**Q-1:-Explain in your own words what a program is and how it functions.**

* **A program is a list of instructions that tells a computer what to do. It guides the computer step by step to complete a task or solve a problem. Without a program, the computer cannot work or understand what to do.**
* **A program works by giving computer instructions one by one. The computer reads these instructions and does exactly what they say, like following steps in a recipe. This way, the computer can perform tasks automatically without needing a person to tell it what to do at every moment.**

**Q-2:-What are the key steps involved in the programming process?**

* **Key Steps in the Programming Process:**

1. **Understand the Problem: Know what you want the program to do.**
2. **Plan the Solution: Think about how to solve the problem step by step.**
3. **Write the Code: Use a programming language to write instructions for the computer.**
4. **test the Program: Run the program to check if it works correctly.**
5. **Fix Errors: Find and correct mistakes in the program (called bugs).**
6. **Improve the Program: Make the program better or add new features.**

**Q-3:-What are the main differences between high-level and low-level programming languages?**

* **High-Level Languages:**  
  **Easy to read and write because they use words like English.  
  Examples: Python, Java, C++.  
  They work on many types of computers.**
* **Low-Level Languages:**  
  **Closer to the computer’s machine code, using numbers and symbols.  
  Examples: Assembly language, Machine code.  
  They are harder to read but run very fast**

**Q-4:-** **Describe the roles of the client and server in web communication**

* **Client:**  
  **The client is the device or program (like a web browser) that requests information or services from the server. It sends a request to see a webpage, download a file, or get data.**
* **Server:**  
  **The server is a powerful computer that listens to requests from clients. It processes these requests and sends back the requested information, like a webpage or file.**

**Q-5:- Explain the function of the TCP/IP model and its layers.**

**Function: TCP/IP model helps computers communicate over the internet. It makes sure data goes from one device to another correctly.**

**Layers:**

1. **Application Layer**

* **Help programs like web browser, email, and file transfer to use the network.**
* **Examples: HTTP, FTP, SMTP, DNS**

1. **Transport Layer**

* **Breaks data into smaller pieces and ensures it reaches safely.**
* **Protocols: TCP (reliable), UDP (fast, no guarantee)**

1. **Internet Layer**

* **Gives IP addresses and finds the path for data to travel across networks.**
* **Protocols: IP, ICMP, ARP**

1. **Network Access Layer**

* **Sends data through cables, Wi-Fi, or other hardware.**
* **Handles physical addresses (MAC) and frames.**
* **Examples: Ethernet, Wi-Fi**

**Q-6:- Explain Client Server Communication**

**Client-Server Communication**

**Definition: It is the way computers (clients) talk to a central computer (server) to request and receive information.**

**How it works:**

1. **Client sends a request – The client (like a web browser) asks the server for something, e.g., a web page or file.**
2. **Server processes the request – The server receives the request, finds the needed data, and prepares a response.**
3. **Server sends a response – The server sends back the requested information to the client.**
4. **Client receives data – The client shows or uses the data (e.g., displaying a web page).**

**Key Points:**

* **The client initiates communication.**
* **The server responds to client requests.**
* **Multiple clients can communicate with the same server at the same time.**
* **Communication happens over TCP/IP using protocols like HTTP, FTP, etc**

**Q-7:-How does broadband differ from fiber-optic internet?**

1. **Definition:**

* **Broadband: A general term for high-speed internet that uses cables, DSL, or satellite.**
* **Fiber-Optic Internet: A type of broadband that uses thin glass fibers to transmit data as light signals.**

1. **Speed:**

* **Broadband: Usually slower (10–100 Mbps depending on type).**
* **Fiber-Optic: Much faster (up to 1 Gbps or more).**

1. **Reliability:**

* **Broadband: Can be affected by distance from the provider and interference.**
* **Fiber-Optic: Less affected by distance or interference, very stable.**

1. **Latency:**

* **Broadband: Higher latency, not ideal for online gaming or video calls.**
* **Fiber-Optic: Very low latency, excellent for real-time applications.**

1. **Cost:**

* **Broadband: Usually cheaper.**
* **Fiber-Optic: Slightly more expensive but worth it for speed and reliability.**

**Q-8:-What are the differences between HTTP and HTTPS protocols?**

**Differences between HTTP and HTTPS:**

1. **Full Form:**

* **HTTP: Hypertext Transfer Protocol**
* **HTTPS: Hypertext Transfer Protocol Secure**

1. **Security:**

* **HTTP: Not secure, data is sent as plain text.**
* **HTTPS: Secure, data is encrypted using SSL/TLS.**

