***THEORY***

***What is a Program?***

***Q2) Explain in your own words what a program is and how it functions. What is Programming?***

***A2) A program is a set of instructions that tell a computer what to do.***

***Programming functions in following manner :-***

1. ***Input: A program receives data or input from a user or another source.***
2. ***Processing: The program then processes this input according to the instructions it has been given.***
3. ***Output: Finally, the program produces an output, which could be displayed on a screen, stored in a file, or used to control other devices.***

***Programming is the process of thinking, problem-solving, and writing the instructions that become a program. It's much more than just typing.***

***It involves:***

1. ***Planning & Logic: First, you have to understand the problem you want to solve. This is the most crucial part of programming.***
2. ***Writing: You then translate those logical steps into a specific programming language (like Python or Java), following its unique vocabulary and grammar rules (its syntax).***
3. ***Testing & Debugging: Programs almost never work perfectly on the first try. A huge part of programming is testing your code, finding mistakes (called "bugs"), and fixing them (a process called "debugging"). It's like a chef taste-testing the dish and adjusting the recipe until it's perfect.***

***Q3) What are the key steps involved in the programming process? Types of Programming Languages***

***A3) The key steps involved in the programming process are as follows:-***

***The programming process generally involves understanding the problem, planning the solution, writing the code, testing and debugging, and finally, documenting and optimizing the code.***

***Types of Programming Languages are as follows:-***

1. ***Procedural Programming***
2. ***Object-Oriented Programming***
3. ***Logical***
4. ***Functional***

***Q4) What are the main differences between high-level and low-level programminglanguages?***

***A4)***

| ***Parameters*** | ***High-Level Language*** | ***Low-Level Language*** |
| --- | --- | --- |
| ***Abstraction Level*** | ***High abstraction, closer to human language*** | ***Low abstraction, closer to machine code*** |
| ***Ease of Use*** | ***Easier to learn and use*** | ***More complex and harder to learn*** |
| ***Portability*** | ***Highly portable across different systems*** | ***Less portable, often system-specific*** |
| ***Development Speed*** | ***Faster development time*** | ***Slower development time*** |
| ***Examples*** | ***Python, Java, C++, JavaScript*** | ***Assembly language, Machine code*** |
| ***Memory Management*** | ***Automatic memory management*** | ***Manual memory management*** |
| ***Error Handling*** | ***Built-in error handling features*** | ***Limited error handling, requires manual checks*** |
| ***Performance*** | ***Generally slower execution*** | ***Generally faster execution*** |
| ***Use Cases*** | ***Application development, scripting, web development*** | ***System programming, embedded systems, device drivers*** |

***World Wide Web & How Internet Works***

***Q1) Describe the roles of the client and server in web communication.***

***A1) Client Role:***

1. ***Initiates Communication: Starts the interaction by sending a request to the server.***
2. ***Sends Requests: Makes HTTP/HTTPS requests (e.g., GET, POST) to access services or data.***
3. ***Receives and Displays Data: Handles and displays the server's response (e.g., webpage, JSON).***
4. ***User Interaction: Interfaces with the user (input, clicks, form submissions).***
5. ***Runs on User Devices: Typically browsers, apps, or software running on PCs, smartphones, etc.***

***Server Role:***

***Waits for Requests: Always running and listening for incoming client requests.***

1. ***Processes Requests: Interprets and processes the request (fetches data, runs logic).***
2. ***Sends Responses: Delivers appropriate responses (webpages, files, status codes).***
3. ***Manages Resources: Stores and manages databases, files, backend logic, etc.***
4. ***Runs on Powerful Machines: Hosted in data centers or cloud platforms (e.g., AWS, Azure).***

***Network Layers on Client and Server***

***Q1) Explain the function of the TCP/IP model and its layers.***

***A1)***

***Standardizes Network Communication***

* ***Provides a universal set of rules for devices to communicate across different networks.***

***Breaks Down Complex Tasks into Layers***

* ***Divides communication into manageable layers, each with a specific role (like sending data, routing, or displaying to the user).***

***Enables Interoperability***

* ***Allows different hardware and software systems to work together by following the same communication protocols.***

***Supports Reliable and Unreliable Communication***

* ***Offers both TCP (reliable) and UDP (fast but unreliable) for different use cases.***

***Manages Addressing and Routing***

* ***Ensures data is delivered to the correct destination using IP addresses and routing mechanisms.***

***Handles Error Detection and Correction***

* ***Ensures data is not lost or corrupted in transit (especially via TCP).***

***Facilitates End-to-End Communication***

* ***Maintains a connection from the source device to the destination across multiple networks and routers.***

***Types of Layers are as follows :-***

1. ***Presentation.***
2. ***Application***
3. ***Business***
4. ***Persistance***
5. ***Database***

