SUMMER – 2022 EXAMINATION
MODEL ANSWER

Subject: Software Engineering **Subject Code: 22413**

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 any equivalent 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.
8) As per the policy decision of Maharashtra State Government, teaching in English/Marathi and Bilingual (English + Marathi) medium is introduced at first year of AICTE diploma Programme from academic year 2021-2022. Hence if the students in first year (first and second semesters) write answers in Marathi or bilingual language (English +Marathi), the Examiner shall consider the same and assess the answer based on matching of concepts with model answer.

Q. No	Sub Q.N.	Answer	Marking Scheme
1.	**a) Ans.**	**Attempt any FIVE of the following:** **List any four types of software** • System software • Application Software • Scientific software • Embedded software • Product line software • Web application • Artificial Intelligence	**10** **2M** ***1/2M each, any four types***
	b) Ans.	**List any four planning principles** 1.Understanding the scope of the project 2. Involve stakeholders in the planning activity 3.Planning is iterative 4.Planning should be based on the information available 5. Consider the risk as the plan is defined	**2M** ***1/2M each, any four principles***

		6. Being realistic 7. Adjust granularity as the plan is defined 8. Define how to ensure quality 9. Describe how to accommodate change 10. Track and monitor the plan frequently and make adjustments if required	
	c) **Ans.**	**Describe following design concepts** **i) Abstraction** **ii) Information hiding** **Abstraction** Abstraction is hiding the internal implementation and highlight the set of services. It is achieved by using the abstract class and interfaces and further implementing the same. **Information Hiding** It 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 if the design decision is changed.	**2M** ***1M for each design concept***
	d) **Ans.**	**List 4P's of Management spectrum** People Product Process Project	**2M** ***1/2M each***
	e) **Ans.**	**Define Quality control and Quality Assurance** **Quality Control**: Software quality control is the set of procedures used by organizations to ensure that a software product meets its quality goals at the best value to the customer, and to continually improve the organization's ability to produce software products in the future **Quality Assurance**: Conformance to explicit stated functional and performance requirements, explicitly documented. It is also the development of standards and implicit characteristic that are expected of all professionally developed software.	**2M** ***1M for each definition***

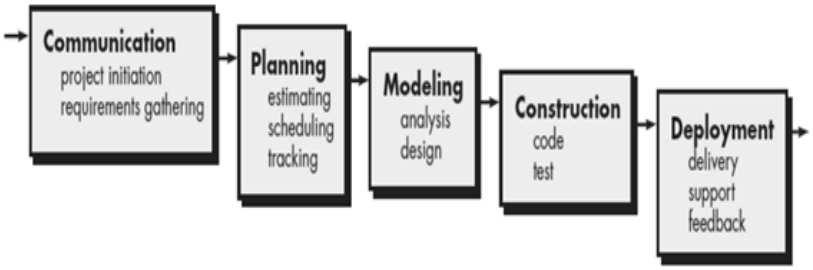
SUMMER – 2022 EXAMINATION
MODEL ANSWER

Subject: Software Engineering **Subject Code: 22413**

	f) Ans.	**List any four selection criteria for Software Process Model** Following are the parameters which is used to select 1. Requirements Characteristics • Reliability of Requirements • How often the requirements can change • Types of requirements • Number of requirements • Can the requirements be defined at an early stage • Requirements indicate the complexity of the system 2. Development team : • Team size • Experience of developers on similar type of projects • Level of understanding of user requirements by the developers • Environment • Domain knowledge of developers • Experience on technologies to be used • Availability of training 3. User involvement in the project : • Expertise of user in project • Involvement of user in all phases of the project • Experience of user in similar project in the past 4. Project type and associated risk : • Stability of funds • Tightness of project schedule • Availability of resources • Type of project • Size of the project • Expected duration for the completion of project • Complexity of the project • Level and the type of associated risk	**2M** ***1/2M each, any four criterias***
	g) Ans.	**Define Project Cost Estimation.** Software cost estimation is the process of predicting the effort required to develop a software system. Project cost estimating is the process of predicting the total cost of the tasks, time, and resources required to deliver a project's scope of work. There are three approaches of project estimation, they are: i) Heuristic ii) Analytical iii) Empirical	**2M** ***2M for correct definition***

SUMMER – 2022 EXAMINATION
MODEL ANSWER

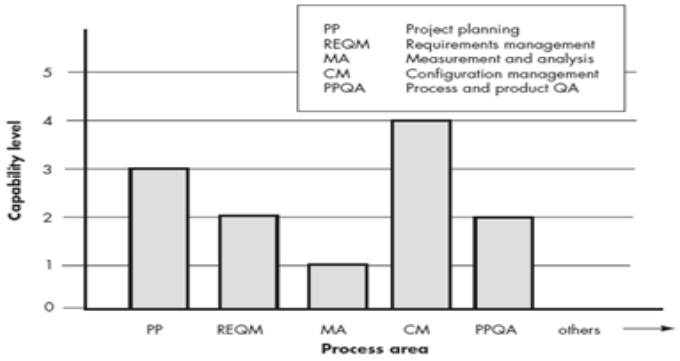
Subject: Software Engineering **Subject Code: 22413**

