# AUTUMN END SEMESTER EXAMINATION-2018

5th Semester B.Tech & B.Tech Dual Degree

## SOFTWARE ENGINEERING

## IT-3003

(Regular-2016 & Back 2015 Admitted Batch)

Time: 3 Hours Full Marks: 60

***Answer any SIX questions including question No.1 which is compulsory.***

*The figures in the margin indicate full marks.*

*Candidates are required to give their answers in their own words as far as practicable and all parts of a question should be answered at one place only.*

| | | | |
|---|---|---|---|
| 1. | | | [2 10] |
| | (a) | statement is false as the name of the problem solving style itself is exploratory hence the solution has to be explored out many times instead of predefined steps. | |
| | (b) | statement is true because SRS document is the first document that defines the functional and non functional requirements as well as acts as the contract between the customer and software developer. | |
| | (c) | Performance – for example Response Time, Throughput, Utilization,Scalability.Capacity.Availability.Reliability.Recoverability.Maintainability.Serviceability. | |
| | (d) | Abstraction is one of the most important principles in object-oriented software engineering and is closely related to several other important concepts, including encapsulation, inheritance and polymorphism. ... These methods are used to reduce the complexity of the design and implementation process of software. | |
| | (e) | Advantages of Prototyping Model - In the development process of this model users are actively involved.The development process is the best platform to understand the system by the user.Earlier error detection takes place in this model.It gives quick user feedback for better solutions.It identifies the missing functionality easily. It also identifies the confusing or difficult functions.<br><br>Limitations of Prototyping Model- The client involvement is more | |



KIIT-DU/2018/SOT/Spring End Semester Examination-2018

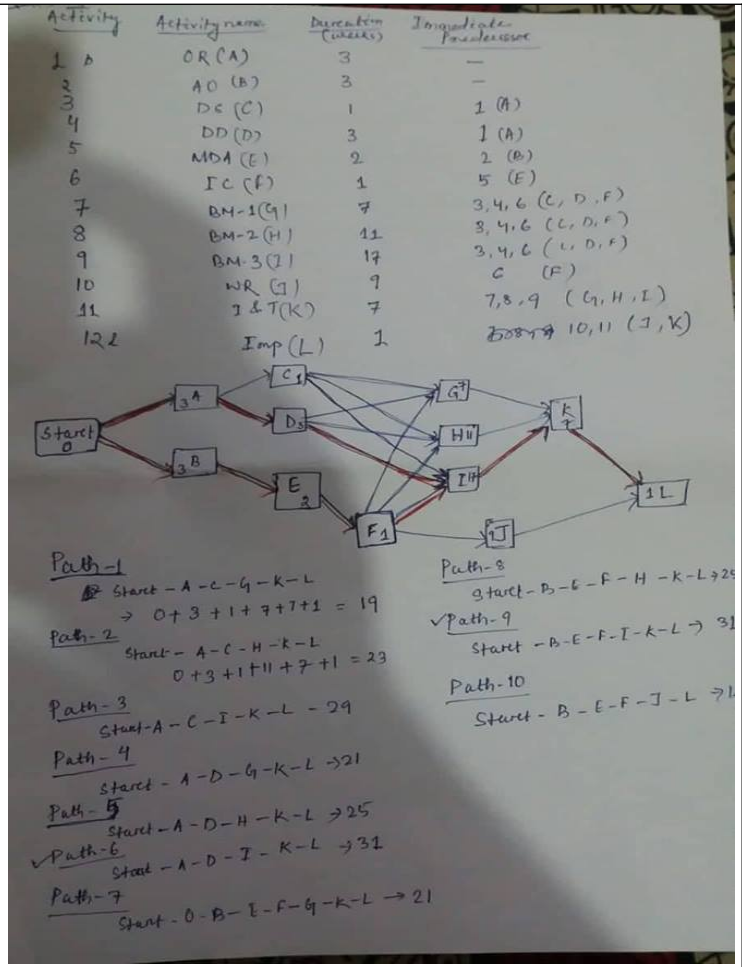
| | | | |
|---|---|---|---|
| | | and it is not always considered by the developer.It is a slow process because it takes more time for development.Many changes can disturb the rhythm of the development team.It is a throw away prototype when the users are confused with it. | |
| | (f) | False, The branch coverage based testing is stronger than the statement coverage based testing. Because, if all branches are covered, then automatically all statements will be covered. | |
| | (g) | Software testing is the process of checking a software whether any fault is there or not. But, debugging is the process of finding the location of the error or fault in the program. Types of Debugging<br>1. Brute Force Method<br>2. Backtracking<br>3. Cause Elimination method<br>4. Program slicing | |
| | (h) | Adaptive maintencance is the modification in software to make it usable in changed environment. In this case, a software product needs maintenance (porting) when customers: a) need the product to run on new platforms,or, on new operating systems, b) need the product to interface with new hardware or software.<br><br>Ex. The maintenance of software when the OS is changed from Windows 7 to Windows 10 | |
| | (i) | In FOD, the basic abstraction is functions and in OOD the basic abstraction is real world entities.<br>In FOD, software is developed using function and in OOD, software is developed using objects. In FOD, DFD and structure chart diagrams are using. But in OOD, UML diagrams are used. | |
| | (j) | | |