***Client and Servers***

***Q1) : Explain Client Server Communication***

***Types of Internet Connections***

***Q1) How does broadband differ from fiber-optic internet?***

|  |  |  |
| --- | --- | --- |
| ***Feature*** | ***Cable Internet*** | ***Fiber-Optic Internet*** |
| ***The Core Technology*** | ***Electrical Signals sent over copper coaxial TV cables.*** | ***Pulses of Light sent through glass or plastic fibers.*** |
| ***Speed*** | ***Fast, typically ranging from 25 Mbps to 1,000 Mbps (1 Gbps).*** | ***Extremely Fast, often starting at 100 Mbps and commonly reaching 1,000 Mbps to 10,000 Mbps (10 Gbps).*** |
| ***Speed Symmetry*** | ***Asymmetrical. Download speeds are much faster than upload speeds (e.g., 400 Mbps download / 20 Mbps upload).*** | ***Symmetrical. Download and upload speeds are the same (e.g., 1,000 Mbps download / 1,000 Mbps upload).*** |
| ***Reliability & Durability*** | ***Less Reliable. Prone to slowdowns during peak usage times in your neighborhood (shared bandwidth). Susceptible to electrical interference and weather damage.*** | ***More Reliable. Each user often has a dedicated line to the node. Immune to electrical interference. Glass fibers are more durable and weather-resistant than copper.*** |
| ***Latency (Ping)*** | ***Higher Latency. Electrical signals travel slower and are more prone to delays.*** | ***Very Low Latency. Data travels at nearly the speed of light, resulting in less "lag." This is critical for gaming and video calls.*** |
| ***Availability*** | ***Widely Available. It uses existing cable TV infrastructure, which is present in most populated areas.*** | ***Less Available. Requires installing brand-new infrastructure, which is expensive and time-consuming. Availability is growing but is still limited, especially in rural areas.*** |

***Protocols***

***Q1) What are the differences between HTTP and HTTPS protocols?***

|  |  |
| --- | --- |
| ***HTTP*** | ***HTTPS*** |
| ***HTTP stands for HyperText Transfer Protocol.*** | ***HTTPS for HyperText Transfer Protocol Secure.*** |
| ***In HTTP, URL begins with “http://”.*** | ***In HTTPs, URL starts with “https://”.*** |
| ***HTTP uses port number 80 for communication.*** | ***HTTPs uses 443 port number for communication.*** |
| ***HTTP is considered to be unsecure.*** | ***HTTPs is considered as secure.*** |
| ***HTTP works at Application Layer.*** | ***HTTPS works at Transport Layer.*** |
| ***In HTTP, Encryption is absent.*** | ***Encryption is present in HTTPS.*** |
| ***HTTP does not require any certificates.*** | ***HTTPS needs SSL Certificates.*** |
| ***HTTP does not improve search ranking*** | ***HTTPS helps to improve search ranking*** |
| ***HTTP faster than HTTPS*** | ***HTTPS slower than HTTP*** |
| ***HTTP does not use data hashtags to secure data.*** | ***While HTTPS will have the data before sending it and return it to its original state on the receiver side.*** |
| ***In HTTP Data is transfer in plaintext.*** | ***In HTTPS Data transfer in ciphertext.*** |
| ***HTTP Should be avoided.*** | ***HTTPS Should be preferred.*** |
| ***Search engines do not favour the insecure website.*** | ***Improved reputation of the website in search engine.*** |
| ***HTTP Does not require SSL/TLS or Certificates*** | ***HTTPS Requires SSL/TLS implementation with Certificates.*** |
| ***In HTTP Users ar  worried about their data.*** | ***In HTTPS Users are  confident about the security of their data.*** |

***Application Security***

***Q1) What is the role of encryption in securing applications?***

***1. Protecting Data at Rest:***

* ***Data-at-rest encryption: This involves encrypting data stored on servers, databases, or devices.***
* ***Preventing unauthorized access: Even if a system is compromised, encrypted data remains unreadable without the decryption key, preventing sensitive information like user credentials, financial records, or personal data from being exposed.***
* ***Compliance: Many regulations, such as GDPR and HIPAA, mandate encryption to protect sensitive personal information.***

***2. Protecting Data in Transit:***

* ***Data-in-transit encryption: This protects data being transmitted between devices or over networks (e.g., emails, instant messages, file transfers).***
* ***Preventing interception: Encryption ensures that even if data is intercepted during transmission, it cannot be read or exploited by malicious actors.***
* ***Secure protocols: Protocols like SSL/TLS and VPNs utilize encryption to safeguard data during transit.***

***3. Enhancing Application Security:***

* ***Confidentiality: Encryption ensures that only authorized users with the correct keys can access the data, maintaining confidentiality.***
* ***Integrity: Encryption can help verify that data hasn't been tampered with during transmission or storage, ensuring data integrity.***
* ***Authentication: Encryption can be used to verify the identity of the sender or recipient, ensuring that the communication is secure.***
* ***Mitigating data breaches: If a breach occurs, encryption can minimize the impact by making the stolen data unreadable.***

***4. Specific Use Cases:***

* ***Mobile application security: Encryption protects sensitive data stored on mobile devices and transmitted over networks.***
* ***Cloud security: Cloud providers use encryption to protect data stored in their environments, both at rest and in transit.***
* ***IoT security: Encryption secures communication between IoT devices and other systems.***

***5. Key Concepts:***

* ***Encryption algorithm: A mathematical formula used to scramble data into an unreadable format.***
* ***Encryption key: A piece of information (either symmetric or asymmetric) used to encrypt and decrypt data.***
* ***Decryption: The process of converting encrypted data back into its original, readable format using the appropriate key.***

***LAB EXERCISE***

***What is a Program?***

***Q1) Write a simple "Hello World" program in two different programming languages of  
 your choice. Compare the structure and syntax.***