2.	**a) Ans.**	**Attempt any THREE of the following:** **Explain Waterfall Model with neat labeled diagram.** The Waterfall Model: The waterfall model is a traditional method, sometimes called the classic life cycle. This is one of the initial models. As the figure implies stages are cascaded and shall be developed one after the other. In other words one stage should be completed before the other begins. Hence, when all the requirements are elicited by the customer, analyzed for completeness and consistency, documented as per requirements, the development and design activities commence.  There are times when the requirements of a problem are reasonably well understood – when work flows from communication through deployment in a reasonably linear fashion. The waterfall model is a traditional method, sometimes called the classic life cycle. This is one of the initial models. As the figure implies stages are cascaded and shall be developed one after the other. In other words one stage should be completed before the other begins. Hence, when all the requirements are elicited by the customer, analyzed for completeness and consistency, documented as per requirements, the development and design activities commence. One of the main needs of this model is the user's explicit prescription of complete requirements at the start of development. For developers it is useful to layout what they need to do at the initial stages. Its simplicity makes it easy to explain to customers who may not be aware of software development process. It makes explicit with intermediate products to begin at every stage of development. One of the biggest limitation is it does not reflect the way code is really developed.	**12** **4M** ***2M for diagram*** ***2M for explanation***

		Problem is well understood but software is developed with great deal of iteration. Often this is a solution to a problem which was not solved earlier and hence software developers shall have extensive experience to develop such application; as neither the user nor the developers are aware of the key factors affecting the desired outcome and the time needed. Hence at times the software development process may remain uncontrolled. Today software work is fast paced and subject to a never-ending stream of changes in features, functions and information content. Waterfall model is inappropriate for such work. This model is useful in situation where the requirements are fixed and work proceeds to completion in a linear manner. Among the problems that are sometimes encountered when the waterfall model is applied are 1. Real projects rarely follow the sequential flow that the model proposes. Although the linear model can accommodate iteration, it does so directly. As a result, changes can cause confusion as the project team proceeds. 2. It is often difficult for the customer to state all requirements explicitly. The Waterfall Model requires this and has difficulty accommodating the natural uncertainty that exists at the beginning of many projects. 3. The customer must have patience. A working version of the program will not be available until late in the project time-span. A major blunder, if undetected until the working program is received, can be disastrous. The waterfall model is often inappropriate for such work. However, it can serve as a useful process model in situations where requirements are fixed and work is to proceed to completion in a linear manner.	

	b) Ans.	**State and describe any four core principles.** The core principles are 1. The reason it all exists: The software system exists in the organization for providing value to its users with, the availability of hardware and software requirements. Hence all the decisions should be made by keeping this in mind. 2. Keep it Simple, Stupid (KISS) Software design is not a haphazard process. There are many factors considered in the design effort. The design should be straight forward and as simple as possible. This facilitates having a system which can be easily understood and easy to maintain. Simple doesn't mean quick and dirty. In fact, it requires lot of thought and effort to simplify multiple iterations of a complex task. This results in the advantage that the software is less error prone and easily maintainable. 3. Maintain the vision A clear vision is essential for the success of a software project. If the vision is missing, the project may end up of two or more minds. The team leader has a critical role to play for maintaining the vision and enforce compliance with the help of the team members. 4. What you produce, others will consume The design and implementation should be done by keeping in mind the user's requirements. The code should permit the system extension. Some other programmers debugging the code should not have any errors and satisfying all the user needs. 5. Be open to future The system with the long lifetime has more value. The industry standard software systems induce for longer. The system should be ready to accept and adapt to new changes. The systems which are designed by keeping in mind the future needs will be more successful and acceptable to the users. 6. Plan ahead for reuse Reuse saves time and efforts. The reuse of code and design is one of the advantages of object oriented technologies. The reuse of parts of the code helps in reducing the cost and time evolved, in the new software development. 7. Think Placing clear and complete thought before action almost always produces better results. With proper thinking, we are most likely to do	**4M** ***2M for stating 2M for description***

		it right. We also gain knowledge about how to do it right again. It becomes a valuable experience, even if something goes wrong, as there was adequate thought process. Hence when clear thought has gone into the system, value comes out, this provides potential rewards.	
	c) **Ans.**	**Explain Test Documentation with the help of following terms** **i) Test Case** **ii) Test Data** **iii) Test Plan** **Test Documentation** Test documentation is documentation of artifacts created before or during the testing of software. It helps the testing team to estimate testing effort needed, test coverage, resource tracking, execution progress, etc. It is a complete suite of documents that allows you to describe and document test planning, test design, test execution, test results that are drawn from the testing activity **Test Case** It is a detailed document that describes step by step procedure to test an application. It consists of the complete navigation steps and inputs and all the scenarios that need to be tested for the application. We will write the test case to maintain the consistency, or every tester will follow the same approach for organizing the test document. It is a document that is prepared by the managers or test lead. **Test Data** Data created or selected to satisfy the execution preconditions and inputs to execute one or more test cases **Test Plan** It consists of all information about the testing activities. The test plan consists of multiple components such as Objectives, Scope, Approach, Test Environments, Test methodology, Template, Role & Responsibility, Effort estimation, Entry and Exit criteria, Schedule, Tools, Defect tracking, Test Deliverable, Assumption, Risk, and Mitigation Plan or Contingency Plan.	**4M** ***1M for each***
	d) **Ans.**	**Explain CMMI in detail with neat diagram** The Capability Maturity Model Integration (CMMI), a comprehensive process meta-model that is predicated on a set of system and software engineering capabilities that should be present as organizations reach different levels of process capability and	**4M** ***3M for explanation***

		maturity. The CMMI represents a process meta-model in two different ways: (1) As a "continuous" model and (2) as a "staged" model. The continuous CMMI meta- model describes a process in two dimensions. Each process area (e.g., project planning or requirements management) is formally assessed against specific goals and practices and is rated according to the following capability levels: Level 0: Incomplete—the process area (e.g., requirements management) is either not performed or does not achieve all goals and objectives defined by the CMMI for level 1 capability for the process area. Level 1: Performed—all of the specific goals of the process area (as defined by the CMMI) have been satisfied. Work tasks required to produce defined work products are being conducted. **CMMI Process Area Capability Profile.**  	***1M for diagram***