| **Inspection** | **Walkthrough** |
|---|---|
| Formal | Informal |
| Initiated by the project team | Initiated by the author |
| Planned meeting with fixed roles assigned to all the members involved | Unplanned. |
| Reader reads the product code. Everyone inspects it and comes up with defects. | Author reads the product coc up with defects or suggestion |
| Recorder records the defects | Author makes a note of defe by team mate |
| Moderator has a role in making sure that the discussions proceed on the productive lines | Informal, so there is no mode |

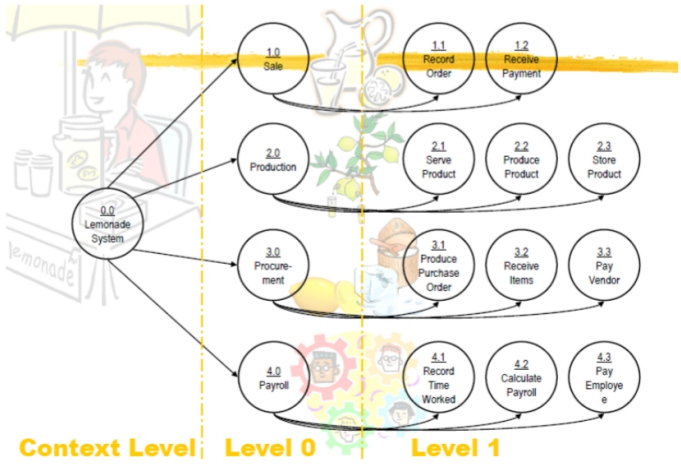


| | | | |
|---|---|---|---|
| 2. | (a) | | 1.5+1+1.5 |

| Node | ES | EF | LS | LF | ST (LS-ES)/(LF- |
|---|---|---|---|---|---|
| A | 0 | 3 | 0 | 3 | 0 |
| B | 0 | 3 | 0 | 3 | 0 |
| C | 3 | 4 | 15 | 16 | 12 |
| D | 3 | 6 | 3 | 6 | 0 |
| E | 3 | 5 | 3 | 5 | 0 |
| F | 5 | 6 | 5 | 6 | 0 |
| G | 6 | 13 | 16 | 23 | 10 |
| H | 6 | 17 | 12 | 23 | 11 |
| I | 6 | 23 | 6 | 23 | 0 |
| J | 6 | 15 | 21 | 30 | 15 |
| K | 23 | 30 | 23 | 30 | 0 |
| L | 30 | 31 | 30 | 31 | 0 |