***A1) Python-print("Hello, World")***

***Java-public class HelloWorld {***

***public static void main(String[] args) {***

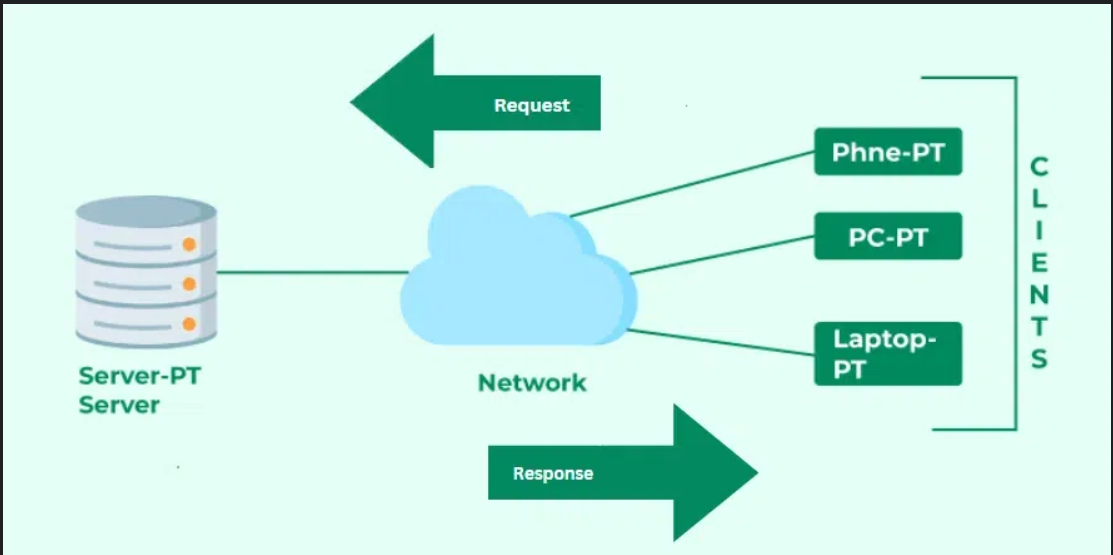
***System.out.println("Hello, World");***

***}***

***}***

***World Wide Web & How Internet Works***

***Q1) Research and create a diagram of how data is transmitted from a client to a server over the internet.***

******

***Network Layers on Client and Server***

***Q1) Design a simple HTTP client-server communication in any language.***

***Types of Internet Connections***

***Q1) Research different types of internet connections (e.g., broadband, fiber, satellite)and list their pros and cons***

***A1) There are mainly 8 types of internet connection :-***

***1. Dial-Up Connection***

***Pros:***

* ***Very cheap and widely available.***
* ***Uses existing phone lines — no new infrastructure.***
* ***Simple setup.***

***Cons:***

* ***Extremely slow (~56 kbps).***
* ***Cannot use phone and internet at the same time.***
* ***Obsolete for modern applications (streaming, gaming).***

***2. Broadband Connection***

***Pros:***

* ***Always-on connection.***
* ***Much faster than dial-up.***
* ***Supports multiple devices.***

***Cons:***

* ***Can be costlier than dial-up.***
* ***Speed depends on connection type (DSL, Cable, etc.).***
* ***May not be available in remote areas.***

***3. DSL (Digital Subscriber Line)***

***Pros:***

* ***Faster than dial-up and uses phone lines.***
* ***Can use phone and internet simultaneously.***
* ***Affordable and widely available.***

***Cons:***

* ***Speed drops with distance from ISP's central office.***
* ***Not as fast as cable or fiber.***
* ***Susceptible to line noise/interference.***

***4. Cable***

***Pros:***

* ***Faster speeds than DSL in most areas.***
* ***Good for streaming, gaming, video calls.***
* ***Widely available in urban areas.***

***Cons:***

* ***Shared bandwidth can slow down during peak hours.***
* ***Slightly more expensive than DSL.***
* ***Not ideal for rural regions.***

***5. Satellite Connection***

***Pros:***

* ***Available almost anywhere, including rural and remote areas.***
* ***Independent of local infrastructure (cables, towers).***

***Cons:***

* ***High latency due to distance (signal travels to space).***
* ***Slower speeds and weather-dependent.***
* ***Expensive with limited data plans.***

***6. Wireless Connection***

***Pros:***

* ***No physical cables needed.***
* ***Quick and easy to set up.***
* ***Good for rural areas with no wired infrastructure.***

***Cons:***

* ***Requires line-of-sight to the transmitter.***
* ***Signal can be disrupted by obstacles or weather.***
* ***Not as fast or stable as fiber or cable.***

***7. Cellular***

***Pros:***

* ***Portable — works anywhere with cell tower coverage.***
* ***5G offers high speed comparable to broadband.***
* ***Great for smartphones, hotspots, and travel.***

***Cons:***

* ***Data caps and throttling on most plans.***
* ***Signal strength varies by location.***
* ***More expensive per GB compared to wired plans.***

***8. ISDN (Integrated Service Digital Network)***

***Pros:***

* ***Supports both voice and data simultaneously.***
* ***Better quality and speed than dial-up.***
* ***Can combine channels for higher speed (128 kbps).***

***Cons:***

* ***Obsolete technology in most places.***
* ***Expensive compared to modern broadband.***
* ***Speeds too slow for modern internet needs.***

***Protocols***

***Q1) Simulate HTTP and FTP requests using command line tools (e.g., curl).***

***Application Security***

***Q1) Identify and explain three common application security vulnerabilities. Suggest possible solutions.***

1. ***SQL Injection (SQLi)***

***Attackers insert malicious SQL code into input fields, which the application then uses in database queries. This can allow them to bypass security measures, access or modify sensitive data, or even take control of the database.***

* ***Solutions:***
  + ***Input Validation: Strictly validate all user inputs to ensure they conform to expected formats and data types.***
  + ***Parameterized Queries/Prepared Statements: Use parameterized queries or prepared statements, which separate the SQL code from the user input, preventing it from being interpreted as code.***
  + ***Least Privilege: Grant database users only the minimum necessary permissions to perform their tasks.***

