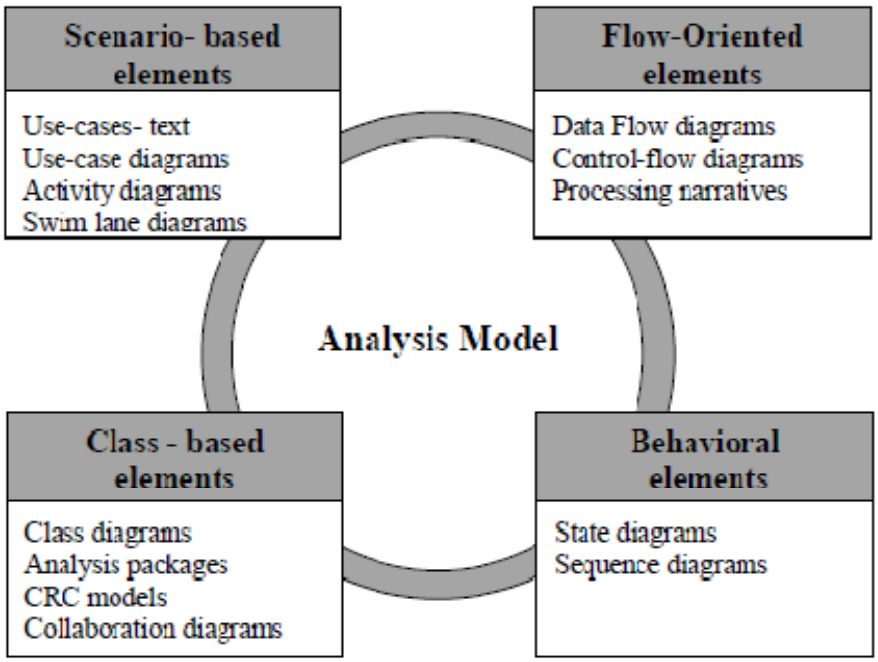
		developer. The customer records all problems (real or imagined) that are encountered during beta testing and reports these to the developer at regular intervals. As a result of problems reported during beta tests, software engineers make modifications and then prepare for release of the software product to the entire customer base.	
	e	**Why software configuration management (SCM) is needed.**	**4**
	Ans	**Need of SCM :** 1. To Identify all items that define the software configuration 2. To Manage changes to one or more configuration items 3. To Facilitate construction of different versions of a software application 4. To ensure that software quality is maintained as configuration evolves. 5. Software Configuration Management (SCM) is a set of activities designed to manage change by identifying the work products that are likely to change, establishing relationships among them, defining mechanisms for managing different versions of these work products, controlling the changes imposed and auditing and reporting on the changes made. 6. SCM is an umbrella activity that is applied throughout the software process. 7. SCM helps to improve software quality and on time delivery. 8. SCM defines the project strategy for change management. When formal SCM is invoked, the change control process produces software change requests, reports and engineering change orders. 9. SCM helps to track, analyze and control every work product.	Any four Points: 1M each
4	**a**	**Attempt any <u>THREE</u> of the following:**	**12**
	i	**List the three domains of analysis model and explain characteristics of analysis modeling.**	**4**
	Ans	 Domains of analysis model: Sources of domain knowledge: 1.Technical literature 2. Existing applications 3. Customer surveys 4.Expert advice	Any three domais:1 M, Any three characteristics: 3 M Note: Any other relevant answer shall be considered

		5. Current/future requirements Outcome of domain analysis : 1. Class taxonomies 2. Reuse standards 3. Functional and behavioral models 4.Domain languages Characteristics of analysis modeling: (1) Mechanism for information domain analysis (2) Approach for functional and/or behavioral representations (3) Definition of interfaces (4) Mechanisms for problem partitioning (5) Support for abstraction (6) Representation of essential and implementation views.	
	ii	**Explain "why it is necessary to design quality guidelines".**	**4**
	Ans	Software Quality guidelines encompasses the entire software development life cycle and the goal is to ensure that the development and maintenance processes are continuously improved to produce products that meet specifications. Note that the scope of Quality is NOT limited to just Software Testing. For example, how well the requirements are stated and managed matters a lot.Once the processes have been defined and implemented, Quality Assurance has responsibility of identifying weaknesses in the processes and correcting those weaknesses to continually improve the processes. OR Guidelines: 1. A design should exhibit an architecture that (a) has been created using recognizable architectural styles or patterns, (b) is composed of components that exhibit good design characteristics and (c) Can be implemented in an evolutionary fashion, thereby facilitating implementation and testing. 2. A design should be modular; that is, the software should be logically partitioned into elements or subsystems. 3. A design should contain distinct representations of data, architecture, Interfaces and components. 4. A design should lead to data structures that are appropriate for the classes to be implemented and are drawn from recognizable data patterns. 5. A design should lead to components that exhibit independent functional Characteristics. 6. A design should lead to interfaces that reduce the complexity of connections between components and with the external environment. 7. A design should be derived using a repeatable method that is driven by information obtained during software requirements analysis.	Explanation: 4 M Note: Any other relevant answer shall be considered