1. **Data Protection:**

* **HTTP: Vulnerable to hackers, data can be intercepted.**
* **HTTPS: Protects data from eavesdropping and tampering.**

1. **Port Number:**

* **HTTP: Uses port 80**
* **HTTPS: Uses port 443**

1. **Performance:**

* **HTTP: Slightly faster because of no encryption.**
* **HTTPS: Slightly slower due to encryption overhead.**

1. **SEO & Trust:**

* **HTTP: Websites may be marked as “Not Secure.”**
* **HTTPS: Preferred by search engines and trusted by users.**

**Q-9:-What is the role of encryption in securing applications?**

1. **Definition:  
   Encryption is the process of converting data into a coded form so that only authorized users can read it.**
2. **Protects Data Privacy:**

* **Sensitive information like passwords, credit card numbers, and personal data is hidden from hackers.**

1. **Ensures Data Integrity:**

* **Prevents data from being altered or tampered with during transmission.**

1. **Authentication:**

* **Confirms that data comes from a trusted source.**

1. **Secure Communication:**

* **Enables safe communication over the internet (e.g., HTTPS, VPNs, messaging apps).**

1. **Compliance & Trust:**

* **Helps applications comply with security regulations and gain users’ trust.**

**Q-10:- What is the difference between system software and application software?**

| **Feature** | **System Software** | **Application Software** |
| --- | --- | --- |
| **Definition** | **Software that manages and controls computer hardware and provides a platform for running application software.** | **Software designed to perform specific tasks or solve problems for the user.** |
| **Purpose** | **To run the computer and provide an environment for applications to work.** | **To help users perform tasks like writing, browsing, gaming, or accounting.** |
| **Examples** | **Operating systems (Windows, Linux, macOS), device drivers, utility programs.** | **Microsoft Word, Photoshop, Chrome, Excel, games.** |
| **Dependency** | **Can run independently (though it may need hardware).** | **It depends on system software to run.** |
| **Interaction with Hardware** | **Directly interacts with hardware components.** | **Usually interact with hardware through system software.** |
| **Users** | **Mostly used indirectly by end-users.** | **Directly used by end-users.** |

**Q-11:- What is the significance of modularity in software architecture?**

**Modularity means dividing a software program into smaller, independent parts called modules, each handling a specific task.**

**significance of modularity**

1. **Easier Maintenance: If something goes wrong, you can fix or update a specific module without affecting the whole system.**
2. **Improved Reusability: Modules can often be reused in other projects, saving time and effort.**
3. **Simplified Testing: Testing smaller, isolated pieces of code is faster and more reliable.**
4. **Better Collaboration: Teams can work on different modules at the same time without stepping on each other’s toes.**
5. **Scalability and Flexibility: It’s easier to scale or modify parts of a modular system as requirements change.**

**Q-12:- Why are layers important in software architecture?**

**Layers are important in software architecture because they organize a system into manageable parts, making it easier to develop, maintain, and scale.**

**Separation of Concerns – Each layer has a specific responsibility (e.g., presentation, business logic, data access). This reduces complexity and makes the system easier to understand.**

1. **Reusability – Layers like data access or utility services can be reused across different parts of the application or even in other projects.**
2. **Maintainability – When a change is needed, it often affects only one layer, minimizing the risk of breaking other parts of the system.**
3. **Scalability – Layers can be modified, replaced, or scaled independently, which helps in accommodating growth without redesigning the entire system.**
4. **Testability – Layers allow you to test components in isolation (unit testing), which improves reliability and simplifies debugging.**
5. **Flexibility – Changing technologies or frameworks in one layer (e.g., switching databases) doesn’t necessarily impact other layers.**

**Q-13:-** **Explain the importance of a development environment in software production.**

**development environment is important in software production because it provides the tools and setup needed to write, test, and debug code efficiently.**

1. **Safe Testing Space: It provides a controlled environment to test code without affecting the live system or users. Bugs and errors can be identified and fixed safely.**
2. **Consistency: Ensures that all developers work with the same tools, libraries, and settings, reducing the chances of "it works on my machine" problems.**
3. **Efficiency: Offers tools like code editors, debuggers, and compilers that speed up coding, error detection, and troubleshooting.**
4. **Version Control & Collaboration: Integrated development environments (IDEs) often support version control systems, making teamwork easier by tracking changes and avoiding conflicts.**
5. **Simulation of Production Environment: It allows developers to mimic the live system closely, so they can foresee and prevent issues before deployment.**
6. **Automation: Development environments often include features like automated testing, builds, and deployment, which make production smoother and less error prone.**