OR

Level	Focus	Process Areas
Optimizing	*Continuous process improvement*	Organizational innovation and deployment Causal analysis and resolution
Quantitatively managed	*Quantitative management*	Organizational process performance Quantitative project management
Defined	*Process standardization*	Requirements development Technical solution Product integration Verification Validation Organizational process focus Organizational process definition Organizational training Integrated project management Integrated supplier management Risk management Decision analysis and resolution Organizational environment for integration Integrated teaming
Managed	*Basic project management*	Requirements management Project planning Project monitoring and control Supplier agreement management Measurement and analysis Process and product quality assurance Configuration management
Performed		

Level 2: Managed—all capability level 1 criteria have been satisfied. In addition, all work associated with the process area conforms to an organizationally defined policy; all people doing the work have access to adequate resources to get the job done; stakeholders are actively involved in the process area as required; all work tasks and work products are "monitored, controlled, and reviewed; and are evaluated for adherence to the process description".

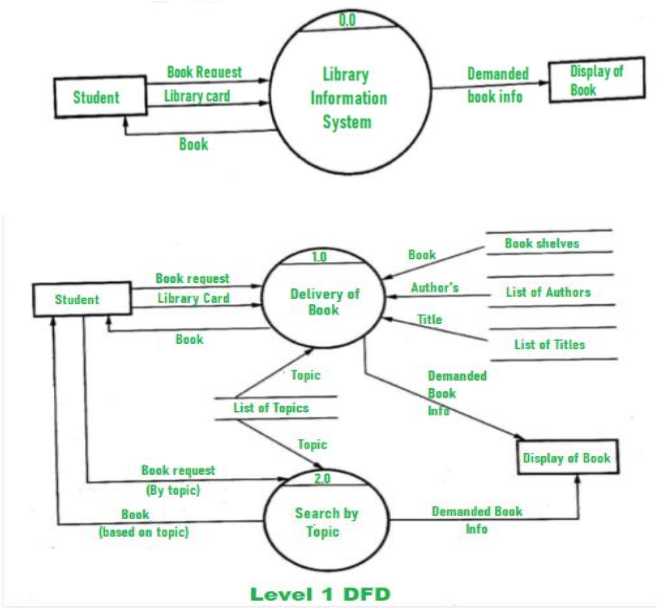
Level 3: Defined—all capability level 2 criteria have been achieved. In addition, the process is "tailored from the organization's set of standard processes according to the organization's tailoring guidelines, and contributes work products, measures, and other process-improvement information to the organizational process assets".

Level 4: Quantitatively managed—all capability level 3 criteria have been achieved. In addition, the process area is controlled and improved using measurement and quantitative assessment. "Quantitative objectives for quality and process performance are established and used as criteria in managing the process".

		Level 5: Optimized—all capability level 4 criteria have been achieved. In addition, the process area is adapted and optimized using quantitative (statistical) means to meet changing customer needs and to continually improve the efficacy of the process area under consideration.	
3.	**a) Ans.**	**Attempt any THREE of the following:** **State and describe any four deployment principles** **Principle 1: Manage customer's expectations.** It always happens that customer wants more than he has stated earlier as his requirements. It may be the case that customer gets disappointed, even after getting all his requirements satisfied. Hence at time of delivery developer must have skills to manage customer's expectations. **Principle 2: Assembly and test complete delivery package.** It is not the case that the deliverable package is ‗only software'. The customer must get all supporting and essential help from developer's side. **Principle 3: Record-keeping mechanism must be established for customer support.** Customer support is important factor in deployment phase. If proper support is not provided, customer will not be satisfied. Hence support should be well planned and with record-keeping mechanism. **Principle 4: Provide essential instructions, documentations and manual**. Many times, developer thinks ―when project is successful deliverable part is only working program‖. But realty is that working program is just part of software product. Actual project delivery includes all documentations, help files and guidance for handling the software by user. **Principle 5: Don't deliver any defective or buggy software to the customer.** In incremental type of software, software organizations may deliver some defective software to the customer by giving assurance that the defects will be removed in next increment.	**12** **4M** ***1M for each principle***

SUMMER – 2022 EXAMINATION
MODEL ANSWER

Subject: Software Engineering **Subject Code: 22413**

	b) **Ans.**	**Draw DFD 0 and DFD 1 diagram for Library Management System.**  **Level 1 DFD**	**4M** ***2M for DFD 0*** ***2M for DFD 1***
	c) **Ans.**	**State and describe two metrics of project size estimation** **Metrics for project Size Estimation** 1.Line of Code 2. Function Point **Lines of Code (LOC)** LOC is the simplest among all metrics available to estimate project size. This metric is very popular because it is the simplest to use. Using this metric, the project size is estimated by counting the number of source instructions in the developed program while counting the number of source instructions, lines used for commenting the code and the header lines should be ignored. Estimation is dependent on programming language. For different programming language lines of code will vary. **Function Point metric** In this method, the number and type of function supported by the software are utilized to find FPC (Function point count).	**4M** ***2M for each metric***

