**ANSWERS TO SOFTWARE ENGINEERING ASSIGNMENT**

**What is software engineering?**

Software engineering is an engineering approach to software development. A practitioner, a software engineer, applies the engineering design process to develop software. The terms programmer and coder overlap software engineer, but they imply only the construction aspect of typical software engineer workload.

**How does software engineering differ from traditional programming?**

 Software engineering often follows an iterative development process, where software is continuously refined and improved based on user feedback. Traditional engineering disciplines typically follow a more linear process, where designs are finalized before construction begins.

**Explain the various phases of software development cycle**

**Phase 1 Planning:** The initial stage of software development, planning, involves defining the software’s purpose and scope, much like pinpointing our destination and plotting the best route. We uncover the tasks at hand during this phase and strategize for efficient execution.

**Phase 2 Requirement analysis:** Requirements Analysis, seeks to identify and record the precise requirements of the final users. In this phase, the team is looking to answer, “What are the expectations of our users from our software?” This is called requirements gathering.

**Phase 3 Design:** The Design phase is all about building the framework. The development team is responsible for software engineering and outlines the software’s functionality and aesthetic. This ultimately results in the software product. The emphasis lies on outlining the software’s structure, navigation, user interfaces, and database design. This phase ensures that the software is user-friendly and performs its tasks efficiently.

**Phase 4 Coding:** This development phase aims to develop software that is functional, efficient, and user-friendly. Developers use an appropriate programming language, Java or otherwise, to write the code, guided by the SDD and coding guidelines. This document, acting as a roadmap, ensures the software aligns with the vision set in earlier phases.

**Phase 5 Testing:** Consider the Testing phase of the SDLC as a stringent quality inspection on a production line. It is when vulnerabilities are uncovered. Software testing involves a thorough examination of the software for any bugs or glitches that might have slipped through during coding. The aim is to ensure flawless software operation before it reaches the end-users. And even identify opportunities for enhancement.

**Phase 6 Deployment:** After crafting a product with precision, it’s time to present it to the users by pushing to the production environment. The Deployment phase involves rolling out the meticulously tested and fine-tuned software to its end-users. The Deployment phase doesn’t signal the end, but rather a notable milestone. It signifies the shift from a project phase to a product phase, where the software begins to fulfill its purpose.

**Phase 7 Maintenance:** Maintenance tasks encompass frequent software updates, implementing patches, and fixing bugs. User support is also a crucial component, offering help and guidance to users facing difficulties with the software.

**Differences between agile and waterfall models?**

|  |  |  |
| --- | --- | --- |
|  | **Agile** | **Waterfall** |
| **Flexibility** | Open to changes | Resistant to changes | |
| **Documentation** | Minimal documentation | Each step laid out | |
| **Speed** | A few weeks; several steps at a time | Several months; one step at a time | |
| **Roles** | Team members organize themselves | Duties assigned to team members; led by project manager | |
| **Communication** | Daily, informal check-ins | Infrequent but formal check-ins | |
| **Budget** | Flexible | Fixed | |

**Similarities between agile and waterfall models**

Despite their differences, agile and waterfall testing also have some similarities. Both methods aim to ensure the quality and functionality of the software, and to satisfy the customer's requirements and expectations. Both methods use various types of testing techniques and tools, such as unit testing, integration testing, system testing, regression testing, performance testing, automation testing, etc. Both methods also follow some common principles and practices of software testing, such as test design, test execution, test reporting, test management, etc.

**What is requirement engineering?**

**Requirements engineering** (**RE**) is the process of defining, documenting, and maintaining requirements in the engineering design process. It is a common role in systems engineering and software engineering. The first use of the term *requirements engineering* was probably in 1964 in the conference paper "Maintenance, Maintainability, and System Requirements Engineering",but it did not come into general use until the late 1990s with the publication of an IEEE Computer Society tutorial in March 1997 and the establishment of a conference series on requirements engineering that has evolved into the International Requirements Engineering Conference**.**

**Processes of requirement engineering?**

**1. Feasibility Study**

The feasibility study mainly concentrates on below five mentioned areas below. Among these Economic Feasibility Study is the most important part of the feasibility analysis and the Legal Feasibility Study is less considered feasibility analysis.

### 2. Requirements Elicitation

It is related to the various ways used to gain knowledge about the project domain and requirements. The various sources of domain knowledge include customers, business manuals, the existing software of the same type, standards, and other stakeholders of the project.

### 3. Requirements Specification

This activity is used to produce formal software requirement models. All the requirements including the functional as well as the non-functional requirements and the constraints are specified by these models in totality. During specification, more knowledge about the problem may be required which can again trigger the elicitation process.

### 4. Requirements Verification and Validation

Requirements verification and validation (V&V) is the process of checking that the requirements for a software system are complete, consistent, and accurate and that they meet the needs and expectations of the stakeholders. The goal of V&V is to ensure that the software system being developed meets the requirements and that it is developed on time, within budget, and to the required quality.

### 5. Requirements Management

Requirement management is the process of analyzing, documenting, tracking, prioritizing, and agreeing on the requirement and controlling the communication with relevant stakeholders.

**Importance of requirement engineering?**

1. It helps to ensure that the final product meets the needs of the stakeholders and users.
2. Properly defined and managed requirements serve as a foundation for the design and development of the system.
3. It helps to reduce costs and improve the quality of the final product by identifying any ambiguities or inconsistencies early on in the development process.

Software Design Principles refer to a set of guidelines and methodologies that help software developers create high-quality, efficient, and maintainable software systems. These principles serve as a foundation for designing software architecture and provide a framework for making important design decisions during the development process. By adhering to these principles, software engineers can ensure that their software is scalable, reusable, and adaptable to changing requirements.