1. ***2. Cross-Site Scripting (XSS)***

***Attackers inject malicious scripts (usually JavaScript) into websites that are then executed by other users' browsers. This can lead to various attacks, including stealing cookies, redirecting users to malicious websites, or defacing the website.***

* ***Solutions:***
* ***Output Encoding: Encode user-provided data before displaying it on the webpage to prevent it from being interpreted as code.***
* ***Input Validation: Validate and sanitize user input to remove or escape any potentially harmful characters.***
* ***Content Security Policy (CSP): Implement CSP to control the resources (scripts, stylesheets, etc.) that a browser can load, limiting the impact of XSS attacks.***
* ***Whitelisting: Instead of blacklisting potentially harmful characters, create a whitelist of allowed characters or patterns.***

1. ***3. Broken Authentication***

***Weak or improperly implemented authentication mechanisms, such as weak passwords, session management issues, or lack of multi-factor authentication, allow attackers to bypass login and gain unauthorized access to accounts.***

* ***Solutions:***
* ***Strong Passwords: Enforce strong password policies, including length and complexity requirements.***
* ***Multi-Factor Authentication (MFA): Implement MFA to add an extra layer of security beyond passwords.***
* ***Secure Session Management: Use strong session IDs, expire sessions properly, and protect session cookies.***
* ***Rate Limiting: Limit the number of login attempts to prevent brute-force attacks.***
* ***Regular Security Audits: Conduct regular security audits to identify and address weaknesses in authentication and authorization.***

1. ***What are the differences between open-source and proprietary software…?  
   Open-source and proprietary software differ primarily in source code accessibility and licensing.***

|  |  |
| --- | --- |
| ***Open-Source Software*** | ***Proprietary Software*** |
| ***The source code is accessible to anyone.*** | ***Source code is kept secret and is not publicly available.*** |
| ***Typically released under licenses that grant broad usage, modification, and distribution rights.*** | ***Users typically require a license to use the software, with restrictions on modification and distribution.*** |
| ***Often free to use, but may have associated costs for support or specific features.*** | ***Usually requires purchase of a license.*** |
| ***Development and maintenance are often collaborative, with contributions from a community of users and developers.*** | ***Development and maintenance are controlled by the company that owns the software.*** |
| ***Highly customizable due to access to the source code.*** | ***Customization options are typically limited to what the vendor provides.*** |
| ***Example : Linux, Android, OpenOffice*** | ***Examples: Windows, macOS, Microsoft Office, Adobe Photoshop*** |

***2. GIT and GITHUB  
revolutionizing code management and collaboration***

|  |  |
| --- | --- |
| ***GIT*** | ***GITHUB*** |
| ***First developed in 2005*** | ***GitHub is designed as a Git repository hosting service*** |
| ***Git is installed and maintained on your local system (rather than in the cloud)*** | ***GitHub is exclusively cloud-based*** |
| ***One thing that really sets git apart is its branching model*** | ***You can share your code with other , giving them the power to make revision or edits*** |
| ***Git hub is a high quality version control system.*** | ***GitHub is a cloud-based hosting service .*** |

***3. How does GIT improve collaboration in a software development team….?  
Git enhances collaboration in software development teams by enabling parallel work on different parts of a project, tracking changes, and facilitating seamless integration through branching and merging.  
Benefits :  
1. Version Control : Every change made to the codebase is tracked, allowing developers to revert to previous versions if needed and providing a clear history of the project.  
2. Branching and Merging : Git's branching and merging features facilitate collaboration by allowing developers to isolate their work, experiment with new ideas, and integrate their changes back into the main codebase smoothly.      
3. Distributed Development : By working on separate branches and using pull requests, developers can minimize conflicts and ensure a more streamlined integration process.    
4. Change Tracking: GIT's commit history provides a clear record of changes, allowing team members to track progress, identify issues, and review code changes.   
5. Collaboration Tools: GIT integrates seamlessly with collaboration tools like GitHub, GitLab, or Bitbucket, facilitating code reviews, issue tracking, and project management.  
Collaboration Advantages***

***1. Reduced Conflicts: GIT's version control and branching features minimize conflicts between team members, allowing for smoother collaboration.***

***2. Improved Communication: GIT's change tracking and commit history facilitate communication among team members, ensuring everyone is on the same page.***

***3. Faster Integration: GIT's merging features enable faster integration of changes, reducing the time and effort required to combine code from multiple developers.***

***4. Increased Transparency: GIT's transparent commit history and change tracking provide visibility into the development process, fostering trust and accountability among team members.  
  
4. Application Software…?  
Application software is a type of computer program designed to perform specific tasks for end-users, allowing them to accomplish things like creating documents, browsing the internet, or managing data.  
Example : Web Broser , Spreadsheet , Database etc…***

***5. What is the role of application software in businesses….?  
Application software plays a crucial role in businesses by streamlining operations, boosting productivity, and enabling informed decision-making. It helps businesses manage data, automate tasks, enhance communication, and improve overall efficiency. Specifically, it enables businesses to manage customer interactions, financial transactions, inventory, and more.   
Key Role   
Streamline Operations : Application software automates business processes, reducing manual errors and increasing efficiency.  
Improved Efficiency: By streamlining workflows and optimizing processes, application software helps businesses complete tasks faster and more accurately, leading to increased productivity.***

***Enhanced Communication : Many applications facilitate communication and collaboration among employees, fostering teamwork and improving information sharing.   
4. Facilitate Collaboration : Application software enables team members to collaborate and share information, promoting communication and teamwork.***