The steps in function point analysis are:

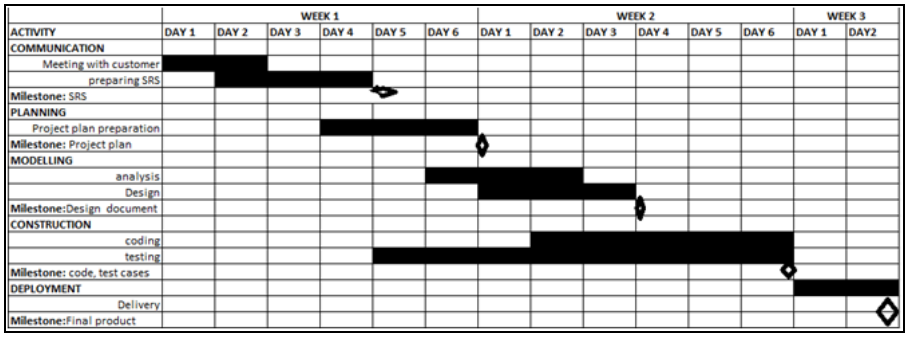
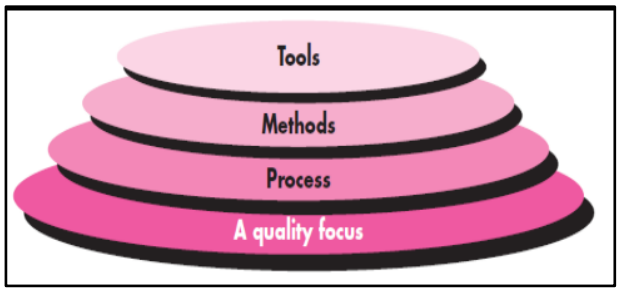
- Count the number of functions of each proposed type
- Compute the unadjusted function point (UFP)
- Find total degree of influence (TDI)
- Compute value adjustment factor (VAF)
- Find the function point count (FPC)

Count the number of functions of each proposed type:
Functions belonging to the following types:
External Input: Functions related to data entering the system.
External Outputs: Functions related to data existing from the system.
External Enquires: They lead to data retrieval from the system.
Internal Files: Logical files maintained within the system.
External interface files: These are logical files of other application used by our application.

Compute the unadjusted function point (UFP)
Categories each of the function types like simple, average or complex based on their complexity. Multiply the count of each function type with its weighing factor and find the weighted sum.

Function type	Simple	Average	Complex
External Inputs	3	4	6
External Output	4	5	7
External Inquiries	3	4	6
Internal Logical Files	7	10	15
External Interface Files	5	7	10

Find total degree of influence (TDI)
Use the 14 general characteristics of system to find the degree of influence of each of them. The sum of all 14 degree of influence will give TDI. The range of TDI is 0 to 70.
Compute value adjustment factor (VAF)
VAF=(TDI*0.01)+0.65
Find the function point count (FPC)
FPC=UFP*VAF

	d) **Ans.**	**Prepare Macro Timeline chart for 20 days of Hotel Management system (6 days a week) consider broad phase of SDLC.**  	**4M** ***4M for correct timeline chart***
4.	**a)** **Ans.**	**Attempt any <u>THREE</u> of the following:** **Draw and explain Software Engineering Layered technology approach.** Software engineering is a layered technology. The layers of software engineering as shown in the below diagram are: -  **1. A Quality Focus:** Any engineering approach (including software engineering) must rest on an organizational commitment to quality. Total quality management, six sigma and similar philosophies foster a continuous process improvement culture, and it is this culture that ultimately leads to the development of increasingly more effective approaches to software engineering. The bedrock that supports software engineering is a quality focus.	**12** **4M** ***2M for diagram 2M for explanation***

		2. Process Layer: The foundation for software engineering is the process layer. Software Engineering process is the glue that holds the technology layers together and enables rational and timely development of computer software. Process defines a framework that must be established for effective delivery of software engineering technology. The software process forms the basis for management control of software projects and establishes the context in which technical methods are applied, works products (models, documents, data, reports, forms etc.) are produced, milestones are established, quantity is ensured and change is properly managed. **3.Methods:** Software Engineering methods provide the technical ―how to building software. Methods encompass a broad array of tasks that include communication, requirements analysis, design modeling, program construction, testing and support. **4.Tools:** Software Engineering tools provide automated or semi-automated support for the process and the methods. When tools are integrated so that information created by one tool can be used by another, a system for the support of software development, called computer–aided software engineering is established.	
	b) **Ans.**	**State the need of SRS and also enlist the characteristics**. The need of SRS document is to provide • A detailed overview of software product, its parameters and goals. • The description regarding the project's target audience and its user interface hardware and software requirements. • How client, team and audience see the product and its functionality. **Characteristics of SRS:** • Correctness • Completeness • Consistency	**4M** ***2M for enlisting*** ***2M for characteristics***

<table>
<tr><td></td><td></td><td>

- Unambiguousness
- Modifiability
- Traceability
- Testability
- Understandable by stakeholder