**What is the concept of modularity in software design?**

Modularity in software design typically involves identifying potential areas of change or evolution in a system and isolating these into separate modules. Each module is responsible for a specific aspect of the systems functionality and interacts with other modules through interfaces.

**How does software design principles improve maintainability and scalability of software systems?**

Software Design Principles refer to a set of guidelines and methodologies that help software developers create high-quality, efficient, and maintainable software systems. These principles serve as a foundation for designing software architecture and provide a framework for making important design decisions during the development process. By adhering to these principles, software engineers can ensure that their software is scalable, reusable, and adaptable to changing requirements.

**Describe the different levels of software testing?**

1. [**Unit Testing**](https://www.geeksforgeeks.org/unit-testing-software-testing/)**:** In this type of testing, errors are detected individually from every component or unit by individually testing the components or units of software to ensure that they are fit for use by the developers. It is the smallest testable part of the software.
2. [**Integration Testing**](https://www.geeksforgeeks.org/software-engineering-integration-testing/)**:** In this testing, two or more modules which are unit tested are integrated to test i.e., technique interacting components, and are then verified if these integrated modules work as per the expectation or not, and interface errors are also detected.

**System Testing:** In system testing, complete and integrated Software are tested i.e., all the system elements forming the system are tested as a whole to meet the requirements of the system.

1. [**Acceptance Testing**](https://www.geeksforgeeks.org/acceptance-testing-software-testing/)**:** This is a kind of testing conducted to ensure that the requirements of the users are fulfilled before its delivery and that the software works correctly in the user’s working environment.

**Why is testing crucial in software development?**

* **Identifies defects early.** Developing complex applications can leave room for errors. Software testing is imperative, as it identifies any issues and defects with the written code so they can be fixed before the software product is delivered.
* **Improves product quality.** When it comes to customer appeal, delivering a quality product is an important metric to consider. An exceptional product can only be delivered if it's tested effectively before launch. Software testing helps the product pass quality assurance (QA) and meet the criteria and specifications defined by the users.
* **Increases customer trust and satisfaction.**Testing a product throughout its development lifecycle builds customer trust and satisfaction, as it provides visibility into the product's strong and weak points. By the time customers receive the product, it has been tried and tested multiple times and delivers on quality.

**What are version control systems and why are they important?**

The process of monitoring and managing changes to software code is known as version control, also sometimes referred to as revision control or source control systems. Software technologies called version control systems assist software development teams in tracking changes to source code over time.

**Importance of version control system in software development**

* Managing and protecting the source code
* Merging and branching
* Traceability
* Comparing earlier versions of the code

**Examples of version control system and features**

**Github:**

GitHub helps software teams to collaborate and maintain the entire history of code changes. You can track changes in code, turn back the clock to undo errors and share your efforts with other team members.

**GitLab:**

GitLab comes with a lot of handy features like an integrated project, a project website, etc. Using the continuous integration (CI) capabilities of GitLab, you can automatically test and deliver the code.

You can access all the aspects of a project, view code, pull requests, and combine the conflict resolution.

**Mecurial:**

Mercurial is known for its efficiency in handling projects of all sizes. It is a free and distributed control management service that provides a simple and intuitive user interface.

**Bitbucket:**

Bitbucket is a part of the Atlassian software suite, so it can be integrated with other Atlassian services including HipChat, Jira, and Bamboo. The main features of Bitbucket are code branches, in-line commenting and discussions, and pull requests.

**Discuss the role of software project manager**

In software, project manager provide consistent leadership and technical expertise to help their teams complete projects correctly and on time. During the software development process, project managers may help their teams design, execute, monitor and complete their work to ensure the clients receive high-quality software products. Software project managers typically lead the production of the deliverable software and the employees performing the project.

**What are the key responsibilities and challenges faced in software project manager?**

**Challenges are:**

* Risk management
* Project management
* Progress monitoring and reporting
* Quality Assurance
* Develop project management skills
* Resource allocation

**Responsibilities are:**

* Involves with the senior managers in the process of appointing team members.
* Builds the project team and assigns tasks to various team members.
* Responsible for effective project planning and scheduling, project monitoring and control activities in order to achieve the project objectives.
* Acts as a communicator between the senior management and the development team and internal and external stakeholders.
* Effectively resolves issues that arise between the team members by changing their roles and responsibilities.
* Modifies the project plan (if required) to deal with the situation.

**What is software maintenance?**

Software maintenance is the process of modifying, updating, and enhancing software applications after they have been delivered to the end users. It involves making changes to the software to correct defects, improve performance, adapt it to changes in the operating environment, or add new features in response to changing user requirements.

**What are the types of maintenance activities?**

**Why is maintenance an essential part of the software lifecycle?**

Software maintenance is as important as the development itself. If you regularly maintain software, you will ensure your trouble-free use and good performance, less problems and adaptation to changes in the business environment. According to statistics, when it comes to software, 60% of the cost associated with maintenance. Even from the total maintenance costs - 60% of improving the solution.

**What are some ethical issues that software engineers might face?**

* Data privacy
* Accessibility
* Addictive design
* Algorithimic bias
* Software security

**How can software engineers ensure they adhere to ethical standards in their work?**

In an era of unprecedented technological advancement, software engineers hold the key to a better future. By adhering to a set of ethical principles such as transparency, inclusivity/accessibility, user-centric approach etc, they can drive positive change, promote accountability, and ensure that technology contributes to the well-being of individuals and society. As guardians of the digital realm, software engineers have a profound responsibility to harness their skills for the greater good, ultimately shaping a world where technology serves as a force for progress and human flourishing.