***5. Support Customer Interactions : Application software helps businesses manage customer interactions, providing better customer service and experience.  
businesses benefits :   
Increased Productivity: Automation, streamlined processes, and improved communication lead to increased productivity across the organization.***

***Reduced Costs : Automating tasks and optimizing processes can lead to cost savings in terms of labor, resources, and materials.   
Enhanced Customer Experience : Applications like CRM can improve customer service, leading to increased customer satisfaction and loyalty.***

***Competitive Advantage: Businesses that effectively utilize application software gain a competitive advantage through increased efficiency, improved decision-making, and enhanced customer experience.  
Examples :  
Customer Relationship Management (CRM) : CRM software helps businesses manage customer interactions, track sales leads, and improve customer service.***

***Enterprise Resource Planning (ERP) : ERP systems integrate various business functions, such as finance, human resources, and supply chain management, into a unified platform.***

***Project Management Software : This type of software helps businesses plan, organize, and track projects, ensuring timely completion and efficient resource allocation.  
6. Software Development Process….?  
The software development process, also known as the Software Development Life Cycle (SDLC), is a structured approach to building software applications.  
Software Development Life Cycle (SDLC) Process  
1. Planning  
2. Analysis  
3. Design (look)  
4. Coding (Backed)  
5. Testing  
6. Maintenance   
  
7. Software Requirement…?  
A software requirement is a capability or condition that a software system must possess to fulfill user needs or solve a problem.  
Types of Software Requirement  
1. Functional requirements :  The specific actions and behaviors that a software system or product must perform to meet user needs.   
2. Non-functional requirements : Define the quality attributes of the software, such as performance, security, usability, and reliability. Examples include response time, security protocols, or user interface design.   
  
8. Why is the requirement analysis phase critical in software development…?***

***The requirement analysis phase is critical in software development because it ensures that the final product meets the needs of stakeholders and avoids costly rework later in the development process.   
 key reason  
Reduces Risk and Rework : Thorough requirement analysis helps identify potential issues and risks early on, allowing for adjustments before significant development effort is invested, thus minimizing costly rework.  
Improves Communication and Collaboration : It fosters clear communication and collaboration between stakeholders and the development team, ensuring everyone is on the same page regarding the project's goals and requirements.   
Ensures Customer Satisfaction : By thoroughly understanding and documenting stakeholder needs, the requirement analysis phase helps deliver a product that meets their expectations, leading to higher customer satisfaction.  
Guides Development and Testing : The documented requirements serve as a blueprint for development and testing, ensuring that the final product is built according to specifications and functions as intended.   
Provides a Foundation for Success : A well-defined requirements phase lays a solid foundation for the entire software development lifecycle, increasing the likelihood of a successful project.  
  
9. Software Analysis…?  
Software analysis is a crucial part of the software development process that involves understanding and defining the requirements of a software system, as well as designing a system that meets those requirements.  
  
Types of Software Analysis :  
Dynamic Analysis : This involves analyzing the behavior of the software while it is running, often by monitoring its execution and observing its behavior.  
Static Analysis : This involves analyzing the source code or binary code of the software without actually running it, often to identify potential errors or vulnerabilities.  
Hybrid Analysis : This combines aspects of both dynamic and static analysis to get a more comprehensive understanding of the software.***

***10. What is the role of software analysis in the development process…?***

***Software analysis plays a crucial role in the development process by bridging the gap between user needs and the final software product.  
  
Role of software analysis :   
1. Requirements Gathering and Definition: Software analysis begins with gathering requirements from stakeholders, including users, clients, and other relevant parties.  
2. Requirements Analysis and Validation : Once requirements are gathered, they are analyzed to ensure they are clear, complete, consistent, and feasible. This involves identifying potential conflicts, ambiguities, and gaps in the requirements. Validation ensures that the defined requirements accurately reflect the needs of the users and stakeholders.  
3. Design and Architecture : Software analysis informs the design and architecture of the software system. It helps in creating a well-structured and modular design that is easier to maintain and update in the future.  
4. Risk Management : By identifying potential risks early in the development process, software analysis helps in mitigating those risks and ensuring the software system is reliable and secure.  
5. Communication and Collaboration : Software analysis models facilitate communication and collaboration among different stakeholders, including users, developers, and testers.  
6. Test Planning : A detailed analysis of requirements is essential for developing comprehensive test plans and strategies that align with the project's objectives.  
7. Reduced Development Costs : By identifying and mitigating risks early, software analysis helps in reducing the overall cost of software development.  
  
11. System Design…?  
System design is the process of defining the architecture , components, interfaces, and data of a system to meet specific requirements.  
Creating a blueprint : It’s the detailed plan that guides the development process.  
Balancing trade-off : System design involves making choices between different options based on requirements and constraints.  
  
12. What are the key elements of system design….?  
architectural patterns, data management, APIs and communication, security, scalability and performance, and reliability.      
  
key elements of System Design :  
  
1. Architecture:***

***High-level structure:Defines the overall organization of the system, including how different components interact.   
Architectural patterns:Microservices, client-server, monolithic, and service-oriented architectures are examples.***

***2. Data Management:  
Database design: Choosing the right database (SQL, NoSQL) and structuring data for efficient access.   
Data flow: Understanding how data moves through the system is critical.   
Caching: Using caching mechanisms to improve performance by storing frequently accessed data.   
Storage: Handling different types of data, such as files, databases, and blob storage.   
Data consistency: Maintaining data integrity across the system.***

***3. APIs and Communication:***

***APIs (Application Programming Interfaces): Define how different parts of the system interact, including communication protocols and data formats.   
Communication protocols: Establishing how different components exchange information.   
API Gateways: Manage incoming API requests, authentication, and routing.***