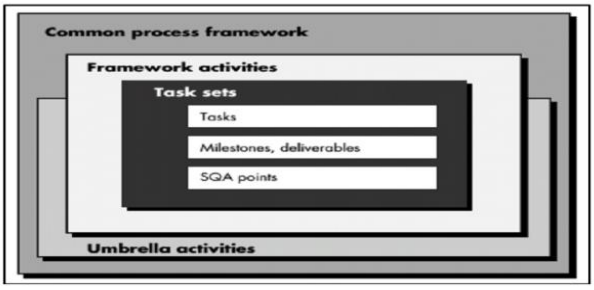
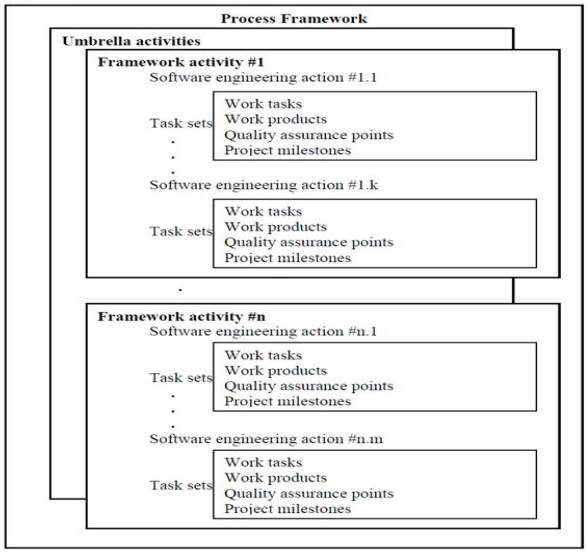
</td><td></td></tr>
<tr><td></td><td>c)

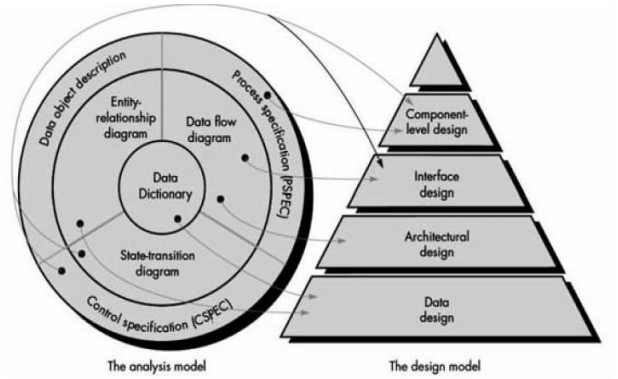
Ans.</td><td>Distinguish between Black Box and White Box testing. (Write any four points)
<table>
<tr><th>White box testing</th><th>Black Box Testing</th></tr>
<tr><td>The tester needs to have the knowledge of internal code or program.</td><td>This technique is used to test the software without the knowledge of internal code or program</td></tr>
<tr><td>It aims at testing the structure of the item being tested.</td><td>It aims at testing the functionality of the software</td></tr>
<tr><td>It is also called structural testing, clear box testing, code-based testing, or glass box testing.</td><td>It also known as data driven, closed box testing, data-, and functional testing.</td></tr>
<tr><td>Testing is best suited for a lower level of testing like Unit Testing, Integration testing.</td><td>This type of testing is ideal for higher levels of testing like System Testing, Acceptance testing.</td></tr>
<tr><td>Statement Coverage, Branch coverage, and Path coverage are White Box testing technique.</td><td>Equivalence partitioning, Boundary value analysis are Black Box testing technique</td></tr>
<tr><td>Can be based on detailed design documents.</td><td>Can be based on Requirement specification document.</td></tr>
</table></td><td>4M

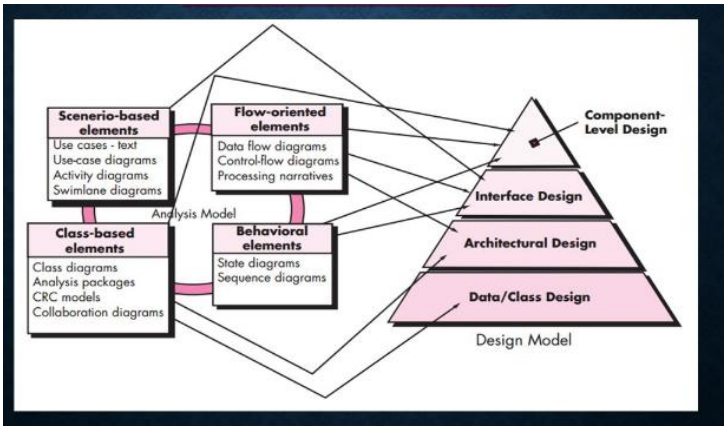
1M for each valid point</td></tr>
<tr><td></td><td>d)
Ans.</td><td>Explain RMMM strategy.
Risk mitigation, monitoring, and management (RMMM) plan. A risk management strategy can be included in the software project plan or the risk management steps can be organized into a separate Risk Mitigation, Monitoring and Management Plan.</td><td>4M
4M for correct explanation</td></tr>
</table>

		The RMMM plan documents all work performed as part of risk analysis and is used by the project manager as part of the overall project plan. Once RMMM has been documented and the project has begun, risk mitigation and monitoring steps commence. **Risk mitigation is a problem avoidance activity.** Risk monitoring is a project tracking activity with three primary objectives: 1) To assess whether predicted risks do, in fact, occur; 2) To ensure that risk aversion steps defined for the risk are being properly applied; and 3) To collect information that can be used for future risk analysis. In many cases, the problems that occur during a project can be traced to more than one risk. Another job of risk monitoring is to attempt to allocate origin (what risk(s) caused which problems throughout the project). **An effective strategy must consider three issues: • Risk avoidance • Risk monitoring • Risk management and contingency planning.** If a software team adopts a proactive approach to risk, avoidance is always the best strategy. This is achieved by developing a plan for risk mitigation. To **mitigate this risk**, project management must develop a strategy for **reducing turnover.** Among the possible steps to be taken are • Meet with current staff to determine causes for turnover (e.g., poor working conditions, low pay, and competitive job market). • Mitigate those causes that are under our control before the project starts. • Once the project commences, assume turnover will occur and develop techniques to ensure continuity when people leave. • Organize project teams so that information about each development activity is widely dispersed. • Define documentation standards and establish mechanisms to be sure that documents are developed in a timely manner. • Conduct peer reviews of all work (so that more than one person is "up to speed).	

