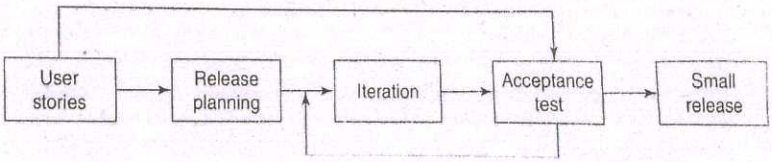
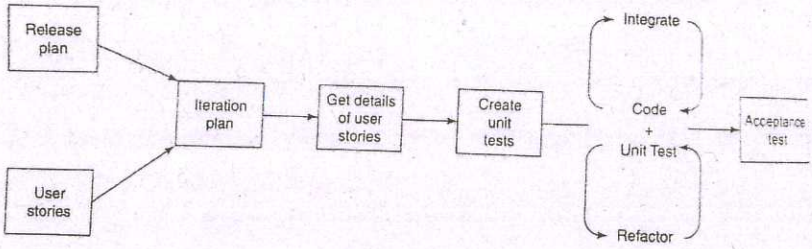
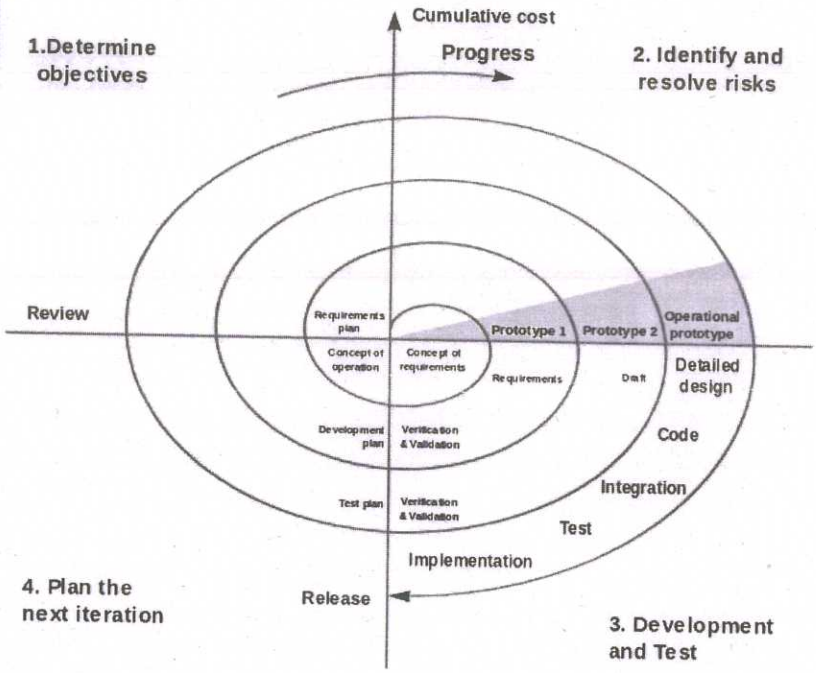
SCHEME OF VALUATION				
Scoring Indicators				
Revision: 2015		**Course Code:5132**		
Course Title: PROJECT MANAGEMENT AND SOFTWARE ENGINEERING				
Qst No	**Scoring Indicators**	**Split up score**	**Sub Total**	**Total**
	Part -A			
I 1)	Software engineering is defined as a systematic approach to the development, operation, maintenance, and retirement of software	2	2	**10**
2)	Security,Legal or Regulatory Requirements, Scalability Portability, Environmental Reliability, Performance requirements , Accuracy and Precision (any two)	Any 2 x 1 Mark	2	
3)	Weighted methods per class(WMC), depth of inheritance tree(DIT), coupling between classes(CBC), response for a class(RFC)	Any 2 x 1 Mark	2	
4)	UNIT TESTING is a level of software testing where individual **units/components** of a software are tested by the **programmer itself**		2	
5)	Risk is defined as an exposure to the chance of injury or loss. Risk implies that there is a possibility that something negative may happen		2	
	Part -B			
II 1)	Phases of software development **Requirement analysis** Software Development Life Cycle begins with Requirement Analysis phase, where the stakeholders discuss the requirements of the software that needs to be developed to achieve a goal. The aim of the requirement analysis phase is to capture the details of each requirement and to make sure everyone understands the scope of the work and how each requirement is going to be fulfilled **Design** The next stage of Software Development Life Cycle is the Design phase. During the design phase, developers and technical architects start the high-level design of the software and system to be able to deliver each requirement. **Coding and implementation**	4 x 1.5 Marks	6	**30**