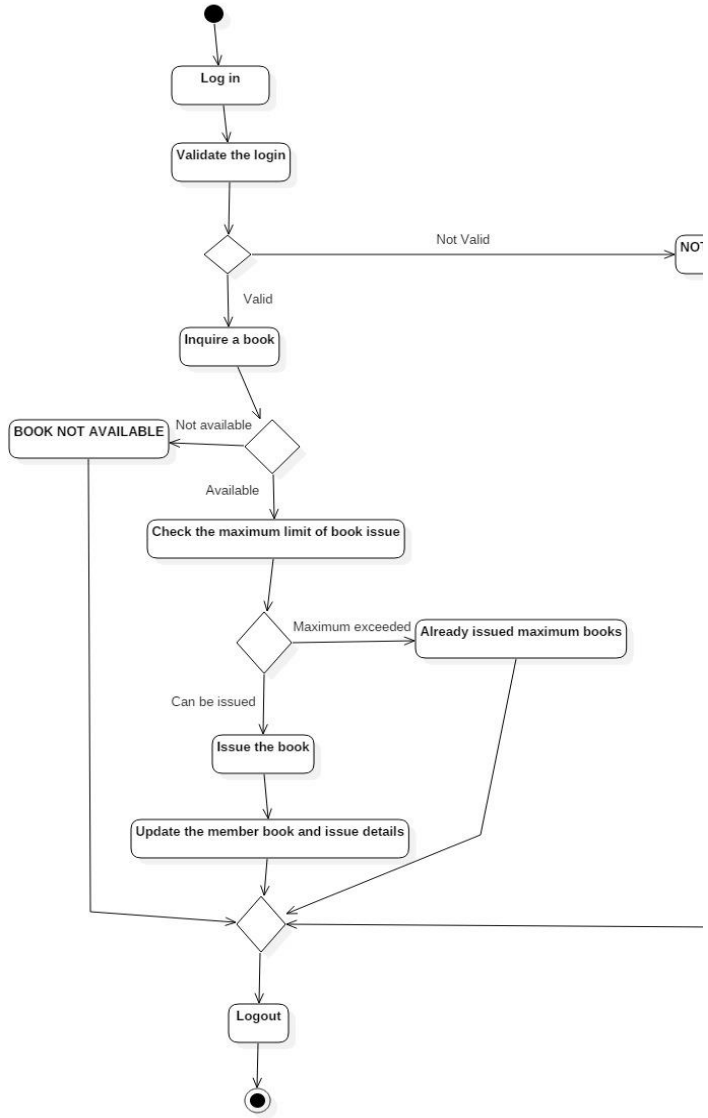
| | | | |
|---|---|---|---|
| | |  | |
| | (b) | i) Software Configuration Management. Text Book page 152. Section 3.14<br>ii)Team Structure : Text Book page 143 to 145<br>Section 3.11.2 | i)1+1<br>ii) 2 |
| | | | |
| 3. | (a) | SEI CMM Textbook page 520. Section 11.6 | 1+2+1 |
| | (b) | LOC and Function point<br>Text book page 100<br>Section 3.4 | 1+1+2 |



| | | | |
|---|---|---|---|
| | | | |
| 4. | (a) | Cohesion and Coupling<br>Text book page 226<br>Section 5.3 | 1+3 |
| | (b) | i)Lemonade System<br>  <br>ii)<br>Significance of Data Dictionary.<br>a) In order to manage the details in large-scale systems.<br>Most systems are ongoing and dynamic and management of all the descriptive details is<br>difficult, therefore an accurate and consistent recording technique is essential.<br>b) To communicate a common meaning for all of the elements in the system.<br>Simply making sure that for all elements, the meaning will remain consistent.<br>c) To document features of the system.<br>It is essential to document the circumstances under which data items occur. For example, | 2.5 + 1.5 |

