1. What is programme:
   - programme is a set of instructions.
2. What is programming:
   - it is a set of words and symbols use to write programs
3. What are the key steps involved in the programming process?
4. Types of Programming Languages:
   1) procedural programming language (c language)
   2) object-oriented language
   3) logical language
   4) functional language (python)
5. What are the main differences between high-level and low-level programming languages?
   - High-level languages: are user-friendly, portable, and efficient for general application development.

   - Low-level languages: provide direct hardware control, making them ideal for system programming and performance-critical applications.

6. World Wide Web & How Internet Works

-The World Wide Web is a system of interlinked hypertext documents and other resources that are accessed via the Internet.

-Users Can access the content of these sites from any part of the world Over the internet using Such as a Computers laptops Cell Phones etc.

7. Describe the roles of the client and server in web communication.

-Client sometimes on.

-Initiate a request to the server when interested.

-Ex. web browser on your laptop or phone,

-Doesn't communication directly with other clients. needs to know the server address.

8. Network Layers on Client and Server

9. Explain the function of the TCP/IP model and its layers.

-The TCP/IP model is a framework that allows communication over interconnected networks, like the Internet. It is based on a suite of communication protocols and stands for Transmission Control Protocol/Internet Protocol.

10. Explain Client Server Communication.

-Client sometimes on.

-Initiate a request to the server when interested.

-Ex. web browser on your laptop or phone,

-Doesn't communication directly with other clients. needs to know the server address.

11. Types of Internet Connections.

-Digital Subscriber Lines,

-Cable Internet,

-Fibber Optic,

-Satellite Internet,

-Wireless,

-Board Band Over Power Line

12. How does broadband differ from fibre-optic internet?

-Broadband is a general term covering different types of internet connections (DSL, cable, satellite, wireless).

-fibber Optic Internet is the fastest and most reliable type of broadband, but it has limited availability and higher costs.

13. Protocols.

-Set of rules which are used in digital communication to connect network devices and exchange information between them.

14. What are the differences between HTTP and HTTPS protocols?

-HTTP (Hypertext Transfer Protocol) and HTTPS (Hypertext Transfer Protocol Secure) are protocols used for transmitting data over the web.

HTTP **(Hypertext Transfer Protocol):**

- Data is transmitted in plain text.
- Uses port 80.
- No encryption, less secure.
- Does not require a security certificate.
- Suitable for non-sensitive information.

HTTPS **(Hypertext Transfer Protocol Secure):**

- Data is encrypted.
- Uses port 443.
- Provides secure communication through SSL/TLS encryption.
- Requires an SSL/TLS certificate.
- Suitable for sensitive information and secure transactions

15.Application Security

-Application Security involves implementing security measures at the application level to protect against threats and vulnerabilities during development, deployment, and usage.

16. What is the role of encryption in securing applications?

Encryption plays a crucial role in securing applications by protecting the confidentiality, integrity, and authenticity of data.

encryption is a fundamental security mechanism that ensures sensitive data remains private, intact, and accessible only to authorized parties. It is an essential tool for securing applications and protecting user data in an increasingly digital world

17. Software Applications and Its Types.

Software applications, often referred to as "apps," are programs or software designed to perform specific tasks for users. These tasks can range from productivity and communication to entertainment and beyond.

-Application Software,

-System Software,

-Driver Software,

-Middleware Software,

-Programming Software

18. What is the difference between system software and application software.

System software is essential for the overall functioning of a computer system and provides a platform for application software, which helps users accomplish specific tasks.

19. Software Architectures.

-Software architecture refers to the high-level structure of a software system, encompassing the organization of its components, their relationships, and the principles guiding its design and evolution.

-It serves as a blueprint for both the system and the project developing it, ensuring that the software meets both functional and non-functional requirements.

20. What is the significance of modularity in software architecture?

Modularity is a fundamental design principle in software architecture that emphasizes dividing a system into smaller, self-contained units or modules.

Each module encapsulates a specific functionality, making the overall system easier to develop, understand, and maintain.

21. Layers in Software Architecture.

-Presentation layer

-Application Layer,

-Business Layer,

-Persistence layer,

-Database Layer

22.Why are layers important in software architecture?

-Layering is a fundamental concept in software architecture, where the software is organized into a series of layers, each with specific responsibilities

23. Software Environments.

-A software environment refers to the collection of software tools, libraries, frameworks, and configurations that support the development, testing, deployment, and operation of software applications.