		• Assign a backup staff member for every critical technologist. As the project proceeds, **risk monitoring** activities commence. The project manager monitors factors that may provide an indication of whether the risk is becoming more or less likely. In the case of **high staff turnover, the following factors can be monitored**: • General attitude of team members based on project pressures. • The degree to which the team has jelled. • Interpersonal relationships among team members. • Potential problems with compensation and benefits. • The availability of jobs within the company and outside it. In addition to monitoring these factors, the **project manager should monitor the effectiveness of risk mitigation steps**. RMMM steps incur additional project cost. Part of risk management, therefore, is to evaluate when the benefits accrued by the RMMM steps are outweighed by the costs associated with implementing them. In essence, the project planner performs a classic cost/benefit analysis.	
	e) Ans.	**State and describe any four basic project scheduling principles**. **Basic principles software project scheduling are:** **Compartmentalization:** The project must be compartmentalized into a number of manageable activities and tasks. To accomplish compartmentalization, both the product and the process are decomposed. **Interdependency:** The interdependency of each compartmentalized activity or task must be determined. Some tasks must occur in sequence while others can occur in parallel. Some activities cannot commence until the work product produced by another is available. Other activities can occur independently. **Time allocation**: Each task to be scheduled must be allocated some number of work units (e.g., person-days of effort). In addition, each task must be assigned a start date and a completion date that are a function of the interdependencies and whether work will be conducted on a fulltime or part-time basis. **Effort validation**: Every project has a defined number of staff members. As time allocation occurs, the project manager must ensure that no more than the allocated number of people has been scheduled at any given time.	**4M** ***1M for each principle***

		Defined responsibilities: Every task that is scheduled should be assigned to a specific team member. **Defined milestones:** Every task or group of tasks should be associated with a project milestone. Program evaluation and review technique (PERT) and critical path method (CPM) are two project scheduling Methods that can be applied to software development. **Defined outcomes** – Every task that is scheduled should have a defined outcome for software projects such as a work product or part of a work product – Work products are often combined in deliverables.	
5.	**a)** **Ans.**	**Attempt any <u>TWO</u> of the following:** **Explain software process framework with neat labeled diagram and also describe software process framework activities.** Software process framework diagram :  OR  	**12** **6M** ***3M for Diagram*** ***3M for description***

		A process framework establishes the foundation for a complete software process by identifying a small number of framework activities that are applicable to all software projects; In addition, the process framework encompasses a set of umbrella activities that are applicable across the entire software process. Basic framework activities: **1. Communication**: This framework activity involves heavy Communication & collaboration with the customer (and the stakeholders) and encompasses requirements gathering and other related activities. **2. Planning**: This activity establishes a plan for the software engineering work that follows. It describes the technical tasks to be conducted; the risks are analyzed. Project tracking should be done. Deadline is fixed. **3. Modeling**: This activity encompasses the creation of models that allow the developer & the customer to better understand software requirements & the design that will achieve those requirements. **4. Construction**: This activity combines code generation and the testing that is required uncovering errors in the code. **5. Deployment**: The software is delivered to the customer who evaluates the delivered product and provides feedback based on the evaluation.	
	b) **Ans.**	**Draw and explain translating requirement model into design model**. Translation of requirement model into design model diagram  OR	**6M** ***3M for diagram*** ***3M for description***



Software requirements, manifested by the data, functional, and behavioural models, feed the design task. Using one of a number of design methods, the design task produces a data design, an architectural design, an interface design, and a component design. Each of the elements of the analysis model provides information that is necessary to create the four design models required for a complete specification of design.

Design is a meaningful engineering representation of something that is to be built. It can be traced to a customer's requirements and at the same time assessed for quality against a set of predefined criteria for—good‖design. In the software engineering context, design focuses on four major areas of concern: data, architecture, interfaces, and components Design begins with the requirements model.

The data design transforms the information domain model created during analysis into the data structures that will be required to implement the software. The data objects and relationships defined in the entity relationship diagram and the detailed data content depicted in the data dictionary provide the basis for the data design activity. Part of data design may occur in conjunction with the design of software architecture. More detailed data design occurs as each software component is designed. The architectural design defines the relationship between major structural elements of the software, the design pattern that can be used to achieve the requirements that have been defined for the system, and the constraints that affect the way in which architectural design patterns can be applied.