	In this phase, developers start coding according to the requirements and the design discussed in previous phases.Database admins create the necessary data in the database, front-end developers create the necessary interfaces and GUI to interact with the back-end all based on guidelines and procedures defined by the company. **Testing** During testing, experienced testers start to test the system against the requirements.The testers aim to find defects within the system as well as verifying whether the application behaves as expected and according to what was documented in the requirements analysis phase **Maintenance** maintenance team that look after any post-production issues.			
2)	Agile development process involved in 1990 as reaction to documentation and Bureaucracy based process particularly the waterfall approach. Agile approaches are based on some common principles given below. 1. Working software is the key measure of progress in a project 2. Software should be developed and delivered rapidly in small increments 3. Even late changes in the requirement should be entertained 4. Face to face communication is preferred over documentation 5. Continuous feedback and involvement of customers is necessary for developing good quality software. 6. Simple design which evolves and improve with time is a better approach than doing an elaborate design upfront for handling all possible scenarios. 7. The delivery dates are decided by empowered teams of talented individuals. Overall process   iteration	Any 4 points/diagram x 1.5 Marks	6	

3)	Difference between object oriented and function oriented design 1. Classes and objects are the basic building blocks of object oriented design Where are functions and procedures are building blocks for function oriented design 2. Objects are entities that encapsulates some state and provide services to be used by a client, which could be another object, program or a user. Due to encapsulation, preserving data integrity is easier in object oriented design than function oriented design. 3. An important property of objects is that the state persist, in contrast to the data defined in a function or procedure which is generally lost once the function stop being active.	3 x 2 marks	6	
4)	Design concepts in software engineering The fundamental design concepts are: **Abstraction**:Focus on solving a problem without being concerned about the internal implementation **Refinement:**It is the process of elaboration where the designer provides successively more detail for each design component **Modularity:** It is the degree to which software can be understood by examining its components independently of one another **Software architecture:** It provides the overall structure of the software components and the ways in which that structure provides conceptual integrity for a system **Information hiding:**Modules must be specified and designed so that the information like algorithm and data presented in a module is not accessible for other modules not requiring that information. **Cohesion:**A cohesive module performs a single task and it requires a small interaction with the other components in other parts of the program. **Coupling:**Coupling is an indication of interconnection between modules in a structure of software. **Data structure:**The Representation of the logical relationship among individual data elements **Software Procedure:** The precise specification of processing	Any 4 x 1.5 Marks	6	
5)	1. **Error**:The difference between the actual output of software and the correct output. 2. **Fault**:It is a condition that causes system to fail in performing its required function. Commonly used the term bug or defect. 3. **Failure:**It is the inability of a system or component to perform required function according to its specification .A software failure of course if the behaviour of the software is different from the specified behaviour.Failure may be caused by functional or performance factors.	3 x 2 Marks	6	
6)	Explain black-box testing Black box testing is not concerned with the structure of the program the test cases are decided only on the basis of the requirements or specifications of the program or module	3 x 2 Marks	6 marks	