		8. A design should be represented using a notation that effectively communicates its meaning.	
	iii	**Explain "why black box testing is better than white box testing" software.**	**4**
	Ans	**Black box testing**: It is the Software testing method which is used to test the software without knowing the internal structure of code or program. **White box testing**: It is the software testing method in which internal structure is being known to tester who is going to test the software. **Why black box testing is better than white box testing:** 1. Programming Knowledge is not required to carry out Black Box Testing but Programming knowledge is required to carry out White Box Testing. 2. Black Box Testing is applicable on higher levels of testing like System Testing, Acceptance testing. White Box Testing is applicable on lower level of testing like Unit Testing, Integration testing. 3. In Black box testing, the tester gives input and checks the output.In White Box Testing Structural testing, Logic testing, Path testing, Loop testing, Code coverage testing is carried out so the tester should have programming knowledge. 4. That's why Black Box Testing is better than White Box Testing as it gives idea whether user/customer will accept the product or not without having knowledge of programming language.	Black box/White Box Testing Definition-1M, Explanation-3M
	iv	**Describe DMAIC approach with respect to six sigma.**	**4**
	Ans	**Six Sigma:** It is the most widely developed by Motorola in 1980, used strategy for statistical quality assurance in industry today. Six Sigma strategy "is a rigorous and disciplined methodology that uses data and statistical analysis to measure and improve a company's operational performance by identifying and eliminating defects' in manufacturing and service-related processes" **DMAIC:** The DMAIC project methodology has five phases: ***Define*** customer requirements and deliverables and project goals via well-defined methods of customer communication. • ***Measure*** the existing process and its output to determine current quality performance (collect defect metrics). • ***Analyze*** defect metrics and determine the vital few causes. • ***Improve*** the process by eliminating the root causes of defects. • ***Control*** the process to ensure that future work does not reintroduce the causes of defects.	Explanation of DMAIC:4 M
	b	**Attempt any <u>ONE</u> of the following:**	**6**
	i	**Draw a neat labeled diagram of RAD model and explain how to overcome the drawbacks of RAD model.**	**6**
	Ans	RAD Model	Diagram:2 M,

		 **OR**  **Drawbacks of RAD model:** 1. RAD needs enough human resources to create the required number of RAD teams. 2. If developers and customers are not committed to the rapid	Drawbacks: 2M, How to overcome:2 M