	c)	**Describe following project cost estimation approaches** **i) Heuristic** **ii) Analytical** **iii) Empirical**	**6M**
	Ans.	i) Heuristic cost estimation approach: This technique basically use the concept of learning from the previous project and estimate the cost. The objective is to find a similar system produced earlier and through knowing how the properties of the new system vary from the existing one. Two classes of different heuristic Estimation Techniques: - Single variable model - Multi variable model 1. Single Variable Estimation Models: It provides a means to estimate the desired characteristics of a problem, using some previously estimated basic (independent) characteristic of the software product such as its size. A single variable estimator model takes the following form: Estimated Parameter = c1 * ed1 e= characteristic which already have been calculated. Estimated parameter is the dependent parameter to be estimated. The dependent parameters to be estimated could be effort, duration, staff size etc. c1 and d1 are constants- calculated from past projects. COCOMO is one of this type of models example. 2. Multi variable Cost Estimation Model: It has the following form Estimated Resources = c1 * e1d1 + c2 * e2d2 + - - - - e1 and e2 are the basic independent characteristics of the software already estimated. c1, c2, d1, d2, are constants. Multivariable Estimation Models are expected to give more accurate estimate compared to the Single Variable Models, since a project parameters is typically influenced by several independent parameters. The independent parameters influence the dependent parameter to different extents. This is modeled by the constants c1,c2,d1,d2..... ii) Analytical cost estimation approach Analytical estimation techniques derive the required results starting with basic assumptions regarding the project.	***2M for each approach***

- Thus, unlike empirical and heuristic techniques, analytical techniques do have scientific basis.
- Halstead's software science is an example of an analytical technique.
- Halstead_s software science can be used to derive some interesting results starting with a few simple assumptions. Halstead_s software science is especially useful for estimating software maintenance efforts.
- In fact, it outperforms both empirical and heuristic techniques when used for predicting software maintenance efforts.
- Halstead's Software Science – An Analytical Technique Halstead_s software science is an analytical technique to measure size, development effort, and development cost of software products.
- Halstead used a few primitive program parameters to develop the expressions for overall program length, potential minimum value, actual volume, effort, and development time.

iii) Empirical cost estimation approach

Empirical estimation techniques are based on making an educated guess of the project parameters. While using this technique, prior experience with development of similar products is helpful. Although empirical estimation techniques are based on common sense, different activities involved in estimation have been formalized over the years. Two popular empirical estimation techniques are:

1. Expert judgment technique and
2. Delphi cost estimation.

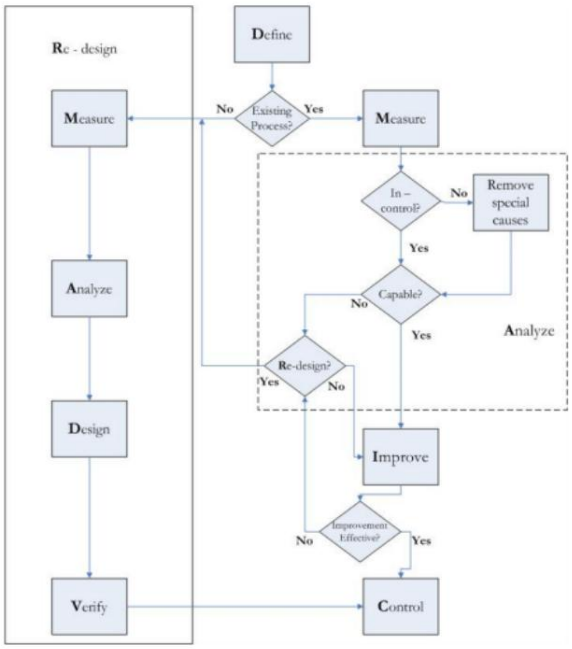
1. Expert Judgment Technique

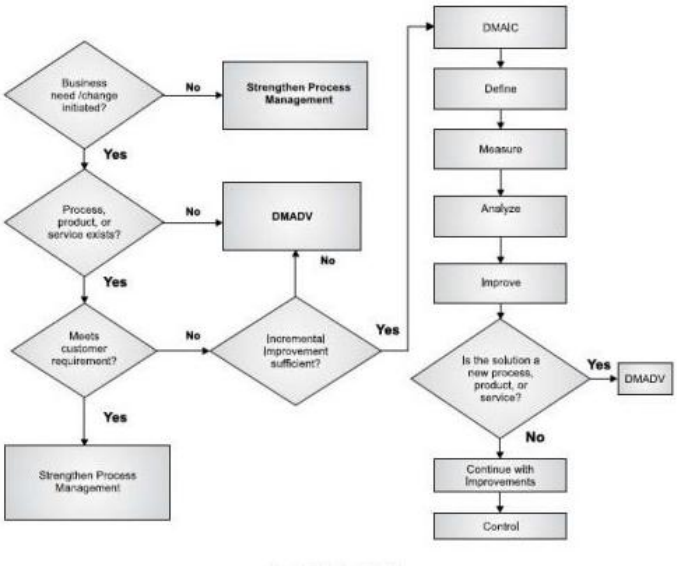
Expert judgment is one of the most widely used estimation techniques.

- In this approach, an expert makes an educated guess of the problem size after analyzing the problem thoroughly.
- Usually, the expert estimates the cost of the different components (i.e. modules or subsystems) of the system and then combines them to arrive at the overall estimate.
- However, this technique is subject to human errors and individual bias.