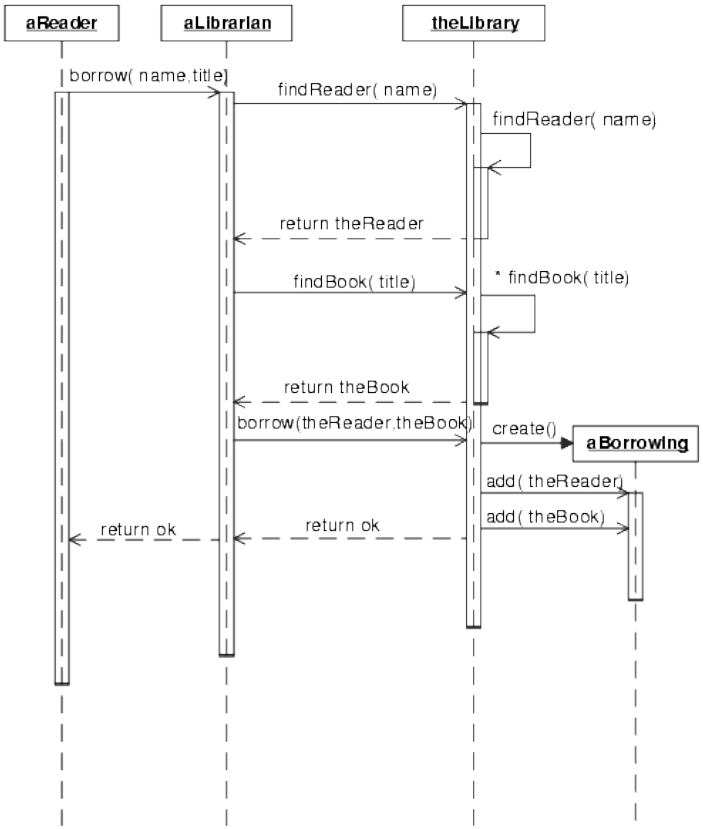
| | | | |
|---|---|---|---|
| | | what is the frequency of this process? Who has access to this datastore? Documenting these<br>features will produce a more complete and better understanding of the system for the analyst.<br>d) To locate errors and omissions in the system.<br>The data dictionary may reveal information that is incomplete and/or inaccurate. It may show<br>stores that are never accessed and/or processes that should be sub-divided, etc. | |
| | | | |
| 5. | (a) | Agile vs Waterfall Text Book page 65-67<br>Scrum Textbook page 70<br>Section 2.4 | 1.5 + 2.5 |
| | (b) | Black box testing<br>Text Book page 441<br>Section 10.6 | 0.5 +1.75 + 1.75 |
| | | | |
| 6. | (a) | Objects are the real-world entities that exist around us and it supports the basic concepts such as abstraction, encapsulation, inheritance, and polymorphism. Object oriented design mostly focuses on the objects that can exist for any system. Based on these objects only several other components like methods, relationships, interaction among the objects etc. can be further designed. Unified Modeling Language, UML, is a modeling language that provides a set of notations to create models of a system. These models are useful in documenting the design and result analysis. It provides the facility to represent the system into a model using different diagrams, known as UML diagrams. Most of these diagrams uses objects and so they provide better understanding of the system. This makes UML suitable for object oriented design.<br>Activity Diagram | 1+1.5+1.5 |