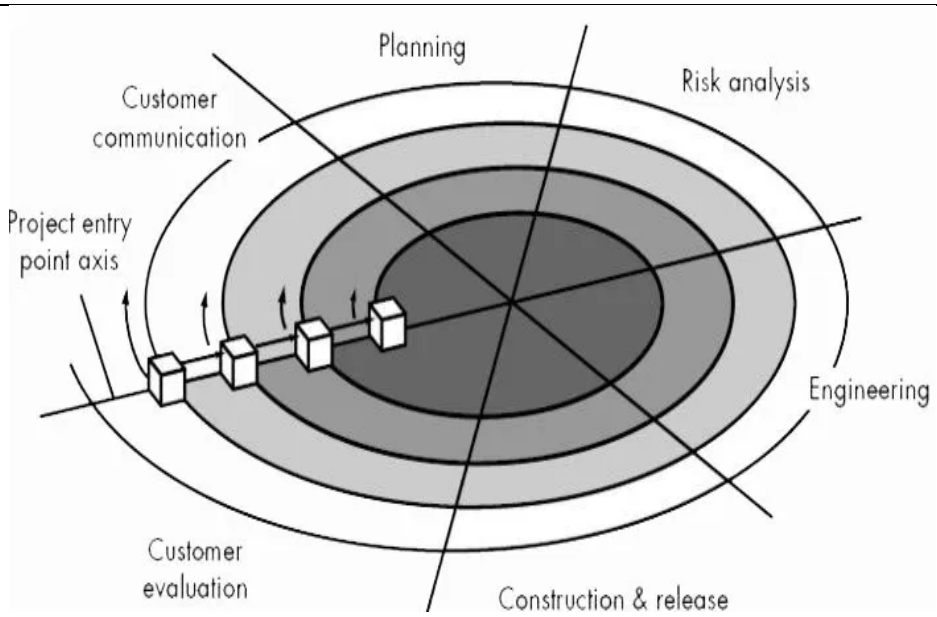
		model, the RAD project fails. 3. Rapid-fire activities need to be completed in very short or small time frame. Time is the major constraint in RAD. 4. RAD has to be modularized in a proper way otherwise creates a lots of confusions and problems. 5. In case of high performance requirement, RAD cannot be ideal model. **How to overcome the drawbacks:** The RAD model must be deployed by those that are familiar with its methodology. They must be capable of maximizing the advantages and overcoming the disadvantages. Sufficient research and a focused work ethic are fundamental to success. Only with precise knowledge and dedicated follow-ups can the RAD model yield optimum performance. RAD model has to be used by tuning between interface and system components.	
	ii	**List and draw the neat labeled diagram of elements of analysis model and explain it in detail.**	**6**
	Ans	**List of elements of analysis model:** **1. Flow-oriented modeling** **2. Scenario-based modeling** **3. Class-based modeling** **4. Behavioral modeling**	List: 1Mark, Diagram:2 M, Explanation:3M



Elements of the analysis model

- **Flow-oriented modeling** – provides an indication of how data objects are transformed by a set of processing functions
- **Scenario-based modeling** – represents the system from the user's point of view
- **Class-based modeling** – defines objects, attributes, and relationships
- **Behavioral modeling** – depicts the states of the classes and the impact of events on these states

5		**Attempt any TWO of the following:**	**16**
	a	**With the help of neat labelled diagram explain spiral process model and give the solution how to overcome the drawbacks of spiral process model.**	**8**

	Ans	 A spiral model is a realistic approach to the development of large-scale software products. It is combination of a waterfall model and iterative model. Each phase in spiral model begins with a design goal and ends with the client reviewing the progress. The spiral model was first mentioned by Barry Boehm in his 1986 paper. The development team in Spiral-SDLC model starts with a small set of requirement and goes through each development phase for those set of requirements. The software engineering team adds functionality for the additional requirement in every-increasing spirals until the application is ready for the production phase. **Spiral Model's phases are:** ▪ Communication ▪ Planning phase ▪ Risk analysis phase ▪ Engineering phase ▪ Construction and release phase ▪ Evaluation Phase **Communication** This phase is meant to understand the exact requirements of the customer and to document them properly. The main aim of feasibility study is to determine whether it would be financially and technically feasible to develop the product.	Digram -2M, Explanation - 2M and 4M for solution

Planning

In planning phase, gathering the requirement and analysis on it is done. It includes creating a schedule of events across time, manpower and other resources.

Risk analysis

The risk analysis phase focuses on identifying the risk and alternate solutions, trying to find out technical as well as managing risk.

Engineering Phase
This phase includes the development of the work product. The deliverables for the engineering phase will be source code, design documents, test cases, test summary, defect report etc.

Construction and Release Phase

It includes constructing and delivering the software to the user along with the user manual, instruction material and providing support to them.

Evaluation

Customer's involvement takes place in this Evaluation phase. Customer evaluates the work product and ensures that product meets all requirements if any changes required to the customer in the product, again all phases will be repeated. It is important to get feedback from the customer before releasing the product.

