Page No.
Date / 20

Q-1 System and software development stages?
Q-2 Explain waterfall & Spiral Model.
Q-3 Which is more important process or product.
Q-4 Difference between ER and DFD diagram.
Q-5 Difference b/w structural and Non-Structural testing.
Q-6 Describe Prototyping ~~tec~~ techniques.
Q-7 Describe the software testing and also the object of testing.
Q-8 Difference b/w white box and black box.
Q-9 Define Cyclomatic Complexity.
Q-10 Write about Boundary value testing and Independent testing.
Q-11 Write brief on Equivalence testing and Integration testing.
Q-12 Difference b/w verification and validation.
Q-13 Difference b/w alpha and Beta testing.
Q-14 What is Unit testing ~~and~~ ?
Q-15 Define Software project Management.
Q-16 Describe COCOMO model.
Q-17 Difference b/w debugging and testing.
Q-18 Difference b/w LOC and FP.
Q-19 What are function point in software mangement.
Q-20 What is Software Case tools and its type.
Q-21 Why is software Maintenance is costly.
Q-22 Discuss the component of software maintenance process.
Q-21 What is a central repository?
Q-22 Difference between CMM & ISO.
Q-23 How IEEE defines in Software Engineering.
Q-24 Write a brief on Gantt chart.
Q-25 Difference between scheduling and staffing.
~~Q-26 Agile, Scrum defind~~
Q-26 Describe the Agile model and Scrum.

Page No.
Date / 20

Q 1 System and Software development stages ?

Ans System development stages :-

1 System planning => In this stage, the overall objectives and scopes of the software system are defined.

2 System analysis => This stage involves studying the existing system (if applicable) and analyzing the user requirement in details. It include activity such as Gathering and documenting functional & Non-functional requirement, identify System component.

3 System Design => In this stage, the software system's architecture is designed. It include defining the system's structure, specifying system component & relationship, and interfaces.

4 System Implementation => This stage focuses on the actual development of the system. It involves writing code, integrating component, and implementing the designed architecture.

Software development stages.

1 Requirement Analysis => This stage involves understanding and documenting the detailed functional and non functional requirement of software.

2 Software Design => It deals with designing the software structure and components based on the requirement.

3 Coading ⇒ This stage involves translating the software design into actual code using a programming language.

4 Testing ⇒ In this stage, the software is tested to ~~identify and~~ identify and fix defects, validate its functionality, and ensure it meets the specified requirement.

5 Deployment ⇒ This stage involves preparing the software for release and installation

6 Maintenance ⇒ Once the software is deployed, it enters the maintenance stage. It involves monitoring & supporting the software in the production environment, addressing user feedback, fixing bugs, and making enhancements or updates as needed.

Q-2 Explain Waterfall and Spiral Model.

Ans Waterfall Model :- The waterfall model is a linear and sequential approach to software development. It consist of a series of distinct phases that are completed one after another, with each phase acting as a prerequisite for next phase. The typical phases in the waterfall model are:

- Requirement Gathering ⇒ The requirement for the software are collected from the stakeholds.
- System Design ⇒ The software systems architecture and designs are planned and documented.

- Implementation ⇒ The software is developed based on the design specifications.
- Testing ⇒ The software is tested to ensure that it meets the specified requirements.
- Deployment ⇒ The software is deployed and made available to the end user.
- Maintain Maintenance ⇒ Ongoing support and maintenance are provided for the stud software.

Spiral Model :- It allows for flexibility and incorporates feedback and changes into the development process. It is particularly useful when dealing with large and complex projects, where requirements may evolve or be uncertain. By iterating through the phases, the project team can address risks and refine the software incrementally, resulting in an improved final product.
It combines elements of both waterfall & prototype models and emphasizes risk analysis and mitigation through development process.

- Planning : Goals, alternatives, and constraints are identified, along with a risk analysis.
- Risk analysis: Risk associated with the project are assessed, and mitigation strategies are planned.
- Engineering : The software is developed, tested, and evaluated in a series of ~~int~~ iterations.
- Evaluation : The current iteration is reviewed, lessons are learned, and plans for the next iteration are refined.

Page No.
Date / 20

Q.3 Which is more important process or product?

Ans Both the process and the product are important in software development. The process ensure efficient and effective development, while the product represents the end result that satisfies user needs. A well-defined process leads to a high-quality product that meets user exceptations. Balancing both aspects is crucial for success.