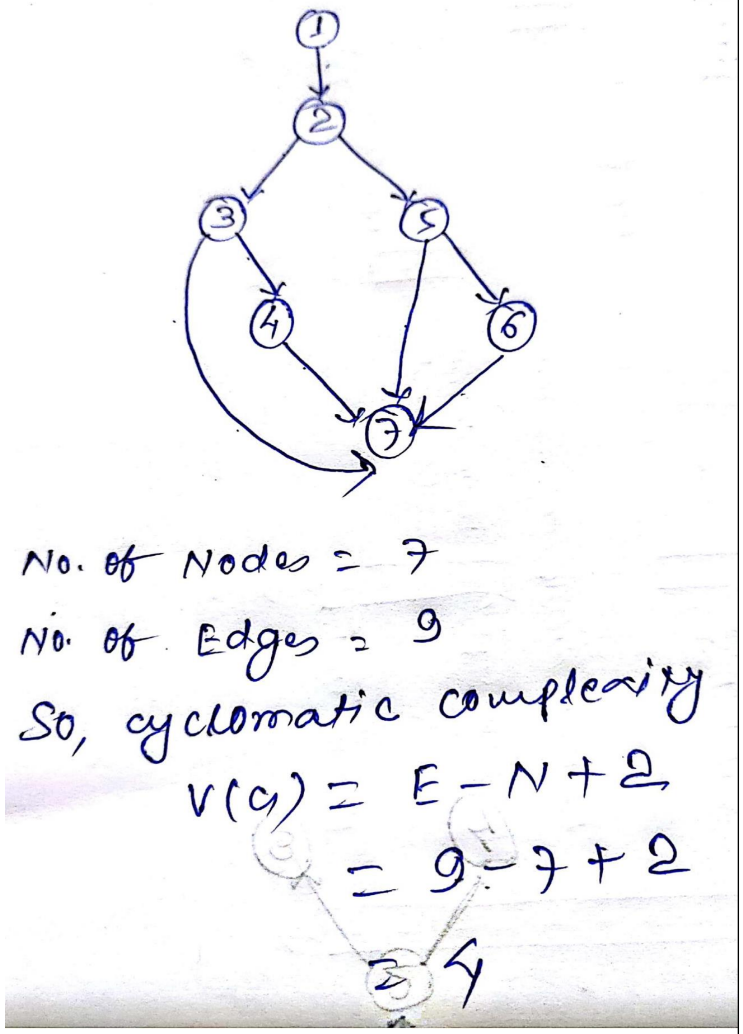



Please note “Checking maximum limit of book issue” is not mandatory here.

Sequence Diagram (example is given here, if student has drawn meaningful design should be awarded marks)

| | | | |
|---|---|---|---|
| | |  | |
| | (b) | Cyclomatic Complexity<br>Option 1 : Check the diagram<br>Option 2: Check the diagram<br>**Now, number of closed region = 3**<br>**Hence Ans = 3+1 = 4** | Drawing = 2<br>Determining the complexity =2 |



| | | | |
|---|---|---|---|
| | |  | |
| | | | |
| 7. | (a) | Software Reliability Metrics<br>Textbook page 503<br>Section 11.1.2 | All six metrics explained = 4 |
| | (b) | Software Reverse Engineering<br><br>Difference between Software Reverse engineering and Software | 1.5 + 2.5 |

| | | | |
|---|---|---|---|
| | | Reengineering:<br>While both refer to the further investigation or engineering of finished products, the methods of doing so, and the desired outcomes, are vastly different. Reverse engineering attempts to discover how something works, while re-engineeringseeks to improve a current design by investigating particular aspects of it.<br>Text Book page 547<br>Cosmetic changes : fig 13.2 page 548 | |
| | | | |
| 8. | | | 4 x 2 |
| | (a) | Validation and Verification :<br>Main points:<br><br>- Both techniques help to remove errors in a software.<br>- Verification is the process by which we determine whether the output of one phase of software development conforms to that of its<br>preceding phase.<br>-Validation is the process by which we determine whether a fully developed software product conforms its requirements specification.<br>Text Book page 435. Section 10.4 | 4 |
| | (b) | Pert Chart and Gantt Chart<br>PERT Charts:<br>- can be perceived as the sophisticated form of activity chart.<br>- can be used to determine the probabilistic time times for reaching different project milestones (including the final one).<br>- consists of boxes (activities) and arrows (task dependencies).<br>- Each task is annotated with 3 estimates: Optimistic(O), Most likely estimate(M), Worst case(W).<br><br>Gantt Charts:<br>- a form of bar chart where each bar represents an activity.<br>- the bars are drawn along a timeline.<br>- the length of each bar is proportional to the time duration planned for the concerned activity.<br>Examples MAY BE provided.<br><br>Text Book page 136, 137. section 3.10.4 | 2+2 |



| | | | |
|---|---|---|---|
| | (c) | UML diagrams : Text book page 316<br>Section 7.2 | Definition2+ Example 2 |
| | d) | Integration testing (top down and bottom up)<br>Text Book section 10.10<br>Page 460 | 2+2 |
| | | ***** | |