**Q-14:- What is the difference between source code and machine code?**

| **Aspect** | **Source Code** | **Machine Code** |
| --- | --- | --- |
| **Definition** | **Human-readable instructions written by a programmer in a programming language (e.g., Python, Java, C++).** | **Low-level code made of binary (0s and 1s) that a computer’s CPU can execute directly.** |
| **Readability** | **Readable and understandable by humans.** | **Not human-readable; only understandable by machines.** |
| **Language** | **High-level programming languages.** | **Machine language is specific to processor architecture.** |
| **Execution** | **Cannot be executed directly by the CPU; requires compilation or interpretation.** | **Executed directly by the CPU.** |

**Q-15:-** **Why is version control important in software development?**

* **Version control is important in software development because it helps keep track of all the changes made to a project over time.**
* **It allows multiple developers to work on the same code without overwriting each other’s work, makes it easy to fix mistakes by going back to earlier versions, and helps organize updates in a clear, manageable way.**
* **Essentially, it makes teamwork smoother, reduces errors, and ensures the project’s history is safe and traceable**

**Q-16:- : What are the benefits of using GitHub for students?**

1. **Practice Version Control: Helps students learn how to track and manage changes in their projects.**
2. **Collaboration Skills: Students can work together on projects and learn team**
3. **Portfolio Building: Projects on GitHub can be shared with teachers, colleges, or future employers.**
4. **Access to Open Source: Students can contribute to real-world projects and gain experience.**
5. **Backup of Work: Keeps code safe online, so nothing is lost.**
6. **Learning Resources: Offers tutorials, examples, and community support.**

**Q-17:- What are the differences between open-source and proprietary software?**

| **Feature** | **Open-Source Software** | **Proprietary Software** |
| --- | --- | --- |
| **Access to Source Code** | **Source code is available to everyone** | **Source code is private and not shared** |
| **Cost** | **Usually, free** | **Usually paid** |
| **Modification** | **Users can modify and customize it** | **Users cannot modify it** |
| **Distribution** | **Can be shared freely** | **Cannot be shared freely** |
| **Support** | **Community-based support** | **Official company support** |
| **Examples** | **Linux, Firefox, LibreOffice** | **Windows, Adobe Photoshop, Microsoft Office** |

**Q-18:- How does GIT improve collaboration in a software development team?**

1. **Multiple People Can Work Together: Team members can work on the same project without overwriting each other’s changes.**
2. **Branches for Features: Each developer can create a separate branch to work on new features safely.**
3. **Easy Merging: Git combines changes from different developers smoothly.**
4. **Track Changes: Every change is recorded with the author and time, making it easy to review.**
5. **Conflict Resolution: Git helps detect and fix conflicts when multiple changes affect the same part of the code.**
6. **Backup: All work is stored in a repository, so nothing is lost.**

**Q-19:-** **What is the role of application software in businesses?**

**Application software helps businesses perform specific tasks efficiently. It is designed to meet the needs of users and can include programs like word processors, spreadsheets, accounting software, customer relationship management (CRM) tools, and inventory management systems.**

**Key roles include:**

1. **Automation of Tasks: Reduces manual work by automating routine operations like billing, payroll, and data entry.**
2. **Improved Productivity: Tools like spreadsheets, project management software, and document editors help employee’s complete tasks faster and more accurately.**
3. **Data Management: Helps store, organize, and analyze business data, making decision-making faster and more informed.**
4. **Communication and Collaboration: Software like email clients, chat apps, and video conferencing tools enable smooth communication within and outside the organization.**
5. **Customer Support: CRM software helps manage customer interactions, track sales, and improve service quality.**
6. **Financial Management: Accounting and finance software helps with budgeting, tracking expenses, and generating financial reports efficiently.**

**Q-20 What are the main stages of the software development process?**

**The main stages of the software development process are:**

1. **Requirement Analysis – Understanding what the users need and documenting the features and functions the software must have.**
2. **Design – Planning how the software will work, including its architecture, user interface, and data flow.**
3. **Implementation (Coding) – Writing the actual code based on the design.**
4. **Testing – Checking the software for errors or bugs to make sure it works correctly and meets the requirements.**
5. **Deployment – Installing and releasing the software so users can start using it.**
6. **Maintenance – Updating, fixing bugs, and improving the software after it’s in use.**

