Reg. No.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

**SRM Institute of Science and Technology**

**SET D**

**College of Engineering and Engineering and Technology**

**School of Computing**

SRM Nagar, Kattankulathur – 603203, Chengalpattu District, Tamilnadu

**Academic Year: 2022-23 EVEN(Answer key)**

**Test: CLAT 2**  **Date: 3 April 2023**

**Course Code & Title: 18CSC206J** Software Engineering and Project Management **Duration:** 100 mins

**Year & Sem:** II Year / IV Sem. **Max. Marks:** 50

**Course Articulation Matrix: *(to be placed)***

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Course Outcomes** | **PO1** | **PO2** | **PO3** | **PO4** | **PO5** | **PO6** | **PO7** | **PO8** | **PO9** | **P10** | **P11** | **P12** |
| **CO1** | H | M | L |  |  |  |  |  | L | L | H | M |
| **CO2** | H | M | M | M | H |  |  |  | L | L | L | M |
| **CO3** | H |  | M |  | H |  |  |  | H | L | L | M |
| **CO4** |  | M |  |  | M |  |  | M | H | M | L |  |
| **CO5** |  |  |  |  |  | M | L | L |  | H | M |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Part – A (1 x 10 = 10 Marks) Answer all** | | | | | | |
| **Q. No** | **Question** | **Marks** | **BL** | **CO** | **PO** | **PI Code** |
| **1** | What is the main difference between software architecture and design patterns?  **A. Software architecture is concerned with the overall structure and organization of a software system, while design patterns focus on solving specific design problems.**  B. Software architecture is concerned with the implementation details of a software system, while design patterns focus on the overall design of the system.  C. Software architecture is concerned with the development of individual components of a software system, while design patterns focus on the overall structure of the system.  D. There is no difference between software architecture and design patterns. | 1 | 2 | 2 | 2 | 2.1.1 |
| **2** | Which of the following is an example of the top-down approach?  A. Starting a business without a clear business plan  **B. Conducting a survey to understand customer needs**  C. Designing a website without considering user experience  D. Developing a product without conducting market research | 1 | 1 | 2 | 1 | 2.1.1 |
| **3** | A traditional component, also called a module, resides within the \_\_\_\_\_\_\_.  **A. Software architecture**  B. Software testing  C. Software debugging  D. Software system | 1 | 1 | 2 | 1 | 2.1.1 |
| **4** | Which of the following is not a design principle that allow the user to maintain control?  A. Provide for flexible interaction  B. Allow user interaction to be interrupt-able and undo-able  **C. Show technical internals from the casual user**  D. Design for direct interaction with objects that appear on the screen | 1 | 1 | 2 | 3 | 1.3.1 |
| **5** | ‘The usage scenarios (use-cases) created as part of interaction analysis define the operations that will be applied to WebApp content and imply other processing functions’- this refers to which one of the following?  A. Content Analysis  B. Interaction Analysis  **C. Functional Analysis**  D. Configuration Analysis | 1 | 2 | 2 | 3 | 1.3.1 |
| **6** | In Software construction, increases software code reuse and thus enhances the productivity of developers and \_\_\_\_\_\_\_\_\_\_\_\_\_\_.  A**. Code Readability**  B. Code Clarity  C. Code Productivity  D. Software Reuse | 1 | 2 | 3 | 2 | 1.7.1 |
| **7** | |  | | --- | | SOA has been evolved recently for the purpose of \_\_\_\_\_\_\_\_\_ | | A. Software Construction | | B. Inspection | | **C.Code Reuse** | | D.Code Review | | 1 | 1 | 3 | 5 | 3.5.2 |
| **8** | When version control management is not properly managed then many \_\_\_\_\_\_ would end up handling incorrect versions of source code then may result in a huge rework at the end.  A. Presales  B. Tester  C. Designer  **D. Developer** | 1 | 2 | 3 | 3 | 1.3.1 |
| **9** | Identify how the generalization is implemented in Object Oriented programming languages?  **A. Inheritance**  B. Polymorphism  C. Encapsulation  D. Abstract Classes | 1 | 1 | 3 | 4 | 4.4.2 |
| **10** | Which of the following tasks is not part of configuration management  A. Change Control  B. Reporting  **C. Statistical quality Control**  D. Version Control | 1 | 1 | 3 | 3 | 3.6.1 |
|  | | | | | | |
| Reg. No.   |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |   **SRM Institute of Science and Technology**  **SET D**  **College of Engineering and Engineering and Technology**  **School of Computing**  SRM Nagar, Kattankulathur – 603203, Chengalpattu District, Tamilnadu  **Academic Year: 2022-23 EVEN**  **Test: CLAT 2**  **Date: 3 April 2023**  **Course Code & Title: 18CSC206J** Software Engineering and Project Management **Duration:** 100 mins  **Year & Sem:** II Year / IV Sem. **Max. Marks:** 50  **Part – B**  **(4 x 4 =16 Marks) Answer any 4** | | | | | | |
| **Q. No** | **Question** | **Marks** | **BL** | **CO** | **PO** | **PI Code** |
| **11** | Identify the characteristics of a good software design and relate each of them with a web-based application, for instance, an online shopping system.  **Answer:**  • Open Architecture, Modularity, Robustness, Security, Scalability, Simplicity  • Above characteristics to be related to online shopping system and justified. | 4 | 2 | 2 | 2 | 2.1.1 |