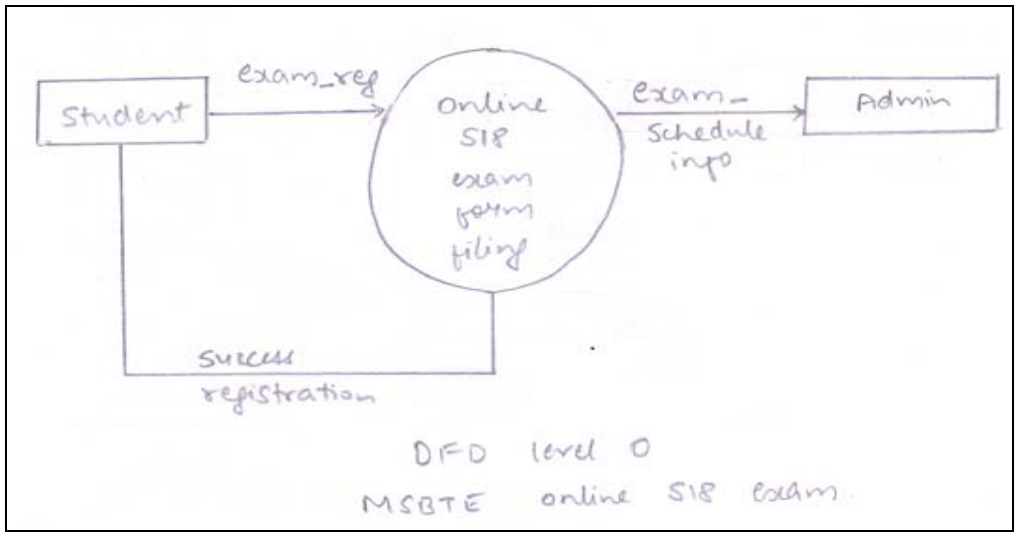
Drawbacks and the ways to overcome the drawbacks are listed below:-

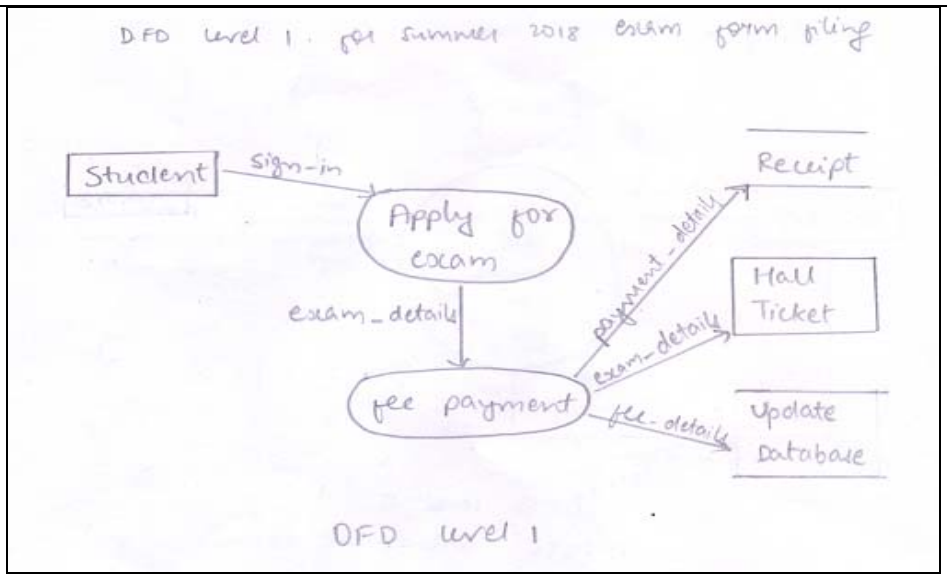
- It works best for large projects only as it also demands risk assessment expertise.

  Solution- Spiral model can be implemented for small projects due to its risk analysis feature. Risk analysis can be carried out individually and by the team by carefully studying the customer requirement, project feasibility and possible risks in market conditions.