**Q-21: Why is the requirement analysis phase critical in software development**

**The requirement analysis phase is critical in software development because it lays the foundation for the entire project. In this phase, developers and stakeholders clearly define what the software should do, what problems it should solve, and what features are needed. If requirements are misunderstood or incomplete, it can lead to wrong design, wasted time, and increased costs later.**

1. **Understanding the client’s needs – ensures the final software meets expectations.**
2. **Reducing errors and risks – catching problems early saves a lot of trouble later.**
3. **Planning resources and timelines – knowing requirements helps estimate effort, cost, and schedule.**
4. **Guiding design and development – clear requirements make coding, testing, and maintenance smoother.**

**Q-22: What is the role of software analysis in the development process?**

**The role of software analysis in the development process is to understand and define the problem clearly before building the software. It is about studying the needs of the users and the business so that developers know exactly what the software must do.**

**Key points:**

1. **Identify requirements: Determine what the software should achieve and what features it must have.**
2. **Understand problems: Analyze existing systems or processes to find issues that the new software should fix.**
3. **Plan the solution: Provide a clear picture of how the software should work, helping designers and developers create the right system.**
4. **Reduce mistakes: Early analysis helps prevent misunderstandings, rework, and waste time later.**
5. **Support decision-making: Helps stakeholders decide priorities, resources, and timelines.**

**Q-23: What are the key elements of system design?**

**The key elements of system design define how a software system will work and are planned before coding starts. It includes deciding the overall structure of the system, how different parts connect, how users and other systems will interact, and how data will be stored and used. It also involves dividing the system into smaller modules, explaining how each part works step by step, and ensuring the system is secure. These elements help make the system organized, efficient, and useful.**

**Q-24 Why is software testing important?**

**Software testing is important because it ensures that the software works correctly and meets user requirements. It helps find and fix errors or bugs before the software is released, which prevents problems and saves time and cost later. Testing also improves the software’s quality, reliability, and performance, making it safe and efficient for users. Overall, it gives confidence that the software will function as expected in real situations.**

**Q-25 What types of software maintenance are there?**

* **Corrective Maintenance: Fixing mistakes or bugs in the software after it is released.**
* **Adaptive Maintenance: Changing the software to work with new devices, operating systems, or other updates.**
* **Perfective Maintenance: Making the software better by adding features or improving performance.**
* **Preventive Maintenance: Taking steps to avoid future problems and keep the software running smoothly.**

**Q-26:- What are the key differences between web and desktop applications?**

1. **Installation**

* **Web Applications: Run in a web browser; no installation required.**
* **Desktop Applications: Must be downloaded and installed on a specific device.**

1. **Accessibility**

* **Web Applications: Accessible from any device with internet and a browser.**
* **Desktop Applications: Tied to the device they’re installed on (unless cloud syncing is used).**

1. **Updates**

* **Web Applications: Updates are made on the server, so users always access the latest version automatically.**
* **Desktop Applications: Users often need to download and install updates manually (or via auto-update tools).**

1. **Performance**

* **Web Applications: Generally, depend on internet speed and browser capabilities; may be slower for heavy tasks.**
* **Desktop Applications: Usually faster and more powerful for resource-intensive tasks since they run directly on the hardware.**

1. **Offline Availability**

* **Web Applications: Usually we need an internet connection (though some offer limited offline functionality).**
* **Desktop Applications: Often work without internet installation.**

1. **Security**

* **Web Applications: More exposed to online threats; security depends heavily on server-side protections.**
* **Desktop Applications: Can be more secure in offline environments, but still vulnerable if not regularly updated.**

1. **Development and Deployment**

* **Web Applications: Easier to deploy and maintain across different platforms.**
* **Desktop Applications: May need separate versions for Windows, macOS, Linux, etc.**

**Q-27 What are the advantages of using web applications over desktop applications?**

1. **No Need to Install  
   You don’t have to download or install anything—just open a browser and start using the app.**
2. **Works on Any Device  
   Web apps run on computers, phones, or tablets, no matter what operating system they use (like Windows, Mac, or Android).**
3. **Always Up to Date  
   You always use the latest version because updates happen on the server automatically.**
4. **Use from Anywhere  
   As long as you have internet, you can access your web app from any place and device.**
5. **Easier to Fix and Maintain  
   If something breaks or needs changing, the developer can fix it once, and it works for everyone.**
6. **No High-End Computer Needed  
   Since the app runs through the browser, it doesn’t need a powerful computer.**
7. **Good for Teamwork  
   Many web apps let people work together in real time, like sharing documents or chatting live.**
8. **Saves Time and Money for Companies  
   Businesses can give access to many users easily, without setting up software on every computer.**