***4. Security:   
Authentication and authorization: Ensuring only authorized users can access the system.  
Encryption: Protecting sensitive data from unauthorized access.  
Security audits: Regularly reviewing security practices to identify and address vulnerabilities.***

***5. Scalability and Performance:  
Load balancing: Distributing traffic across multiple servers to prevent overload.   
Scaling: Adding resources to handle increased load, either vertically (more powerful servers) or horizontally (more servers).   
Monitoring systems: Tracking system performance and identifying bottlenecks.   
Rate limiting: Limiting the number of requests to prevent abuse and ensure availability.***

***6. Reliability:***

***Redundancy: Having backup systems to handle failures and maintain availability.   
Fault tolerance: Designing the system to continue operating even if some components fail.   
Monitoring systems: Detecting and alerting on issues to enable timely responses.***

***7. Additional Components:  
Content Delivery Networks (CDNs): Distributing content geographically to improve delivery speed.   
Key-value stores: Storing data as key-value pairs for efficient access.   
Queues: Facilitating asynchronous communication between components.   
Distributed unique ID generators: Generating unique identifiers across distributed systems.  
  
13. Software Testing….?  
Why is software testing important….?  
Software testing is crucial for ensuring the reliability, security, and performance of software applications, leading to higher customer satisfaction and reduced costs.  
   
Important of Software Testing  
1. Bug Detection and Prevention 2. Improved Quality and Reliability  
3. Cost Reduction 4. Enhanced Security 5. Meeting User Expectations  
6. Compliance with Regulations 7. Competitive Advantage***

***14. Maintenance….?  
Software maintenance is the process of modifying and updating software after its initial release to correct faults, improve performance, adapt to changing environments, or add new features.  
  
15. What types of software maintenance are there..?  
Types of Software Maintenance:***

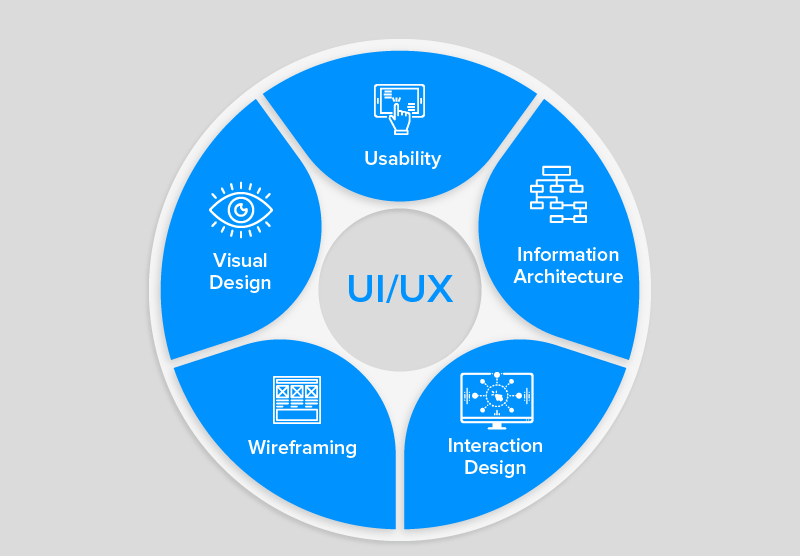
***1. Corrective Maintenance 2. Adaptive Maintenance   
3. Perfective Maintenance 4. Preventive Maintenance  
  
16. Development….?  
Software development is the process of designing, creating, testing, and maintaining software applications or programs.  
  
17. What are the key differences between web and desktop applications…?***

|  |  |  |
| --- | --- | --- |
|  | ***Web Applications*** | ***Desktop Applications*** |
| ***Accessibility*** | ***Accessed through a web browser (like Chrome, Firefox, Safari).*** | ***Accessed by launching the application directly from the computer's operating system.*** |
| ***Installation*** | ***No installation required on the user's device.*** | ***Accessed by launching the application directly from the computer's operating system.*** |
| ***Internet Dependency*** | ***Requires an internet connection to function.*** | ***Some desktop applications require internet connectivity for certain features, but many can operate offline.*** |
| ***Updates*** | ***Updates are managed by the application provider and are usually applied automatically.*** | ***Updates are typically managed by the user, requiring downloads and installations.*** |
| ***Cross-platform*** | ***Can be accessed on various operating systems and devices as long as they have a compatible browser.*** | ***Often developed specifically for a particular operating system (Windows, macOS, Linux).*** |
| ***Examples*** | ***Gmail, Google Docs, Facebook.*** | ***Microsoft Office (Word, Excel, PowerPoint), Adobe Photoshop, games.*** |

***18. Web Application…?  
A web application is a software program that is accessed via a web browser over the internet.  
  
19. What are the advantages of using web applications over desktop applications…?  
Web applications offer several advantages over desktop applications, primarily due to their accessibility, ease of maintenance, and scalability.  
Platform Independence: Web applications are accessible from any device with a web browser, regardless of the operating system (Windows, macOS, Linux, etc.).  
Cross-Device Access: Users can access web applications from their work computer, home laptop, or even mobile devices.  
No Installation Required: Web applications don't require users to download or install software, streamlining access and eliminating the need for end-user maintenance***

***20. Designing….?***

***Software design is the process of planning, creating, and documenting the structure and behaviour of a software system to fulfil specific requirements.***

***21. What role does UI/UX design play in application development…?  
  
UI/UX design plays a crucial role in application development by ensuring the application is both visually appealing and easy to use, leading to increased user satisfaction and engagement.  
  