-It provides the necessary infrastructure and setup for software projects to run efficiently.

24. Explain the importance of a development environment in software production.

25. Source Code.

-source code is the source of a computer program.

-Source code is the human-readable set of instructions and statements written by a programmer in a programming language to create a software application.

-It is the fundamental component of any software and serves as the blueprint for the software's functionality

26. What is the difference between source code and machine code.

Aspect	**Source Code**	**Machine Code**
Readability	Human-readable	Machine-readable (binary)
Editability	Easily editable by programmers	Not editable by humans
Language	High-level programming languages	Low-level binary instructions
Execution	Needs to be compiled or interpreted	Directly executed by the CPU
Portability	Portable across different platforms	Specific to the hardware architecture

27.GitHub and Introductions.

-GitHub is an essential tool for modern software development, offering powerful features for version control, collaboration, and automation.

-It's widely used by developers and organizations to manage and share code effectively.

28. Why is version control important in software development

-Version control is a critical aspect of software development, and it offers numerous benefits that enhance the efficiency, collaboration, and reliability of software projects.

-Here are some key reasons why version control is important.

29. What are the benefits of using GitHub for students.

-GitHub offers numerous benefits for students, helping them enhance their coding skills, collaborate with others, and build a professional portfolio.

-Here are some key benefits of using GitHub for students

30. Types of Software.

-Application Software,

-System Software,

-Driver Software,

-Middleware Software,

-Programming Software

31. What are the differences between open-source and proprietary software?

Aspect	Open-Source Software	Proprietary Software
Source Code Accessibility	Publicly available	Not available
Licensing	Open-source licenses	Restrictive licenses
Cost	Usually free	Often requires a purchase or subscription
Development	Community-driven	Controlled by a single organization
Flexibility	Highly customizable	Limited customization
Support	Community and optional professional	Professional support by vendor
Security	Peer-reviewed, quick patches	Vendor-managed, potential delays

32. How does GIT improve collaboration in a software development team.

1. Distributed Version Control

- **Function:** Git is a distributed version control system, meaning each developer has a full copy of the repository, including its history, on their local machine.
- **Benefit:** This allows developers to work independently and offline, committing changes locally and synchronizing with the team when convenient.

2. Branching and Merging

- **Function:** Git supports lightweight branching and merging, enabling developers to create branches for new features, bug fixes, or experiments.
- **Benefit:** This allows parallel development efforts without disrupting the main codebase. Merging integrates changes back into the main branch seamlessly.

3. Pull Requests

- **Function:** Pull requests (also known as merge requests) are a way to propose changes to the codebase and request reviews from team members.
- **Benefit:** This facilitates code review, feedback, and collaboration, ensuring code quality and collective ownership of the codebase.

4. Commit History

- **Function:** Git maintains a detailed history of all changes made to the repository, including who made the changes, when, and why (with commit messages).
- **Benefit:** This provides transparency and accountability, making it easier to track progress, understand the rationale behind changes, and identify potential issues.

5. Conflict Resolution

- **Function:** Git helps manage and resolve conflicts that arise when multiple developers make changes to the same part of the codebase.
- **Benefit:** Conflict resolution tools and workflows ensure that changes are integrated smoothly, minimizing disruptions and maintaining a cohesive codebase.

6. Collaboration and Contribution

- **Function:** Git enables collaboration through shared repositories, forks, and cloning. Developers can contribute to projects, both internal and open-source, by submitting pull requests.
- **Benefit:** This fosters a collaborative environment, where team members can share knowledge, contribute to each other’s work, and collectively improve the project.

7. Continuous Integration/Continuous Deployment (CI/CD)

- **Function:** Git integrates with CI/CD tools to automate the process of building, testing, and deploying code changes.

33. Application Software:

-Application software is a type of computer program designed to help users perform specific tasks or activities.

-Unlike system software (which manages hardware and basic system functions), application software is user-focused and enables productivity, creativity, or entertainment.

34. what is the role of application software in businesses?

-Application software is essential for modern businesses, enabling efficiency, automation, data management, and improved decision-making.

-It helps organizations streamline operations, enhance productivity, and maintain a competitive edge.

35. Software Development Process.

-The **Software Development Process** (also known as the Software Development Life Cycle, or SDLC) is a structured approach to designing, developing, testing, and deploying software applications.