**Q-28 What role does UI/UX design play in application development?**

* **UI and UX design decide how an app looks and how easy it is to use.**
* **UI focuses on visuals like buttons, colors, and layout, while UX focuses on the user’s overall experience.**
* **Good UI/UX makes the app simple, smooth, and enjoyable for users.**
* **It helps keep users engaged and encourages them to return.**
* **Proper design can save time and money by avoiding problems later.**
* **Overall, UI/UX makes the app both attractive and user-friendly, which is crucial for its success.**

**Q-29 : What are the differences between native and hybrid mobile apps?**

**Native apps are developed specifically for one platform, like iOS or Android, using the platform’s own programming languages—Swift or Objective-C for iOS, and Kotlin or Java for Android. Because they are built for a specific platform, native apps have high performance, smooth animations, and full access to device features like the camera, GPS, and sensors. They also provide better user experience, but building separate apps for different platforms can be more time-consuming and costly.**

**Hybrid apps, on the other hand, are built using web technologies like HTML, CSS, and JavaScript and run inside a native container that allows them to work on multiple platforms. This makes development faster and cheaper because a single codebase can work on both iOS and Android. However, hybrid apps may have slower performance, limited access to device features, and sometimes feel less smooth compared to native apps.**

**Q-30 What is the significance of DFD’s system analysis?**

**Significance of DFDs in System Analysis:**

**DFDs, or Data Flow Diagrams, are important because they show how data moves through a system in a clear and simple way. They help analysts and developers understand how the system works, what data is needed, and where it goes.**

**They make complex systems easier to understand by breaking them into smaller parts. DFDs also help in finding problems like missing steps, repeated tasks, or inefficient data flow early, before building the system.**

**Moreover, they improve communication between technical teams and non-technical people, because even someone without a technical background can see how the system works.**

**Overall, DFDs are useful for planning, analyzing, and designing a system effectively.**

**Q-31 What are the pros and cons of desktop applications compared to web applications?**

**Desktop Applications**

**Pros:**

1. **Performance: They run directly on the user’s computer, so they are usually faster and can handle heavy processing.**
2. **Offline Access: Can often work without an internet connection.**
3. **Full Hardware Access: Can use all computer resources like GPU, RAM, and peripherals efficiently.**
4. **Security: Data is stored locally, reducing dependence on internet security (though local security matters).**

**Cons:**

1. **Installation Required: Must be installed on each device, which can be time-consuming.**
2. **Updates: Users have to update the software manually unless an auto-update system is in place.**
3. **Platform Dependency: Usually designed for a specific operating system (Windows, macOS, Linux), requiring different versions for each.**
4. **Limited Accessibility: Can’t be accessed from anywhere unless remote desktop tools are used.**

**Web Applications**

**Pros:**

1. **Accessibility: Can be accessed from any device with a web browser and internet connection.**
2. **No Installation Needed: Runs directly from the browser, saving storage space and setup time.**
3. **Automatic Updates: Updates are applied centrally on the server; users always get the latest version.**
4. **Cross-Platform: Works on multiple operating systems without modification.**

**Cons:**

1. **Internet Dependency: Usually requires an internet connection to function properly.**
2. **Performance Limitations: May be slower than desktop apps for resource-heavy tasks.**
3. **Limited Hardware Access: Cannot fully utilize system resources compared to desktop applications.**
4. **Security Risks: Data is stored on servers, making it potentially more vulnerable to cyberattacks.**

**Q-32 *:-*How do flowcharts help in programming and system design?**

**Flowcharts are visual diagrams that represent the sequence of steps, decisions, and processes in a program or system. They are very useful in programming and system design for several reasons:**

1. **Clarifying Logic: Flowcharts help programmers and system designers visualize the logic of a program or system before coding. This makes it easier to understand complex processes.**
2. **Simplifying Communication: They provide a clear, visual way to communicate ideas among team members, stakeholders, or clients who may not be familiar with programming.**
3. **Identifying Errors Early: By mapping out the flow of operations and decisions, potential logical errors or inefficiencies can be detected early, reducing debugging time later.**
4. **Planning and Documentation: Flowcharts serve as a blueprint for system design and as documentation for future maintenance or updates, helping others understand how the program works.**
5. **Improving Efficiency: By analyzing the flowchart, unnecessary steps or redundancies in the process can be identified and eliminated, improving overall system efficiency.**
6. **Decision Making: They clearly show decision points and alternative paths, which helps in designing programs that handle different scenarios effectively.**