	Some of the test technique used to select test cases **Equivalence class partitioning:** 1. Divide the programs input space into domain such that all inputs within a domain are equivalent 2. Write test cases by using one element from each equivalence class **Boundary value analysis:** Boundary value analysis is a set of input data that lies on the edge or boundary of a class of input data or that generate output that lies at the boundary of a class of output data.Boundary values for each equivalence class including different class of the output should be covered. 1. Compliments equivalence Partitioning 2. Selection of test cases at the edges			
7)	During the project many products are produced which are typically composed of many items for example the final source code may be composed of many source files.These items keep evolving as the project proceeds, creating many versions on the way. As development process generally do not focus on evolution and changes, to handle them another process called software configuration management is used. The Objective of this component process is primarily deals with managing change so that integrity of the product is not violated despite change. The five disciplines are: 1. CM Planning and Management: 2. Configuration Identification (CI) 3. Configuration Control: 4. Configuration Status Accounting 5. Configuration Verification and Audit	6 Marks	6	
	Part -C			
III a)	Advantages and disadvantages of waterfall model The waterfall model is the simplest model for software development.where the requirement ,design,coding and testing phases are performed in linear progression. Advantages 1. This model is simple and easy to understand and use. 2. Straightforward and divide the large task of building a software system into series of cleanly divided phases,each phase dealing with a separate logical concern 3. In this model phases are processed and completed one at a time 4. Waterfall model works well for smaller projects where requirements are clearly defined and very well understood. 5. Well understood milestones. Disadvantages 1. Assume requirement can be frozen before the design begins,for a new system difficult to determine	8x1Marks	**8**	**15**

	requirements,Hence unchanging requirements is unrealistic for such projects 2. Freezing requirement usually requires choosing the hardware.Large project might take few years to complete.By this time the selected hardware may be outdated 3. Follows the big bang approach - the entire software is delivered in one shot at the end. this entails heavy risks as the user does not know until the very end what they are getting 4. Encourage requirement bloating.Since all the requirements must be specified at the start ,users may add even those features which they might not be needed 5. It is a document driven process that requires formal documents at the end of each phase			
III b)	• A process is a sequence of steps executed to achieve a goal.While developing software, the purpose is to develop software to satisfy the needs of some users or client. A software project is one instance of this problem and the development processes what is used to achieve this purpose. • Project development process define the task the project should perform an in the order in which they should be done. • A project limit the degrees of freedom for a project by specifying what type of activities must be undertaken and in what order the shortest path is obtained from the user need to the software satisfying these needs the Software satisfying this needs. • The process drives a project and heavily influence the outcome. • Some of process models are 1. Waterfall model 2. prototyping model 3. iterative development model	2+2+2+1	7	
IV a)	spiral model and its advantages This model was first described by Barry Boehm in his 1986 paper "A Spiral Model of Software Development and Enhancement". Provides support for Risk Handling Spiral Model is a combination of a waterfall model and iterative model. Each phase in the spiral model begins with a design goal and ends with the client reviewing the progress. In its diagrammatic representation, it looks like a spiral with many loops.	Diagram 3 Marks + Phases 3 marks + Any 2 advantages 2 Marks	**8**	**15**



Spiral Model Phases

Planning

It includes estimating the cost, schedule and resources for the iteration. It also involves understanding the system requirements for continuous communication between the system analyst and the customer

Risk Analysis

Identification of potential risk is done while risk mitigation strategy is planned and finalized