-It ensures that software meets user requirements, is efficient, and is maintainable over time.

-Planning,

-Analysis,

-Designing,

-Implementation,

-Testing,

-Maintenance

36. What are the main stages of the software development process?

-The **Software Development Process**, also known as the **Software Development Life Cycle (SDLC)**, consists of several key stages to ensure the successful creation, deployment, and maintenance of software applications.

-Planning, Analysis, Designing, Implementation, Testing, Maintenance

37. **Software Requirement.**

-Software Requirement refers to the specifications of what a software system should do.

-It defines the functionalities, constraints, and expected behaviour of the system.

-Well-defined requirements help ensure that the final product meets user expectations and business goals.

38. Why is the requirement analysis phase critical in software development?

-The **Requirement Analysis** phase is one of the most crucial stages of the **Software Development Life Cycle (SDLC)**.

-It sets the foundation for the entire project by identifying and defining what the software should do.

-A well-executed requirement analysis helps prevent costly errors, ensures alignment with business goals, and improves software quality.

39. Software Analysis

-Software Analysis is the process of examining and understanding the requirements, structure, and functionality of a software system.

-It plays a crucial role in the **Software Development Life Cycle (SDLC)** by ensuring that software solutions meet business needs, are efficient, and are maintainable.

40. What is the role of software analysis in the development process?

-Software analysis is a **critical phase** in the **Software Development Life Cycle (SDLC)** that ensures software systems meet user needs, are efficient, secure, and maintainable.

- It helps developers, testers, and stakeholders **understand, evaluate, and improve** the software before and after implementation.

41. System Design

-System Design is the process of defining the architecture, components, modules, interfaces, and data of a system to satisfy specified requirements.

-It is a crucial phase in the **Software Development Life Cycle (SDLC)** that ensures the system is scalable, efficient, and maintainable.

42. What are the key elements of system design?

-System design involves planning the structure and functionality of a system to ensure it meets business requirements efficiently.

- It includes various components that define **architecture, data flow, security, performance, and usability**.

43. Why is software testing important?

-Software testing is a **critical phase** in the **Software Development Life Cycle (SDLC)** that ensures software is **functional, reliable, secure, and high-performing**.

-It helps detect defects early, reducing costs and improving user satisfaction.

44. Development

-Software development is the **process of designing, coding, testing, and maintaining** applications or systems to meet user and business needs.

-It follows a structured approach, ensuring **efficiency, security, and scalability**.

45. Web Application

-A **web application** is a software program that runs on a web server and is accessed through a **web browser**.

-Unlike traditional desktop applications, web applications do not require installation and can be used on multiple devices.

46. Designing

-Designing is a **critical phase** in software development that defines the **structure, functionality, and user experience** of an application.

-A well-planned design ensures the software is **scalable, efficient, secure, and user-friendly**.

47. What role does UI/UX design play in application development?

-UI/UX design plays a **crucial role** in application development by ensuring a **visually appealing, intuitive, and user-friendly experience**.

-A well-designed UI/UX improves **engagement, usability, and overall user satisfaction**, leading to higher adoption rates and business success.

48. Mobile Application.

-A **mobile application (mobile app)** is software designed to run on **smartphones, tablets, and other mobile devices**.

-Mobile apps can be built for various platforms like **Android, iOS, or cross-platform frameworks**

49. What is the significance of DFDs in system analysis?

-A **Data Flow Diagram (DFD)** is a crucial tool in **system analysis** that helps visualize the flow of data within a system.

-It plays a key role in **understanding, designing, and optimizing** business processes and software systems.

50 What are the pros and cons of desktop applications compared to web applications?

Pros of Desktop Applications:

-**Better Performance** – Uses device resources directly, leading to **faster processing and smoother execution**.
- **Offline Access** – Can function **without an internet connection**, unlike web apps.
-**Enhanced Security** – Data is stored locally, reducing exposure to **cyber threats and hacking**.
- **Full Hardware Access** – Can interact with **printers, USB devices, GPUs**, and other peripherals more efficiently.
- **Customization & Stability** – Offers more control over **user settings, UI, and system integration**.

51. how do flowcharts help in programming and system design?

-A **flowchart** is a **visual representation** of a process, algorithm, or system workflow using **symbols and arrows**.

-It plays a critical role in both **programming and system design** by improving clarity, efficiency, and problem-solving.