Process :-

- The software development process refers to the set of activities, methodologies, and technique followed to create a software product.
- It includes tasks like requirement analysis, design, coading, testing, and deployment, as well as project management, communication, collaboration, and ~~Quality~~ quality assurance practices.
- A well defined & well executed process brings benefit such as consistency, quality, efficient time and resource management, effective collaboration, and risk management.

Product :-

- The software product is the tangible outcome of the development process, meeting user needs & exceptation.
- Its quality, functionally, usability, reliability, and performance are key factors for its success and user satisfaction.

- The product is the tangible outcome
- The product is heavily influenced by the development process, as a well-executed process enhances the likehood of producing a high-quality product, while a flawed or poorly managed process can negatively impact the product.

Q.3 Difference between ER and DFD diagram?

Ans

1- Purpose

- ER Diagrams => It primarily focus on representing the relationships and structure of data entities within a system. They are used to model the data requirements and conceptual schema of a system, emphasizing entities, attributes, and their relationship.
- DFD Diagrams => It emphasize the flow of data within a system. They illustrate the processes, data stores, external entities, and data flows, providing an overview of how data moves through the system.

2. Scope

- ER => It used during the early stages of system analysis and design to capture the data requirement and relationship b/w entities. They are especially useful for database design and modeling.
- DFD => It used to model the entire systems data flow, ~~, data stores, and external~~ including both data transformation processes and data stores. They provide a broader view of the system functionality, how data moves b/w diff components

3 Representation

ER ⇒ It use entities, attributes, and relationships to represent the structure and association of data elements. Entities are represented as rectangle attributes as oval, and relationships as line connecting entities.

DFD ⇒ It use processes, data flows, data stores and external entities to represent the flow of data. Processes are represented as rectangles with a label, data flow as arrows, data stores as two parallel lines, and external entities as rectangle with rounded corners.

4 Level of Abstraction

ER ⇒ It provide a high-level conceptual view of the systems data entities and their relationships. They are more abstract and focus on the logical data modeling.

DFD ⇒ It provide a functional view of the system, illustrating how data flows and processes are interconnected. They are relatively more detailed and represent the systems processes and data flow paths.

Page No.
Date / 20

Q-1 Difference between functional and Non functional Requirement

Ans • functional req

- It define the specific functions, features, and behaviour that the software system should exhibit.
- They describe what the system should do and how it should respond to various inputs and user interactions.
- They are typically expressed in the form of use cases, user stories, or specific system functionalities.
- Examples of functional requirements include user authentication, data validation, report generation, and search functionalities.
- They are usually specifics, measurable, and can be validated through testing.

• Non-functional req

- It specify the qualities, constraints, and characteristics that the software system should possess.
- They focus on how the system should perform rather that what it should do.
- Non-functional req address aspects such as performance, security, usability, reliability, scalability, maintainability, and availability.
- They define the overall attributes and qualities that the system should exhibit to meet user expectations.
- Example of non-functional requirements include response time, system availability, security measures, user interface aesthetics, and error handling.

Date / 20

Q.5 Difference b/w structural and non-structural Testing:

- Structural Testing

→ It is also known as white-box testing or code-based technique, focuses on examining the internal ~~structure~~ structure of the software system.
→ It involves testing the individual components, such as functions, methods and classes to, ensure that they behave as expected.
→ Structural testing aims to cover different path, conditions, and statements within the code to identify potential defects and ensure code coverage.
→ Techniques used in structural testing include statement coverage, branch coverage, path coverage, and condition coverage.
→ The goal of structural testing is to validate the design and implementation of the software system by examing its internal code structure.

- Non-Structural Testing

→ It is not a widely ~~recognize~~ recognized term in software testing. However, based on the context we can assume it refers to testing aspects that are not directly related to the internal structure of the software system.
→ It may encompass various types of testing that focus on the system's external behavior, functionality and user experience.
→ Example of non-structural testing types include functional testing, security testing, compatibility testing, and user acceptance testing.

- Non-Structural testing, security typically involves testing the software system as a whole, without delving into the internal code structure or implementation details.

Q6 Describe Prototyping techniques in Software Engineering?

Ans There are several prototyping techniques commonly used.

1. Throwaway prototyping ⇒ Also known as rapid or low-fidelity prototyping, this technique involves quickly building a basic prototype with limited functionality. The primary goal is to gather feedback and validate design choice.