| **12** | What are some benefits of refactoring in software development? Provide an example of how refactoring can improve code quality.  Answer:  Refactoring is a process of restructuring existing code without changing its external behavior. The main goal of refactoring is to improve the readability, maintainability, and extensibility of the code. Here are some benefits of refactoring in software development:   1. Improved Code Quality: Refactoring can improve the quality of the code by making it more modular, readable, and easier to understand. This, in turn, leads to a reduction in the number of bugs and makes the codebase more maintainable and extensible. 2. Enhanced Performance: Refactoring can improve the performance of the code by removing redundant code, reducing complexity, and improving the design of the system. 3. Increased Team Productivity: Refactoring can increase the productivity of the development team by reducing the time required to understand and modify the code. This, in turn, can lead to faster development cycles and improved overall team efficiency. 4. Reduced Technical Debt: Refactoring can reduce technical debt by improving the quality of the code and reducing the number of bugs and other issues. This, in turn, reduces the effort required for maintenance and future enhancements. | 4 | 1 | 2 | 3 | 5.4.2 |
| **13** | Illustrate the Source Code Production (Conversion) from Software Design.  Answer:   * Developers are given software design specifications in the form of use cases, flow diagrams, UI mock ups, etc., and they are supposed to write a code so that the built software matches these specifications. * Converting the specifications into software code is totally dependent on the construction team. * How well they do it depends on their experience, skills and the process they follow to do their job. * Apart from these facilities, they also need some standards in their coding so that the work is fast as well as has other benefits like maintainability, readability and reusability (Figure-Source Code Production (Conversion) from Software Design). | 4 | 1 | 2 | 1 | 2.3.1 |
| **14** | Assume an automatic code generator is used to construct a large number of codes. This drastically reduces the time to create code but at the same time, it is poorly documented which also leads to poor maintenance. In this scenario, is it worth using the automatic code generator? Also, what is the impact of poor documentation?  **Answer:**  No, it is not worth to use automatic code generator. There is no excuse for poor code. If a programmer wants to be clever and use automatic code generation then they should take the time to ensure that the code generated is good code. It's only pushing the buck down the rode and when it comes to code production, the buck should stop at the developer who wrote it.  When a new coder is brought into a team, poor documentation will not allow the new coder to understand the purpose/flow of the code. | 4 | 2 | 3 | 3 | 1.3.1 |
| **15** | Summarize why software construction is referred to as one of the laborious stages in the software development lifecycle.  Answer:   * The software construction phase is one of the most labor-intensive phases in software development cycle as this phase generates the complete source code of the application. * Apart from source code, documentation is also made so that when any maintenance is required on the built application, the source code could be well understood, and changing any source code will be easy. * Review reports are also generated after reviews are conducted. | 4 | 1 | 3 | 2 | 1.2.1 |
| **Part – C (2 x 12 =24 Marks) Answer all** | | | | | | |
| **16.a** | An ERP is a big software system that contains all of the typical administrative functionalities most of organizations use: accounts payable, accounts receivable, stock management, HR management, production management, provider management, purchasing, treasury, finance, accounting, etc. These modules contain standard, predefined functionalities that are adapted when the software is implemented. Suggest the type of architecture suitable for the above case study.  Answer:  **Client-Server Architecture:**  ERPs are usually built on a clientserver architecture since they have a core module that manages the functionalities and communicates with the database.  They are clients of the system (the users) scattered over the whole organization. These users have different devices, hardware, and channels of communication, and they all need to  speak with the server.  Front-End: This is the piece of software that interacts with users, even if they are on different platforms with different technologies. Any front-end module in a client-server architecture is designed to interact with all existing devices on the market. This level contains the login screens, menus, data screens, and reports that give and take information to/from users. For example, most development tools and frameworks allow the creation of one version of a program that works for PCs, tablets, and phones. Application server: This is the server where the software modules of the application are installed. It connects to the database (called back-end) and interacts with users (calledfront-end). The application server is like the waiter on our restaurant example. Database server: This server contains the tables, indexes, and data managed by the application. Searches and insert/delete/update operations are executed here. 1. The user makes a request via the front-end, for example, “Bring me all customer invoices from January 1st until today, for customers that have bought product X.” 2. The application server gets this request and sends it to the database. The database server executes this request and sends it to the application server, which sends the result to the client using the front-end module. There are several advantages to using client-server architecture: As you can see in the example, the client-server architecture separates hardware, software, and functionality of the system. For instance, if a software adaptation is needed in a particular country, i.e., a change in functionality is necessary. It can be adapted in the system without having to develop a version for phones, tablets, or laptops. Since it separates among hardware, software, and functionality of the system, only the frontend must be adapted to communicate with different devices | 12 | 3 | 2 | 3 | 5.4.2 |