Engineering

It includes testing, coding and deploying software at the customer site

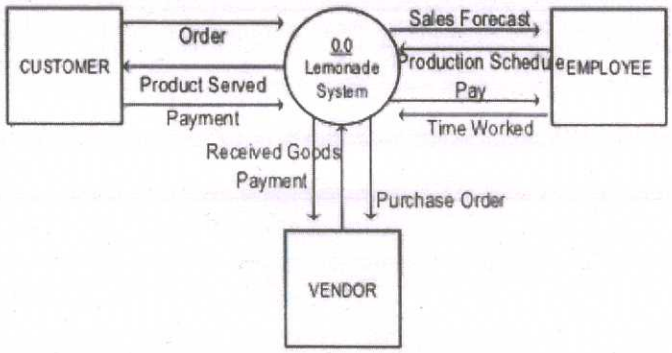
Evaluation

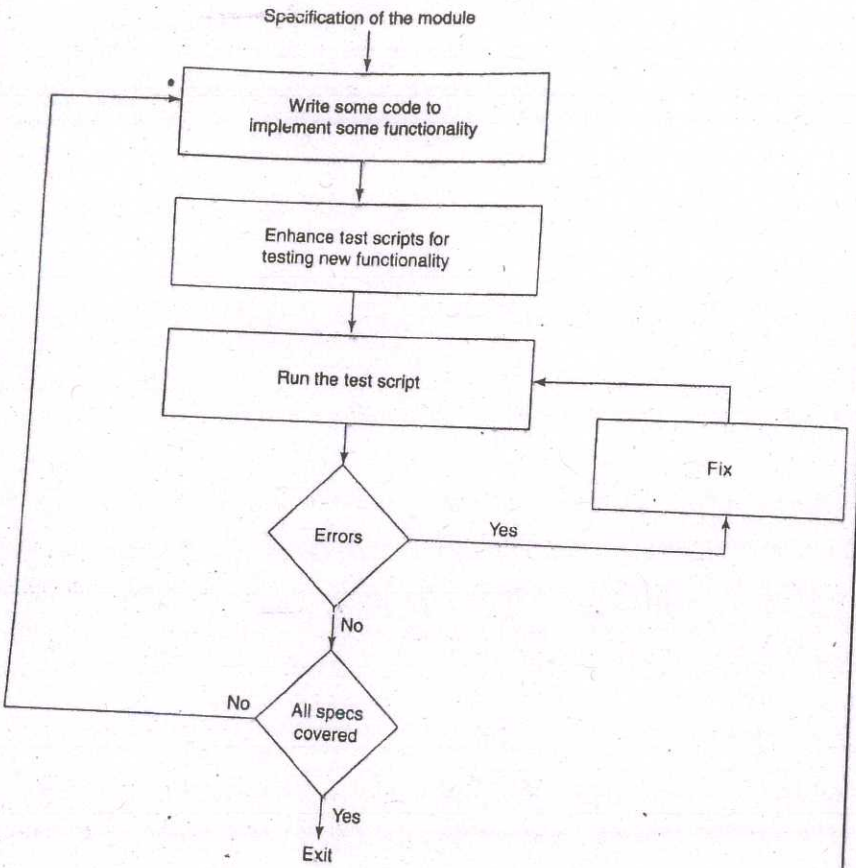
Evaluation of software by the customer. Also, includes identifying and monitoring risks such as schedule slippage and cost overrun

Advantages of Spiral Model

1. Additional functionality or changes can be done at a later stage
2. Cost estimation becomes easy as the prototype building is done in small fragments
3. Continuous or repeated development helps in risk management
4. Development is fast and features are added in a systematic way
5. There is always a space for customer feedback
6. Allows extensive use of prototypes.
7. Users see the system early.

IV b)	Explain the importance of software engineering Software Engineering is a part of computer science in which several kinds of methods, thoughts and techniques used for getting the high quality software and computer programs. 1. To provide the best output of software system. 2. To make easy to use the software systems and develop them. 3. To improve the rate of production. 4. To maintain the budget for development of Software system. 5. Job satisfaction of software engineering. 6. Reduces complexity 7. To minimize software cost 8. To decrease time 9. Handling big projects 10. The quality and productivity achieved in a software projects depends on the process for executing the project,due to this the processes form the heart of the software engineering	Any 7 x 1 Marks	**7**	
V a)	<u>Value of a good SRS</u> A basic purpose of the SRS is to bridge this communication gap so they have shared vision of the software being built. 1. An SRS establishes the basis for agreement between the client and the supplier on what the software product will do. 2. An SRS provides a reference for validation of the final product 3. This helps the client to determine if the software meets the requirements. 4. A high quality SRS is a prerequisite to high-quality software. 5. An high-quality SRS reduces the development cost 6. Cost of fixing an error increases almost exponentially as time progress,Hence improving the quality of requirement ,we can have a huge savings in the future expensive defect removal	Any 4 x 2 Marks	**8**	**15**
V b)	<u>Explain Data Flow diagram with suitable example</u> A data flow diagram shows the flow of data through a system. It Views a system as a function that transforms the input into desired output. Any Complex system will typically undergo a series of transformations before it becomes output, the DFT to capture this transformation.The agent that performs the transformation of data from one state to another is called process (or a bubble).The processes are represented by named arrows entering or leaving the bubbles a rectangle represent a source of sinks and is a net originator or consumer of data.	Explanation 4 marks Diagram 3 marks	**7**	