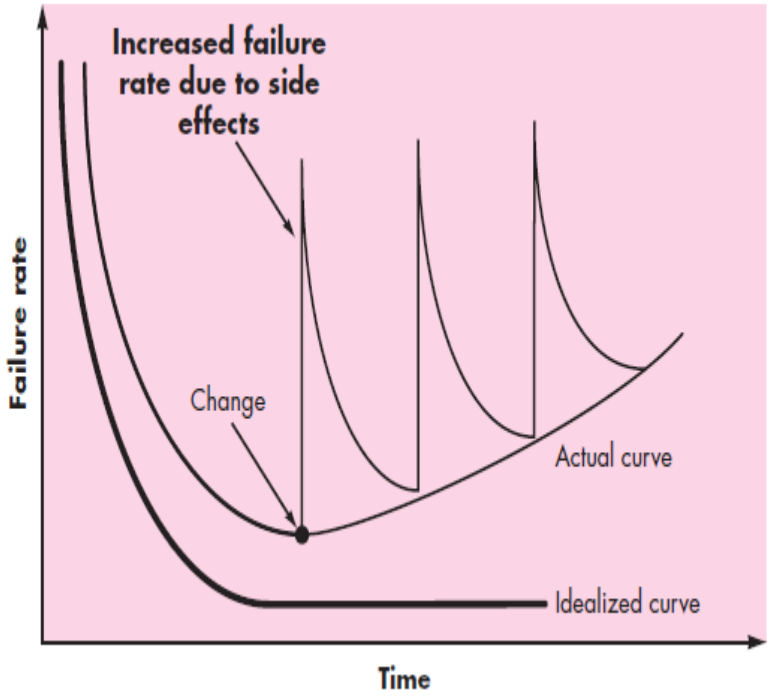
- Project cost could be infinite because of the spiral feature.

  Solution- Project cost can be substantiated if risk analysis is carried out by the team member instead of an expert.

		• Each spiral requires specific expertise, which makes the management process more complex. Solution-The complexity of the process can be reduced by breaking the process into individual models that designers work on. Every iteration has its own testing phase. • It is not suitable for low risk projects. Solution- The quality of low risk projects can be improved by using spiral model. Risk analysis can be carried out at a broader stage rather than in-depth experimentation.	
	b	**Draw the data flow diagram for MSBTE online s18 exam form filing considering level 0 and 1.**	**8**
	Ans	 **Level 0 DFD MSBTE online s18 exam form filling**	For DFD level 0 4M, For DFD level1 4M

		 **Level 1 DFD MSBTE online s18 exam form filling**	
	c	**Explain the following terms:** **i) Reactive risk strategy** **ii) Proactive risk strategy**	**8**
	Ans	**i) Reactive risk strategy** • Reactive risk strategy follows that the risks have to be tackled at the time of their occurrence. • No precautions are to be taken as per this strategy. • They are meant for risks with relatively smaller impact. • More commonly, the software team does nothing about risks until something goes wrong. • Then, the team flies into action in an attempt to correct the problem rapidly. This is often called a fire-fighting mode. The reactive risk management is an essential element of: • Mitigating safety events after hazard has occurred; • Minimizing damage from critical safety situations; • Acting quickly and efficiently in response to undesirable incidents; and • High quality decision making in reaction to safety data (threats, risk, etc.). **ii)Proactive risk strategy** It follows that the risks have to be identified before start of the project.	For Explanation 4M For Explanation 4M

		• They have to be analyzed by assessing their probability of occurrence, their impact after occurrence, and steps to be followed for its precaution. • They are meant for risks with relatively higher impact. The primary goals of proactive risk management are: • Identify behaviors that lead to hazard occurrence, and stop it before it happens; • Identify root causes before they lead to hazard occurrence; and • Understand safety "inputs" of your program – i.e., underlying causes that lead to safety performance. Being able to practice proactive risk management generally requires: • A great deal of safety data; • The ability to monitor complex safety metrics; and • A mature safety culture.	
6		**Attempt any <u>FOUR</u> of the following:**	**16**
	a	**List the characteristics of software and explain any two of them in detail.**	**4**
	Ans	Characteristics of software: i) **Software is developed or engineered**; it is not manufactured in the classical sense. Software is virtual. That is, software can be used using proper hardware. And we can only use it. But we can use, touch, and see hardware. Thus software never gets manufactured, they are developed. ii) **Software doesn't "wear out"** like hardware and it is not degradable over a period. Software is not susceptible to the environmental maladies that cause hardware to wear out. In theory, therefore, the failure rate curve for software should take the form of the "idealized curve" shown in Figure below	Listing of characteristic 2M – Explanation 2M

