1. Define Software Engineering:

What is software engineering, and how does it differ from traditional programming?

Software engineering is a holistic approach to designing and maintain software. It makes use of engineering principles to ensure that software meets user's needs, is reliable and efficient. It differs from traditional programming as traditional programming focus more on code writing whereas in software engineering encompasses an entire life cycle called SDLC.

2. Software Development Life Cycle (SDLC):

Explain the various phases of the Software Development Life Cycle. Provide a brief description of each phase.

The Software Development Life Cycle has 5 steps. The 1st being Requirements which is the process of gathering and documenting the needs and system requirements of the system to be developed. The 2nd is Design in this step high level and details designs are created, this would include software architecture, implementation and writing code. 3rd is Testing which involves conducting various tests to ensure it meets quality standards and functional requirements. 4th is Deployment in this step the software is released. 5th Maintenance which involves providing ongoing support, updates and enhancements to the software.

3. Agile vs. Waterfall Models:

Compare and contrast the Agile and Waterfall models of software development. What are the key differences, and in what scenarios might each be preferred?

In Waterfall models, designers use a sequential approach to software development in this scenario each if the SDLC steps are dependent on one another like there is no designing a software without getting the requirements first. This model is best used in projects where the requirements are clear, stable and unlikely to change throughout the design process. An example would be developing software for an educational training system where the curriculum and objectives do not change . Whereas in Agile it is focused more on flexibility, collaboration and responding to changes, this approach is best applied to long term projects. An example would be developing software for product development as in this model products benefit from regular releases and updates to keep uses engaged, and real-life world example

like CRM (customer relationship management) systems, that continually rely on user feedback and company relies on their reliability to satisfy customers.

4. Requirements Engineering:

What is requirements engineering? Describe the process and its importance in the software development lifecycle.

Requirement Engineering id the process of defining, documenting and maintaining the requirements for a software project. In this step the engineer has to understand the user's needs so that the product to be developed can match those needs. This process has 5 stages which can include, gathering on information (can be through interviewing customers), analysis (involves understanding requirements and organizing the gathered information so that it makes sense, specification (which is documenting the requirements clearly and precisely), validation (which involves checking the gathered information against the set out requirements ensure they are correct), Management(which is ensuring that requirements are up-to-date as the project progresses.)

5. Explain the concept of modularity in software design. How does it improve maintainability and scalability of software systems?

Modularity is a principle of breaking down a software system into smaller, manageable and independent parts that are called modules. Each module represents a certain functionality or a set of related functions. This improves the maintainability as small modules are easier to test and debug, in this way issues can be isolated within a model, and it makes finding and fixing bugs easier. In terms of scalability, it can lead to increase production as different teams can work on different modules simultaneously.

6. Testing in Software Engineering:

Describe the different levels of software testing (unit testing, integration testing, system testing, acceptance testing). Why is testing crucial in software development?

Testing is important because it identifies and fixes bugs, and it ensures that the software meets requirements, enhances quality and reliability, prevents costly issues post deployment and ensures users satisfaction. The 4 levels of testing are unit testing which involves testing individual components or functions of the software to ensure that they work correctly, integration testing this stages test the interaction between integrated modules to identify issues in their interfaces, system testing which test the system in its entirety and test whether the system meets the specified requirements and lastly acceptance testing which tests the software from the end-users- perspective to make sure it meets the requirements and is ready for deployment.

7. Version Control Systems:

What are version control systems, and why are they important in software development? Give examples of popular version control systems and their features.

Version control are known as source control, it is the practice of tracking and managing software changes to code. They are important as they help facilitate collaboration, maintain a history of changes, provide backup and recovery options and supports branching and merging. Popular versions of VC are Git, Subversion, Mercurial and Perforce.

8.Software Project Management:

Discuss the role of a software project manager. What are some key responsibilities and challenges faced in managing software projects?

Project managers play an important role in project from start to finish. They are responsible for planning, team management, risk mitigation, budget control and quality assurance. Challenges they face are things such as scope creep which is uncontrolled changes to project scope, challenge of meeting project deadlines whilst ensure quality is maintained, limited resources, which can take the form of team members, tools and budget, managing diverse teams with different skills and personalities and work styles and can have many challenges with balancing the needs and expectations of various stakeholders whilst completing project in time.

9. Software Maintenance:

Define software maintenance and explain the different types of maintenance activities. Why is maintenance an essential part of the software lifecycle?

This is the process of modifying and updating software applications after they have been deployed. This is the last step in the SDLC lifecycle and an important one as it helps in correcting faults, improve performance as well as to adapt to change environment thus keeping the software useful and relevant to intended users.

10. Ethical Considerations in Software Engineering:

What are some ethical issues that software engineers might face? How can software engineers ensure they adhere to ethical standards in their work?

One of the issues is that of data privacy, as they can create systems that can exploit user data without bearing the brunt and the organization being labile for it, bias is another issue, as software can only learn from the training it is given and if the developers are biased it can creep into a system undetected. Another can be software developers favouring the release of software without addressing security concerns which should have been addressed during development and now are done after, which can cause a lot of problems. Software developers can ensure they adhere to ethical standards by writing

code that is clean, well-documented and in the process of doing so to ensure that they prioritize security measures, implementing robust authentication and mechanism to ensure they protect user data.

References:

1. Author: Kamil JeDreko

Date: 06/07/2023

Link: https://softwaremind.com/blog/software-requirements-engineering-the-driving-force-behind-successful-and-efficient-it-

2. Author: George Lawton

   Date: 22/12/2020

   Link: https://www.techtarget.com/searchsoftwarequality/tip/5-examples-of-ethical-issues-in-software-development

3. Author: n/a

   Date: n/a

Link: https://www.atlassian.com/git/tutorials/what-is-version-control#:~:text=Version%20control%20systems%20are%20software,teams%20work%20faster%20and%20smarter.

4. Link: https://chatgpt.com/