	Context Level DFD  			
VI a)	Explain the characteristics of an SRS Correct,complete,unambiguous,verifiable,consistent SRS is correct if every requirement is included in the SRS, represent something required in the final system. It is complete if everything software is supposed to do and represent all classes of input that are specified in the SRS. It is unambiguous if and only if every requirement that has one and only one interpretation. Verifiable if and only if every stated requirement is verifiable. It is consistent if there is no requirement that contacts with another. High quality SRS is prerequisite to high quality software and reduces the development cost	Any 4 with explanation x 2 Marks	**7**	**15**
VI b)	Coupling and Cohesion are two modularization criteria which are often used together. The two models are considered independent if one can function completely without the presence of others. However all the modules in the system cannot be independent of each other as they must interact so that together they produce decide external behaviour of the system. Coupling between modules is the strength of interconnection **between modules** or measure of interdependence among modules i.e. higher the Independence higher is the coupling. Coupling increases the complexity and obscurity of the interface between modules. Complexity of the interface is another factor affecting coupling. The more complex each interface is, the higher will be the degree of coupling Cohesion is to strengthen the bond between elements of the **same module** by maximizing the relationship between elements of the same module. Cohesion of a module represent how tightly bond the internal elements of the module are to one another.	Any 7 x 1 Marks	**7**	

	Cohesion is to strengthen the bond between elements of the **same module** by maximizing the relationship between elements of the same module. Cohesion of a module represent how tightly bond the internal elements of the module are to one another. Cohesion of a module gives the designer an idea about whether the different elements of a module belong together in the same module. Cohesion and coupling are clearly related. Usually the greater the cohesion of each module in the system, the lower the coupling between modules.			
VII a)	Explain test-driven development and incremental coding process In **incremental coding** write code for implementing only part of the functionality of the module, this code is compiled and tested with some quick test to check the part that has been written so far.When the code passes these tests, the developer for proceed to add further functionality to the code, Which is then tested again. In code first approach it is all too common to write a long piece of code but then only write a few tests which cover only some part of the code In test driven development a programmer first write the test case and then write the code to pass the test the whole process is done incrementally.Write just enough code to pass the test  **Figure 1** An incremental coding process.	Explanation (2+2) marks Diagram (2+ 2) Marks	**8**	**15**