2. Evolutionary prototyping ⇒ This technique focuses on gradually refining and enhancing the initial prototype based on user feedback and evolving requirement. The prototype evolves through multiple iterations, with each iteration building upon the previous one.

3. Incremental prototyping ⇒ In incremental prototyping, the process is divided into small increments or modules. Each module represent a subset of the system's functionality and it is developed and tested independently. This technique allow for the flexibility and allows for the integration of new features incrementally.

Q-1 Describe software testing and also the object of testing.

Ans Software testing is a critical process in software development that involves evaluating a software system or application to identify defects, error, or any deviation from expected ~~behaviour~~ behavior. The primary objective of testing is to ensure that the software meets the specified requirements, functions correctly, and delivers the intended value to its user.

Here are key objectives of software testing.

1. Identifying defect.
2. Verifying functionality.
3. Ensuring reliability and stability
4. Enhancing quality.
5. Validating user expectation
6. Mitigating risks.
7. Compliance and standards.

The object of testing can be broadly categorized into the following.

- Functional testing => It ensure that the software performs its intended functions and operations correctly.

- Non-functional testing => It ensure performance, scalability, reliability, security, usability, and accessibility. It ensure that the software meets the desire quality attributes and user expectations.

- Integration testing => It aims to verify the correct interaction and communication between different components or modules of the software.
- Performance testing => It measure response time, throughput, scalability and stability to ensure that the software performs ~~optimal~~ optimally.
- Security testing => It aims to protect against unauthorized access, data breaches, and other security risks.
- Usability testing => It assure the softwares user ~~friendly~~ friendliness, intuitiveness, and ease of use.

Q-8 Difference between Black box and White box testing.

Ans. They are two distinct approaches to software testing that differ in their focus, prespective, and methadologies. Here are key difference between Black box and white box testing.

Black box Testing

- focus: Black box testing emphasizes the external behaviour of the software without considering its internal structure or implementation details.
- ~~Test~~ Pospective: Testers approach the software as a "black box" and have no knowledge of its internal working.

Page No.
Date / 20

- Knowledge: Testers do not require knowledge of the internal code or system architecture.
- Testing Approach: It is based on functional requirement, specifications, and user expectations.
- Test Design: It derived from the systems functional specification, user stories, use cases, or other external documents.
- Testing type: It encompasses techniques such as functional testing, integration testing, system testing, acceptance testing and usability testing.
- Advantages: It focuses on end user experience, ensure that the software meets user requirements, and dose not rely on internal implementation knowledge.
- Disadvantages: It may not fully explore internal code paths, making it possible to miss certain defects or error that only revealed through white box testing.

White box testing

- focus: It is concerned with the internal structure, logic, and implementation details of the software.
- perspective: ~~Testing~~ Testers have access to the internal code and system architecture.
- Knowledge: Tester require knowledge of the internal code, programming languages, algorithms, and system design.

- Testing approach: It is based on code coverage criteria to exercise various path, conditions, and logic within the code.
- Testing types: white box testing encompasses techniques such as unit testing, integration testing, and code coverage analysis.
- Advantages: It provides insights into the internal code quality ensure through coverage of code path, and can uncover defects specific to implementation.
- Disadvantages: It requires deep technical knowledge, can be time-consuming, and may not directly address end-user requirement or usability issues.

Q-9 Define Cyclomatic Complexity.

Ans - Cyclomatic Complexity is a software metric that quantifies the complexity of d'programs control flow. It measures the number of decision points and independent path within in the programs source code. By analyzing the control flow graph, which represents the program's structure, it counts conditional statements, loops, and other branching constructs. Higher complexity values indicate greater potential for errors and reduce code readability. Cyclomatic complexity helps assess maintainability and testability, guides code refactoring, and determines appropriate

test coverage. It serves as a useful tool for understanding and managing the complexity of software systems, ensuring code quality and facilitating effective testing.

Q-10 Write about Boundary value testing & Independent testing.

Ans Boundary Value Testing => Boundary value testing is a software testing technique that focuses on testing input values at the boundaries of valid range. It recognizes that errors are more likely to occur at the edges of input ranges rather that than within the middle range. Test cases are designed to test the minimum, maximum, and boundary values just before and after a valid range. By testing boundary value conditions, testers aim to uncover defects related to input validation, calculations, and system behavior. This technique helps improve software robustness by ensuring that the software handles edge case correctly and avoids issues such as off-by-one errors or incorrect handling of limits.

Independent testing =>