		• Also, it is possible that the expert may overlook some factors inadvertently. • Further, an expert making an estimate may not have experience and knowledge of all aspects of a project. 2. Delphi cost estimation • Delphi cost estimation approach tries to overcome some of the short comings of the expert judgment approach. • Delphi estimation is carried out by a team comprising of a group of experts and a coordinator. • In this approach, the coordinator provides each estimator with a copy of the software requirements specification (SRS) document and a form for recording his cost estimate. • Estimators complete their individual estimates anonymously and submit to the coordinator. In their estimates, the estimators mention any unusual characteristic of the product which has influenced his estimation. • The coordinator prepares and distributes the summary of the responses of all the estimators, and includes any unusual rationale noted by any of the estimators. • Based on this summary, the estimators re-estimate. This process is iterated for several rounds. However, no discussion among the estimators is allowed during the entire estimation process. The idea behind this is that if any discussion is allowed among the estimators, then many estimators may easily get influenced by the rationale of an estimator who may be more experienced or senior. After the completion of several iterations of estimations, the coordinator takes the responsibility of compiling the results and preparing the final estimate.	
6.	**a) Ans.**	**Attempt any <u>TWO</u> of the following:** **State and describe any six communication principles.** Communication principles are as given below: **1. Listen carefully** i. To collect lots of data from the client, the developer team has to listen carefully. ii. Maximum information with respect to requirement and the specifications should be collected before the implementation and the designing of the software.	**12** **6M** ***1M for each principles with descriptio n***

		2. Prepare before you communicate i. A proper agenda or the guidelines for the meetings should be prepared before the start of the meeting. ii. Complete detail and the description about the clients and their work area should be gathered to deliver the software up to the best expectation. **3. Have a facilitator for any communication meeting** i. The requirement gathering and the specification are important for any software development, hence the communication should continue till the requirement gathering is over. **4. Face-to-face communication is best** i. It is always better to sit across the table and have discussion on the requirement on the software development by the client and the developer. ii. Distant communication does not help gathering data properly. **5. Take notes and document decisions** i. The important points discussed should also be recorded. ii. Proper notes and the documentation is important for the successful completion and deployment of the project. **6. Strive for collaboration** i. Collaboration in terms of teamwork is required for the successful completion of the software. ii. The collective knowledge of the team members should be implemented in the development. **7. Stay focused and modularize your discussion** i. As the development is the working of many team members, so the possibility of the discussion going from one topic to the other topic is quite possible. ii. As a good software developer it is required that the discussion remains focused on the specified area. **8. Draw a picture if something is unclear** i. Drawing flowcharts, E-R diagrams and other supporting graphical representations give clarity to the discussion and the documentation. **9. Move on once you agree, move on when you can't agree, move on if something unclear can't be clarified at the moment** i. Healthy discussion leads to the final conclusion of successful implementation of the software. ii. Once reached to final statement recorded should move to the next step.	

		iii. If no conclusion is reached than that point should be left and move ahead with new implementation which is cost effective. **10. Negotiation is not a contest or game** i. Negotiation should be mutual not to put someone down or make them feel to be the loser.	
	b) Ans.	**Describe six sigma strategy in details.** 1. Six Sigma is the process of producing high and improved quality output. 2. This can be done in two phases – identification and elimination. The cause of defects is identified and appropriate elimination is done which reduces variation in whole processes. 3. Six Sigma projects follow two project methodologies: 	**6M** ***3M (1.5 each) DMAIC & DMADV Description*** ***3M for diagram***



(Note :Any other correct diagram should be given marks)

a) DMAIC

It specifies a data-driven quality strategy for improving processes. This methodology is used to enhance an existing business process. The DMAIC project methodology has five phases:

i) Define:-It covers the process mapping and flow-charting, project charter development, problem- solving tools, and so-called 7-M tools.

ii) Measure:-It includes the principles of measurement, continuous and discrete data, and scales of measurement, an overview of the principle of variations and repeatability and reproducibility (RR) studies for continuous and discrete data.

iii) Analyze:-It covers establishing a process baseline, how to determine process improvement goals, knowledge discovery, including descriptive and exploratory data analysis and data mining tools, the basic principle of Statistical Process Control (SPC), specialized control charts, process capability analysis, correlation and regression analysis, analysis of categorical data, and non-parametric statistical methods.

iv) Improve:-It covers project management, risk assessment, process simulation, and design of experiments (DOE), robust design concepts, and process optimization.

		v) Control:-It covers process control planning, using SPC for operational control and PRE-Control. a) DMADV It specifies a data-driven quality strategy for designing products and processes. This method is used to create new product designs or process designs in such a way that it results in a more predictable, mature, and detect free performance. The DMADV project methodology has five phases: **a. Define:**-It defines the problem or project goal that needs to be addressed. **b. Measure:**-It measures and determines the customer's needs and specifications. **c. Analyze:**-It analyzes the process to meet customer needs. **d. Design:**-It can design a process that will meet customer needs. **e. Verify:**-It can verify the design performance and ability to meet customer needs.	
	c) **Ans.**	**Use COCOMO model to calculate** **i) Effort** **ii) Development Time** **iii) Average staff size** **iv) Productivity** **If estimated size of project is 400 KLOC using embedded mode.** Given size if project= 400 KLOC; mode = embedded In embedded mode : a= 1.8 b=1.20 c=2.5 d=0.32 i) Effort E = ai (KLOC) bi E=1.8* (400) 1.20 =1.8 * 1325.78 = 2386.40 per month ii)Development time D= c * E d = 2.5 * (2386.40) 0.32 =2.5 * 12.04 =75.25 months	**6M** ***2M for each correct answer and formula of effort, developme nt time and productivi ty***

		iii) Average staff size ss= E/ D =2386.40 / 75.25 =31.71 persons iv) Productivity P=KLOC /E =400/ 2386.40 =0.16	