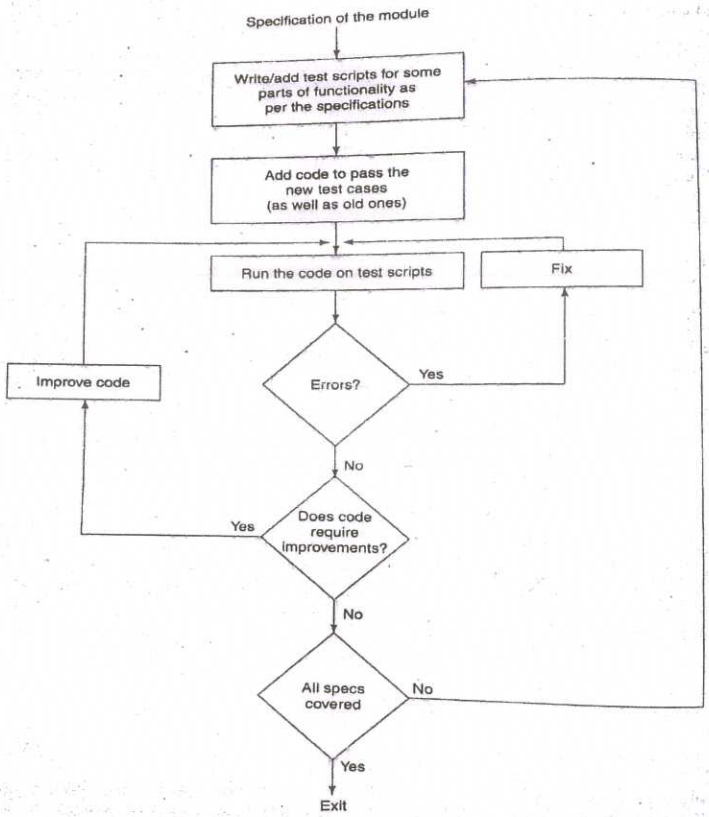


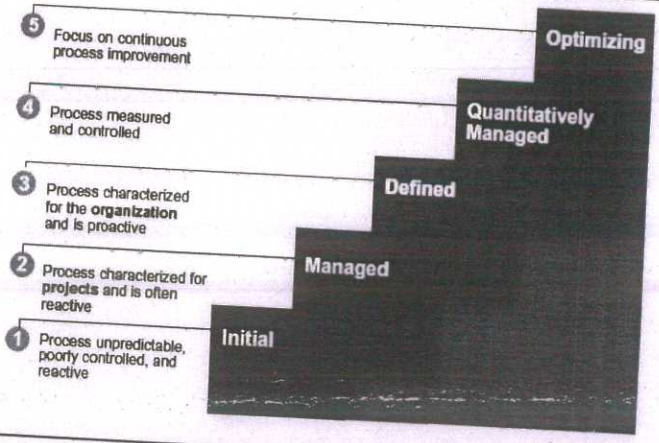
Figure 2 Test-driven development process.

VII b)	At programmer level the testing done for checking the code the programmer has developed is called unit testing. A unit may be a function or a small collection of functions for procedural programming language, and a class or small collection of classes for an object oriented languages. Unit testing is like regular testing where programs are executed with some test cases except that the focus is on testing smaller programs or modules which are typically assigned to one programmer. During unit testing the tester, who is generally the programmer, will execute the unit with a variety of test cases and study the actual behaviour of the units being tested for these test cases. Based on the behaviour, the tester decide whether unit is working correctly or not. if behaviour is not expected some test cases then the programmer find defects in the program and fix it. An issue with unit testing is that as the unit being tested is not complete system but just a part, it is not executable by itself furthermore in its execution it may use other module that have not been developed yet due to this unit testing often requires drivers or stubs to be written.	Any 4 points.	7	

<table>
<tr>
<td>VIII a)</td>
<td>
Testing process
Testing is a quality control activity which focuses on identifying defects .
The testing process for a project consists of three high level task test planning, test cases design, and test case execution
Test plan
In a project testing commences with a test plan and terminate with successful execution of acceptance testing. A test plan is a General document for the entire project that defines the scope, approach to be taken, and schedule of testing, as well as identifies the test item for testing and personnel responsible for different activities of testing.The input for test plan are (1) project plan (2) Requirements document, (3) architecture or design document
Test plan should contain following
Test unit specification
Features to be tested
Approach for testing
Test deliverables
Schedule and task allocation
Test case design
Based on the approach specified in the test plan and the features to be tested the test cases are designed and specified for testing the unit. test case specification gives , for each unit to be tested, all test cases ,input to be used in the test cases, conditions being tested by the test case and outputs expected for those test cases.
Test case specification
<table>
<tr><td>Requirement Number</td><td>Condition to be tested</td><td>Test data and settings</td><td>expected output</td></tr>
<tr><td></td><td></td><td></td><td></td></tr>
</table>
Test case execution
With the specification of test cases the next step in the testing process is to execute them
During test case execution defects are found. This defects are then fixed and testing is done again to verify the fix.
defect life cycle:
Submitted->fixed ->closed.
When a defect is found it is logged in a defect control system along with sufficient information about the defect. The defect is then in the state submitted.The job of fixing the defect is then assigned to some person who is generally the author of the document or code. The assigned person does the debugging and fixes the reported defect and defect then moved to “fixed” state. The verification will be done by another person and marked as “closed” if it pasess all the related test cases.
</td>
<td>2+2+2+2</td>
<td>8</td>
<td>15</td>
</tr>
<tr>
<td>VIII b)</td>
<td>in a project many different people developed source code each programmer creates different source files which are eventually</td>
<td>Any 5 point</td>
<td>7</td>
<td></td>
</tr>
</table>