		 During itslife, software will undergo change. As changes are made, it is likely that errors will be introduced, causingthe failure rate curve to spike as shown in the "actual curve" .Before the curve can return to the originalsteady-state failure rate, another change is requested, causing the curve to spike again. Slowly, theminimum failure rate level begins to rise—the software is deteriorating due to change. iii) **Although the industry is moving toward component-based construction, most** **Software continues to be custom built.** Custom software or application is a kind of software, which is specifically design and develops for an organization or a group of users with unique needs and requirements. Most of the organizations are opting for custom built applications for its unique benefits.	
	b	**List the elements of design model and explain the characteristics of design model.**	**4**
	Ans	Following are the elements of design model are as following: **1. Data design elements** **2. Architectural design elements** **3. Interface design elements** **4.Component level diagram elements** **5. Deployment level design elements** **The characteristics of design model are** The design must implement all of the explicit requirements contained in the	Listing of elements 2M Explanation of characteristics 2M

		requirements model, and it must accommodate all of the implicit requirements desired by stakeholders. • The design must be a readable, understandable guide for those who generate code and for those who test and subsequently support the software. • The design should provide a complete picture of the software, addressing the data, functional, and behavioral domains from an implementation perspective.	
	c	**Explain** **(i)Patterns** **(ii)Functional independence w.r.t data oriented design**	**4**
	Ans	**(i)Patterns** A pattern provides a description of the solution to a recurring design problem of some specific domain in such a way that the solution can be used again and again. The objective of each pattern is to provide an insight to a designer who can determine the following. Whether the pattern can be reused Whether the pattern is applicable to the current project Whether the pattern can be used to develop a similar but functionally or structurally different design pattern. ***Types of Design Patterns*** Software engineer can use the design pattern during the entire software design process. When the analysis model is developed, the designer can examine the problem description at different levels of abstraction to determine whether it complies with one or more of the following types of design patterns. **Architectural pattern**. **Design patterns** **Idioms** **(ii)Functional independence w.r.t data oriented design** • Functional independence is achieved by developing modules with "single-minded" function and an "aversion" to excessive interaction with other modules. • Stated in other way, we want to design software so that each module addresses a specific sub function of requirements and has a simple interface when viewed from other part of program structure. • Independence is important because, software with effective modularity I.e. independent modules is easier to develop because function may be	Pattern explanation-2M

		compartmentalized and interfaces are simplified. • Independent modules are easier to maintain because secondary effects caused by design or code modification are limited, error progression are reduced, and reusable modules are possible • Functional independence is a key to good design, and design is the key to software quality. • Independence is assessed using two qualitative criteria 1. Cohesion A cohesive module perform a single task, requiring little interaction with other components in other parts of program 2. Coupling Coupling is an indication of the relative interdependence among modules.	Functional Independence explanation-2M
	d	**Define smoke testing, list characteristics of it and give its demerits (two points)**	**4**
	Ans	Smoke testing is integration testing approach that is commonly used for as a pacing mechanism for time critical projects, allowing the software team to assess its projects on a frequent basis. Characteristics of Smoke testing are:- _ Software components that have been translated into code are integrated into a "build." • A build includes all data files, libraries, reusable modules, and engineered components that are required to implement one or more product functions. _ A series of tests is designed to expose errors that will keep the build from properly performing its function. • The intent should be to uncover "show stopper" errors that have the highest likelihood of throwing the software project behind schedule. _ The build is integrated with other builds and the entire product (in its current form) is smoke tested daily. • The integration approach may be top down or bottom up. Demerits of Smoke testing are • Smoke testing does not cover the detailed testing. • It's a non-exhaustive testing with small number of test cases because of which we not are able to find the other critical issues. • Smoke testing is **not performed with negative scenarios** and with invalid data.	Definition, 1M–Listing of characteristics, 2M– Demerits
	e	**List and explain the features of Software Configuration Management (SCM)**	**4**

	Ans	The features of Software Configuration Management (SCM) are a.**Versioning:** As a project progresses, many versions of individual work products will be created. The repository must be able to save all of these versions to enable effective management of product releases and to permit developers to go back to previous versions during testing and debugging. b.**Dependency tracking and change management**: The repository manages a wide variety of relationships among the data elements stored in it. These include relationships between enterprise entities and processes, among the parts of an application design, between design components and the enterprise information architecture, between design elements and deliverables, and so on. c.**Requirements tracing**: This special function depends on link management and provides the ability to track all the design and construction components and deliverables that result from a specific requirements specification (forward tracing). In addition, it provides the ability to identify which requirement generated any given work product (backward tracing). d.**Configuration management:** A configuration management facility keeps track of a series of configurations representing specific project milestones or production releases. e.**Audit trails:** An audit trail establishes additional information about when, why, and by whom changes are made. Information about the source of changes can be entered as attributes of specific objects in the repository.	For listing 2M For Explanation 2M