1. Define software.
   Software is a collection of computer programs and associated documentation. Software products may be developed for a particular customer or may be developed for a general market.

2. What are the attributes of good software?
   Good software should deliver the required functionality and performance to the user and should be maintainable, dependable, and usable.

3. Define software engineering.
   Software engineering is an engineering discipline that is concerned with all aspects of software production.

4. What are the major activities involved in software development.
   Software specification, software development, software validation, and software evolution.

5. What are the key challenges facing software engineering?
   Coping with increasing diversity, demands for reduced delivery times, and developing trustworthy software.

6. What are the costs of software engineering?
   Roughly 60% of software costs are development costs; 40% are testing costs. For custom software, evolution costs often exceed development costs.
7. What are the best software engineering techniques and methods?
   While all software projects have to be professionally managed and developed, different techniques are appropriate for different types of system. For example, games should always be developed using a series of prototypes whereas safety critical control systems require a complete and analyzable specification to be developed. You can't, therefore, say that one method is better than another.
8. What differences has the Web made to software engineering?
   The Web has led to the availability of software services and the possibility of developing highly distributed service-based systems. Web-based systems development has led to important advances in programming languages and software reuse.

9. What is the difference between software engineering and system engineering?

10. What are the other responsibilities of software engineer besides technical aspects?
11. Define requirement engineering.

    Requirements engineering is the process of developing a software specification. Specifications are intended to communicate the system needs of the customer to the system developers.
12. What is software validation?

    Software validation is the process of checking that the system conforms to its specification and that it meets the real needs of the users of the system.
13. What is software evolution?

    Software evolution takes place when you change existing software systems to meet new requirements. Changes are continuous and the software must evolve to remain useful
14. Define RUP.

    The Rational Unified Process is a modern generic process model that is organized into phases (inception, elaboration, construction, and transition) but separates activities (requirements, analysis, and design, etc.) from these phases.
15. State the particular strength of extreme programming.

    A particular strength of extreme programming is the development of automated tests before a program feature is created. All tests must successfully execute when an increment is integrated into a system.
16. Define scrum.

    The Scrum method is an agile method that provides a project management framework. It is centered around a set of sprints, which are fixed time periods when a system increment is developed. Planning is based on prioritizing a backlog of work and selecting the highest priority tasks for a sprint.
17. Scaling agile methods for large systems is difficult. Why?

    Scaling agile methods for large systems is difficult. Large systems need up-front design and some documentation. Continuous integration is practically impossible when there are several separate development teams working on a project.
18. Define Extreme Programming.

    Extreme programming is a well-known agile method that integrates a range of good programming practices such as frequent releases of the software, continuous software improvement, and customer participation in the development team.
19. Define requirement elicitation and analysis.

Requirements elicitation and analysis is an iterative process that can be represented as a spiral of activities—requirements discovery, requirements classification and organization, requirements negotiation, and requirements documentation.

20. What is requirement validation?

    Requirements validation is the process of checking the requirements for validity, consistency, completeness, realism, and verifiability.

21. Define SRS document.

    The software requirements document is an agreed statement of the system requirements. It should be organized so that both system customers and software developers can use it.

22. Define non-functional requirements.

    Non-functional requirements often constrain the system being developed and the development process being used. These might be product requirements, organizational requirements, or external requirements. They often relate to the emergent properties of the system and therefore apply to the system as a whole.

23. Define Requirement management.

24. What is model driven engineering.

    Model-driven engineering is an approach to software development in which a system is represented as a set of models that can be automatically transformed to executable code.

25. What are the commonly used architectural patterns?

    Commonly used architectural patterns include Model-View-Controller, Layered Architecture, Repository, Client–server, and Pipe and Filter.

26. Define Transaction Processing System.

    Transaction processing systems are interactive systems that allow information in a database to be remotely accessed and modified by a number of users. Information systems and resource management systems are examples of transaction processing systems.

27. What are the activities involved in object oriented design?

    The process of object-oriented design includes activities to design the system architecture, identify objects in the system, describe the design using different object models, and document the component interfaces

28. List the static models produced during an object oriented design process?

    class models, generalization models, association models

29. List the dynamic models produced during an object oriented design process?

sequence models, state machine model

30. What is configuration management?

    Configuration management is the process of managing changes to an evolving software system. It is essential when a team of people are cooperating to develop software.

31. Define Open Source development.

    Open source development involves making the source code of a system publicly available. This means that many people can propose changes and improvements to the software.

32. Define test-first development approach.

    Test-first development is an approach to development where tests are written before the code to be tested. Small code changes are made and the code is refactored until all tests execute successfully.

33. Define white box testing.
34. Define black box testing.
35. Define unit testing.
36. Define acceptance testing.

    Acceptance testing is a user testing process where the aim is to decide if the software is good enough to be deployed and used in its operational environment.

37. State the concept of Lehman's laws.

    Lehman's laws, such as the notion that change is continuous, describe a number of insights derived from long-term studies of system evolution.

38. What are the types of software maintenance?

    There are three types of software maintenance, namely bug fixing, modifying the software to work in a new environment, and implementing new or changed requirements.

39. Define System Re-engineering.

    Software reengineering is concerned with restructuring and re-documenting software to make it easier to understand and change.

40. What is sociotechnical system?

    Sociotechnical systems include computer hardware, software, and people, and are situated within an organization. They are designed to support organizational or business goals and objectives.

41. What are the emergent properties of a system?

    The emergent properties of a system are characteristics of the system as a whole rather than of its component parts. They include properties such as performance, reliability, usability, safety, and security. The success or failure of a system is often dependent on these emergent properties.

42. Define system procurement process.

System procurement covers all of the activities involved in deciding what system to buy and who should supply that system. High-level requirements are developed as part of the procurement process.

43. What are the consequences of critical computer system failure?

    Failure of critical computer systems can lead to large economic losses, serious information loss, physical damage, or threats to human life.

44. Define the dependability of a computer system.

    The dependability of a computer system is a system property that reflects the user's degree of trust in the system. The most important dimensions of dependability are availability, reliability, safety, and security.

45. Differentiate system availability and reliability.

    The availability of a system is the probability that the system will be able to deliver services to its users when requested to do so. Reliability is the probability that system services will be delivered as specified.

46. Differentiate safety and security.

    The safety of a system is a system attribute that reflects the system's ability to operate, normally or abnormally, without injury to people or damage to the environment.

    Security reflects the ability of a system to protect itself against external attacks. Security failures may lead to loss of availability, damage to the system or its data, or the leakage of information to unauthorized people.

47. Define Risk analysis.

    Risk analysis is an important activity in the specification of security and dependability requirements. It involves identifying risks that can result in accidents or incidents. System requirements are then generated to ensure that these risks do not occur and, if they do, that they do not lead to an incident or accident.

48. What is the use of a hazard-driven approach for a system?

    A hazard-driven approach may be used to understand the safety requirements for a system. You identify potential hazards and decompose these (using methods such as fault tree analysis) to discover their root causes. You then specify requirements to avoid or recover from these problems.

49. Define POFOD.

    Reliability requirements can be defined quantitatively in the system requirements specification. Reliability metrics include probability of failure on demand (POFOD)

50. Define ROCOF.

Reliability requirements can be defined quantitatively in the system requirements specification. Reliability metrics include probability of Rate of Occurrence of Failure (ROCOF).

51. Security requirements are more difficult to identify than safety requirements. Give Reason.

    Security requirements are more difficult to identify than safety requirements because a system attacker can use knowledge of system vulnerabilities to plan a system attack, and can learn about vulnerabilities from unsuccessful attacks.