	combined together to create executables. programmers keep changing their source file as the code evolves,and often make changes in other source files as well. In order to keep control over the source and their evolution source code control is almost always used in project using tools like CVS, VSS, SVN GIT etc. Modern source code control system contains a repository which is essentially a controlled directory structure which keeps the full revision history of all the files produced by different programmers in the project team. For efficiency, a file history generally kept as delta increment from the base file. The normal behaviour of a project member will be as follows, • check out the latest version of the files to be changed • make the planned changes to them. • validate that changes have desired effect • commit the changes back to repository Besides using repository for maintaining the different versions, it is also used for constructing the software system from the source. an activity often called build.The build get the latest version of the sources from the repository and create executable from the source.			
IX a)	All elements used to develop a software product may be assumed as resources for that project. This may include human resource, productive tools and software libraries. The resources are available in limited quantity and stay in the organization as a pool of assets. The shortage of resources hampers the development of project and it can lag behind the schedule. Allocating extra resources increases development cost in the end. It is therefore necessary to estimate and allocate adequate resources for the project. Resource management includes - 1. Defining proper organization project by creating a project team and allocating responsibilities to each team member 2. Determining resources required at a particular stage and their availability 3. Manage Resources by generating resource request when they are required and de-allocating them when they are no more needed.	3 + 3	**6**	**15**
IX b)	Change management is the handling of change requests. A change request leads to the creation of a new release General change process (5 Marks) 1. The change is requested (this can be done by anyone including users and developers) 2. The change request is assessed against project goals 3. Following the assessment, the change is accepted or rejected 4. If it is accepted, the change is assigned to a developer and implemented 5. The implemented change is audited.	5+4	**9**	

	6. The complexity of the change management process varies with the project. 7. Small projects can perform change requests informally and quickly while complex projects require detailed change request forms and the official approval by one more managers. **Controlling Changes (4 Marks)** **Two types of controlling change:** 1. Promotion: The internal development state of a software is changed. 2. Release: A changed software system is made visible outside the development organization. Approaches for controlling change (Change Policy) 1. Informal (good for research type environments and promotions) 2. Formal approach (good for externally developed CIs and for releases)			
X a)	In CMMI models with a staged representation, there are five maturity levels designated by the numbers 1 through 5 **1. Initial** At maturity level 1, processes are usually ad hoc and chaotic. The organization usually does not provide a stable environment. Success in these organizations depends on the competence and heroics of the people in the organization and not on the use of proven processes. **2. Managed** At maturity level 2, an organization has achieved all the specific and generic goals of the maturity level 2 process areas. In other words, the projects of the organization have ensured that requirements are managed and that processes are planned, performed, measured, and controlled. **3. Defined** At maturity level 3, an organization has achieved all the specific and generic goals of the process areas assigned to maturity levels 2 and 3. At maturity level 3, processes are well characterized and understood, and are described in standards, procedures, tools, and methods. **4. Quantitatively Managed** **A**t maturity level 4, an organization has achieved all the specific goals of the process areas assigned to maturity levels 2, 3, and 4 and the generic goals assigned to maturity levels 2 and 3. **5. Optimizing** At maturity level 5, an organization has achieved all the specific goals of the process areas assigned to maturity levels 2, 3, 4, and 5 and the generic goals assigned to maturity levels 2 and 3. Processes are continually improved based on a quantitative understanding of the common causes of variation inherent in processes.	Diagram 3 marks CMMI Levels - 7 marks	**10**	**15**