| **or** | | | | | | |
| **16.b** | **Navigation modeling – Brief the need for this modeling and list the various outcomes expected out of this modeling as part of a web application design. Give an example system.**  **Answer:**  A “navigation model” describes the paths to each user goal on a website. Usually a product will have more than one user, and each user may have multiple goals, so models like this can be pretty extensive.  A web designer’s job is to help users find their way to what they’re looking for. It can be easy to put the needs of your users to one side, but knowing your users, understanding their roles, goals, motives and behavior will confirm how you structure your navigation model.  Expected outcomes:   * Should certain elements be easier to reach (require fewer navigation steps) than others? What is the priority for presentation? * Should certain elements be emphasized to force users to navigate in their direction? * How should navigation errors be handled? * Should navigation to related groups of elements be given priority over navigation to a specific element. * Should navigation be accomplished via links, via search-based access, or by some other means? * Should certain elements be presented to users based on the context of previous navigation actions? * Should a navigation log be maintained for users? * Should a full navigation map or menu (as opposed to a single “back” link or directed pointer be available at every point in a user’s interaction? * Should navigation design be driven by the most commonly expected user behaviors or by the perceived importance of the defined WebApp elements? * Can a user “store” his previous navigation through the WebApp to expedite future usage? * For which user category should optimal navigation be designed?   The navigation of any website or web portal can be discussed as an example. | 12 | 3 | 2 | 3 | 2.5.2 |
| **17.a** | Categorize the various coding standards and explain their characteristics with examples  Answer:  **Coding Standards**   * At any time, a code written by a developer will always be different from that written by any other developer. * This poses a challenge in terms of comprehending the code while reusing the code, maintaining it, or simply reviewing it. * A uniform coding standard across all construction teams working on the same project will make sure that these issues can be minimized if not eliminated (Figure below - Software Construction Characteristics). * Some of the coding standards include standards for code modularity, clarity, simplicity, reliability, safety and maintainability.     **Coding Standards – Modularity**   * The produced software code should be modular in nature. * Each major function should be contained inside a software code module. * The module should contain not only structure, but it should also process data. * Each time a particular functionality is needed in the software construction, it can be implemented using that particular module of software code. * This increases software code reuse and thus enhances productivity of developers and code readability.   **Coding Standards – Clarity**   * The produced code should be clear for any person who would read the source code. * Standard naming conventions should be used so that the code has ample clarity. * There should be sufficient documentation inside the code block, so that anybody reading the code could understand what a piece of code is supposed to do. * There should also be ample white spaces in the code blocks, so that no piece of code should look crammed. White spaces enhance readability of written code.   **Coding Standards – Simplicity**   * The source code should have simplicity and no unnecessary complex logic; improvisation should be involved, if the same functionality can be achieved by a simpler piece of source code. * Simplicity makes the code readable and will help in removing any defects found in the source code. * Simplicity of written code can be enhanced by adopting best practices for many programming paradigms. * For instance, in the case of object-oriented programming, abstraction and information hiding add a great degree of simplicity. * Similarly, breaking the product to be developed into meaningful pieces that mimic real life parts makes the software product simple.   **Coding Standards – Reliability**   * Reliability is one of the most important aspects of industry strength software products. * If the software product is not reliable and contains critical defects, then it will not be of much use for end users.   **Coding Standards – Reliability**   * Reliability of source code can be increased by sticking to the standard processes for software construction. * During reviews, if any defects are found, they can be fixed easily if the source code is neat, simple, and clear. * Reliable source code can be achieved by first designing the software product with future enhancement in consideration as well as by having a solid structure on which the software product is to be built. * When writing pieces of source code based on this structure, there will be little chance of defects entering into the source code. * Generally during enhancements, the existing structure is not able to take load of additional source code and thus the structure becomes shaky. * If the development team feels that this is the case, then it is far better to restructure the software design and then write a code based on the new structure than to add a spaghetti code on top of a crumbling structure.   **Coding Standards – Safety**   * Safety is important, considering that software products are used by many industries where human lives are concerned and that human lives could be in danger because of faulty machine operation or exposure to a harmful environment. * In these industries, the software product must be ensured to operate correctly and chances of error are less than 0.00001%. * Industries like medicine and healthcare, road safety, hazardous material handling need foolproof software products to ensure that either human lives are saved (in case of medicine and healthcare) or human lives are not in danger. * Here the software code must have inbuilt safety harnesses.   **Coding Standards – Maintainability**   * As it has been pointed out after several studies, maintenance costs are more than 70% of all costs including software development, implementation, and maintenance. * To make sure that maintenance costs are under limit during software construction, it should be made sure that the source code is maintainable.   It will be easy to change the source code for fixing defects during maintenance. | 12 | 2 | 3 | 3 | 1.3.1 |
| **Or** | | | | | | |
| **17.b** | i) Evaluate in detail the components that are used in software reuse. (6)  Answer:  Components and reuse The concept of component is still evolving, although  the general thought is that most software artifacts can be considered as components.  Definitions of a component are: A component is a subsystem, use case, actor, or any  object class; A component is a good abstraction for higher-level design, with access  restricted by visibility rules; A component is not bound to any specific application; A  component is a high-quality product due to careful design and testing; A component  is packaged for reuse with well-designed interface, documentation, A component is  general so that it can be used in several places; A component is specialized when  used. This emphasizes components as high-quality, generic software products with  well-defined public interfaces, designed to be used in many contexts. Another  possible, though much simpler, definition of a component is: A component is a  package of functionality, deployed within specific technology framework. Here,  package of functionality refers to a high-level, reusable abstraction with one or more  public interfaces. This emphasizes that a component is not necessarily restricted to  object technology. In fact, components have been in existence for decades in the form  of high-quality libraries of routines and functions. A reusable component is any  component, developed for reuse and actually used in more than one context.  Reusable components can be code, design specifications, processes, methods,  documentation, system or subsystems, models, patterns, frameworks, classes, object  implementations etc. Before something can be reused, we must assure that it is ready  for reuse. Before the reuse is possible, we have to: locate the component, know, what  the component is doing and know, how to reuse the component.  ii)Illustrate with appropriate reasons why configuration management plays a significant role in software construction. (6)  Answer:  - Configuration management plays an important role in the construction phase.  - Due to changes in requirements and design, an already developed source code  needs to be changed.  - So it happens that the development team ends up with many versions of a source  code during the project.  - If the version control management is not handled properly, then many developers  may start working on a wrong version of source code, and thus a lot of rework  may be needed in the end.  - There is one more dimension to configuration management for the construction  phase.  - During construction, many software builds are maintained for different versions  of the product being developed.  - These builds can break if a bad piece of code is checked into the build by any  developer.  - When the build is broken, then no other developer can check in his code.  - Thus, development is halted until the build is rebuilt with the correct code.  - Imagine what may happen in the case of distributed teams located at far-flung  locations with different time zones and a central build is being maintained.  - It will be difficult to communicate and manage the build process in such a  scenario.  - In such scenarios, smoke test application can be deployed, which can run  whenever a new code is checked-in in the build.  - If the smoke test fails, that means the build has failed and thus the automated  system can e-mail the build information to concerned people.  - If the build fails, then the developer who had checked-in the code gets the  message and immediately tries to fix the build.  - Once the build is fixed, then other developers can check-in their code.  - Thus, configuration management plays an important role in the construction  phase. | 12 | 2 | 3 | 3 | 1.3.1 |

**\*Performance Indicators are available separately for Computer Science and Engineering in AICTE examination reforms policy.**

**Course Outcome (CO) and Bloom’s level (BL) Coverage in Questions**

**Approved by the Audit Professor/Course Coordinator**