  
  
22. Mobile Application  
  
A mobile application, or app, is a software program designed to run on mobile devices like smartphones and tablets.***

***23. What are the differences between native and hybrid mobile apps…?  
  
Native and hybrid mobile apps differ in their development approach, performance, user experience, and cost.  
  
Key Differences Summarized:***

|  |  |  |
| --- | --- | --- |
| ***Feature*** | ***Native Apps*** | ***Hybrid Apps*** |
| ***Development*** | ***Platform-specific languages (Swift, Kotlin)*** | ***Web technologies (HTML, CSS, JavaScript)*** |
| ***Performance*** | ***Generally faster and more responsive*** | ***Can be slower, especially with complex UI/UX*** |
| ***User Experience*** | ***Optimized for platform, fluid and responsive*** | ***Can be less consistent or responsive*** |
| ***Cost*** | ***More expensive*** | ***Less expensive*** |
| ***Development Time*** | ***Longer*** | ***Shorter*** |
| ***Platform Compatibility*** | ***Single platform (iOS or Android)*** | ***Cross-platform (iOS and Android)*** |
| ***Access to Device Features*** | ***Full access*** | ***Limited access*** |
| ***Maintenance*** | ***Can be more complex with separate codebases for each platform*** | ***Easier to maintain with a single codebase*** |

***DFD(Data Flow Diagram)***

***Q30) LAB EXERCISE: Create a DFD for a hospital management system.***

***Ans.***

***>Theyory : what is the significane of DFD in system analysis?***

* ***DFD means (Data Flow Diagram).***
* ***DFD is a. significant modeling technique for analyzing and constructing information processes.***
* ***Higher levels of DFD provide a broad overview of the system, while lower levels provide more detail about the system's processes, data flows, and data stores.***
* ***Visualizing Data Flow:***

***DFDs provide a clear and concise way to understand how data moves through a system. They are particularly useful for complex systems where it can be difficult to grasp the data flow without a visual aid.***

* ***System Analysis:***

***DFDs help in analyzing the existing system by identifying its components, their interactions, and potential bottlenecks or inefficiencies.***

* ***System Design:***

***DFDs are used to design new systems by providing a blueprint for how data should flow and be processed.***

* ***Communication:***

***DFDs serve as a common language between technical and non-technical stakeholders, facilitating communication and collaboration during the system development lifecycle.***

* ***Documentation:***

***DFDs act as a valuable form of documentation, capturing essential information about the system's data flow and processes.***

***Desktop application***

***Q31)LAB EXERCISE:Bulid a simple Desktop caclutor application using a gui library.***

***Ans.. A basic desktop calculator application can be constructed using a GUI library such as Python's Tkinter.***

***>The process involves creating a main window, an entry field for display, and buttons for numbers and operations.***

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

***>Pros is a These are the positive aspects, benefits, or merits of a particular thing.***

***>Cons is a These are the negative aspects, drawbacks, or disadvantages of something.***

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ***Aspect*** | | ***Web Application*** | ***Desktop Application*** |  |
| ***Deployment*** | | ***Accessed via web browsers.*** | ***Installed on local computers.*** |  |
| ***Platform***  ***Compatibility*** | | ***Compatible with various devices and operating systems.*** | ***Typically designed for specitic operating systems.*** |  |
| ***Internet***  ***Connection*** | | ***Requires internet connection.*** | ***Can operate offline(depending on functionalty).*** |  |
| ***Updates*** | | ***Updates applied centrally.*** | ***Updates need to be manually installed by users.*** |  |
| ***Development***  ***Technology*** | | ***Typically uses web technologies. (HTML,CSS,JavaScript)*** | ***Utilazes programming languages(Java,c#,etc..)*** |  |
| ***storage*** | | ***Data stored on remote servers.*** | ***Data stored locally or on servers .***  ***(depending on a app)*** |  |
| ***accessbility*** | | ***Accessible from anywhere with internet access.*** | ***Resticted to device with installed software.*** |  |
| ***Resource***  ***Consumption*** | | ***Relies on server resources for processing.*** | ***Utillizes local computer resources.*** |  |
| ***Security*** | ***Vunearable to web\_based attacks.*** | | ***Vunearable to local attacks.*** |
| ***Customization*** | ***Limited customization options.*** | | ***Highly customizable based on user requriements.*** |

***Flow Chart***

***Q32)draw a flow chart respresentinfg the logic of basic online registeration system.***

***Ans.***

***-> How do flowcharts help in programming and system design?***

* ***Flowcharts is a visual representation of a program's logic, aiding in planning, understanding, and debugging.***
* ***They help in breaking down complex problems into smaller, manageable steps, making it easier to design, communicate, and troubleshoot code.***

***->Flowcharts help in programming:***

***1. Planning and Design:***

* ***Visualizing the Logic:***

***Flowcharts provide a clear, visual representation of the program's flow, making it easier to understand the overall structure and logic before writing any code.***

* ***Breaking Down Complexity:***

***They help in breaking down complex problems into smaller, sequential steps, making the design process more manageable.***

* ***Identifying Potential Problems:***

***Flowcharts can highlight potential issues or inefficiencies in the program's logic early in the development process, allowing for adjustments before coding.***

***2. Communication and Collaboration:***

* ***Explaining the Logic:***

***Flowcharts provide a clear and concise way to communicate the program's logic to other programmers or stakeholders.***

* ***Team Collaboration:***

***They can be used to facilitate discussions and collaboration among team members, ensuring everyone understands the program's flow.***

***3. Debugging and Troubleshooting:***

* ***Identifying Errors:***

***Flowcharts can be used to trace the execution path of the program and pinpoint the exact location of errors.***

* ***Simplifying Debugging:***

***By visualizing the flow of data and control, flowcharts can simplify the debugging process, making it easier to identify the root cause of issues.***