				
X b)	Top-down approach and bottom-up approach of cost estimation **Top down approach** Starts from system level,examine the overall functionality of the product and how that functionality is provided by interacting sub-sanctions. The costs of system-level activities such as integration,configuration management and documentation are taken into account. **Bottom up approach** Starts from the component level.The system decomposed into components and the effort required to develop each of these are computed.These cost then added to give the effort required for the whole system development In top down estimation we can underestimate the cost of solving difficult technical problems associated with a component.There is no detailed justification of the estimate produced. In bottom up estimation produces such a justification and consider each component.	2 x 2.5 Marks	**5**	

QUESTION WISE ANALYSIS

COURSE : PROJECT MANAGEMENT AND SOFTWARE ENGINEERING (~~4261~~ 5132)

VERSION : A

Qn No.	Specific Outcome (as per syllabus)	Module	Content Details	Score	Time in Minutes
I. 1	1.1.11	I	Define software engineering and its importance	2	2
I. 2	2.1.4	II	Explain structure of an SRS document	2	2
I. 3	2.1.11	II	Explain Object Oriented Design and its Complexity Metrics	2	2
I. 4	3.1.5	III	Explain unit testing and Code Inspection	2	2
I. 5	4.1.1.4	IV	Describe how Project Risks can be identified, analyzed, mitigated, and monitored	2	2
II. 1	1.1.4	I	State Phases of software development	6	10
II. 2	1.1.3	I	Describe Software Process	6	10
II. 3	2.1.10	II	Explain Function Oriented Design and its Complexity Metrics	6	10
II. 4	2.1.9	II	Describe software design concepts	6	10
II. 5	3.1.6	III	Explain the testing concepts and testing process	6	10
II. 6	3.1.6	III	Explain unit testing and Code Inspection	6	10
II. 7	4.1.1.2	IV	Describe methods to Estimate project time and cost	6	10
III. a)	1.1.11	I	Describe Life Cycle Models-Classical waterfall, Iterative, prototyping, Spiral and Agile	8	16
III. b)	1.1.11	I	Describe Life Cycle Models-Classical waterfall, Iterative, prototyping, Spiral and Agile	7	14
IV. a)	1.1.11	I	Describe Life Cycle Models-Classical waterfall, Iterative, prototyping, Spiral and Agile	8	16
IV. b)	1.1.1	I	Define software engineering and its importance	7	14

V. a)	2.1.2	II	Describe Requirements specification	8	16
V. b)	2.1.9	II	Describe software design concepts	7	14
VI. a)	2.1.3	II	Describe the desirable characteristics of an SRS	8	16
VI. b)	2.1.9	II	Describe software design concepts	7	14
VII. a)	3.1.2	III	Describe the method of incrementally developing code	8	16
VII. b)	3.1.8	III	Describe Black-box testing	7	14
VIII. a)	3.1.1	III	Explain Programming principles and coding guidelines	8	16
VIII b)	3.1.6	III	Explain the testing concepts and testing process	7	14
IX. a)	4.1.1.3	IV	Describe about Resource Management	6	12
IX. b)	4.1.1.7	IV	Describe about Resource Management	9	18
X. a)	4.1.1.8	IV	Explain about CMMI, different levels and need of accreditation	10	20
X.b)	4.1.1.6	IV	Describe about Configuration Management	5	10
Total time					320

CODE: TED (15) ~~4251~~ 5132

COURSE: **PROJECT MANAGEMENT AND SOFTWARE ENGINEERING**

VERSION: A

BLUE PRINT

Sl No.	Module	Type of Questions							
		Part A		Part B		Part C		Total	
		No. Of Questions	Score	No. of Questions	Score	No. of Questions	Score	No. of Questions	Score
1	Module – I	1	2	2	12	4	30	7	44
2	Module – II	2	4	2	12	4	30	7	46
3	Module – III	1	2	2	12	4	30	8	44
4	Module - IV	1	2	1	6	4	30	6	38
Total		5	10	7	42	16	120	28	172