***4. Documentation:***

* ***Creating Clear Documentation:***

***Flowcharts serve as valuable documentation for the program, making it easier to understand and maintain the code in the future.***

* ***Understanding Existing Code:***

***They can also be used to understand the logic of existing programs, especially when the original documentation is missing or unclear.***

***In essence, flowcharts are a powerful tool that helps programmers plan, design, communicate, debug, and document their code, leading to more efficient and effective software development.***

***-->Flowcharts help in system design:***

***provide a visual representation of a system's workflow, making it easier to understand, design, and troubleshoot***

* ***Visualization and Understanding:***

***Flowcharts transform complex system logic into a visual format, making it easier for designers, developers, and other stakeholders to grasp the system's overall structure and functionality.***

* ***Early Problem Detection:***

***By mapping out the workflow, flowcharts can help identify potential bottlenecks, inefficiencies, or logical errors early in the design process. This allows for timely adjustments and optimizations, preventing costly rework later in the development cycle.***

* ***Improved Communication and Collaboration:***

***Flowcharts serve as a common visual language, enabling clear communication and collaboration among different teams involved in the system design process.***

* ***Documentation:***

***Flowcharts act as a valuable form of documentation for the system's design, providing a clear and concise representation of the system's workflow.***

* ***Debugging and Maintenance:***

***Flowcharts can be used to debug and maintain the system, as they provide a visual reference for tracing the flow of data and logic.***

***Software Applications and Its Types***

***Software Applications and Its Types***

***LAB EXERCISE: Identify and classify 5 applications you use daily as either system software or application software.***

|  |  |
| --- | --- |
| ***Application Name*** | ***Type*** |
| ***Windows 10*** | ***System Software*** |
| ***Google Chrome*** | ***Application Software*** |
| ***MS Word*** | ***Application Software*** |
| ***Quick Heal Antivirus*** | ***System Software (Utility)*** |
| ***File Explorer*** | ***System Software*** |

***THEORY EXERCISE: What is the difference between system software and application software?***

* ***System Software: Controls and manages computer hardware. Example: Operating Systems, Device Drivers.***
* ***Application Software: Helps the user perform specific tasks. Example: MS Word, Browsers.***

***Software Architecture***

***LAB EXERCISE: Design a basic three-tier software architecture diagram for a web application.***

1. ***Presentation Layer: User interface (HTML, CSS, JS).***
2. ***Business Logic Layer: Application processing (Java, Python, PHP).***
3. ***Data Layer: Database storage (MySQL, MongoDB).***

***THEORY EXERCISE: What is the significance of modularity in software architecture?***

* ***Modularity allows easier maintenance, scalability, code reusability, and debugging by dividing software into independent modules.***

***Layers in Software Architecture***

***LAB EXERCISE: Create a case study on functionality of presentation, business logic, and data access layers.***

***Example: Online Shopping System***

* ***Presentation Layer: Displays product pages, cart, checkout UI to user.***
* ***Business Logic Layer: Processes orders, payment, inventory management.***
* ***Data Access Layer: Retrieves and updates product and user data in database.***

***THEORY EXERCISE: Why are layers important in software architecture?***

* ***They separate concerns, improve maintainability, scalability, and enable team collaboration on different layers.***

***Software Environments***

***LAB EXERCISE: Explore different types of software environments (development, testing, production). Set up a basic environment in a virtual machine.***

* ***Development: For coding and initial testing.***
* ***Testing: For bug testing and QA.***
* ***Production: Live environment for end users.***

***Setup example: Install Ubuntu on VirtualBox with Python and VS Code for development.***

***THEORY EXERCISE: Explain the importance of a development environment in software production.***

* ***Provides isolated space for coding and testing without affecting production; ensures compatibility and reduces risks.***

***Source Code***

***LAB EXERCISE: Write and upload your first source code file to GitHub.***

***Example:***

***# hello.py***

***print("Hello, GitHub!")***

***Steps:***

1. ***Create repository***
2. ***Commit file***
3. ***Push to GitHub***

***THEORY EXERCISE: What is the difference between source code and machine code?***

* ***Source Code: Human-readable programming language.***
* ***Machine Code: Binary instructions executed by CPU.***

***GitHub and Introductions***

***LAB EXERCISE: Create a GitHub repository and document how to commit and push code changes.***

***Commands:***

* ***git init***
* ***git add .***
* ***git commit -m "Initial commit"***
* ***git branch -M main***
* ***git remote add origin <repo-url>***
* ***git push -u origin main***

***THEORY EXERCISE: Why is version control important in software development?***

* ***Tracks changes, enables collaboration, rollback, and avoids code conflicts.***

***Student Account in GitHub***

***LAB EXERCISE: Create a student account on GitHub and collaborate on a small project with a classmate.***

***Steps:***

1. ***Sign up at github.com***
2. ***Create repository***
3. ***Add classmate as collaborator***
4. ***Both push/pull changes***

***THEORY EXERCISE: What are the benefits of using GitHub for students?***

* ***Builds portfolio, practices version control, enhances collaboration, and improves job opportunities.***

***Types of Software***

***LAB EXERCISE: Create a list of software you use regularly and classify them.***

|  |  |
| --- | --- |
| ***Software*** | ***Type*** |
| ***Windows 10*** | ***System Software*** |
| ***MS Excel*** | ***Application Software*** |
| ***Disk Cleanup*** | ***Utility Software*** |
| ***VLC Media Player*** | ***Application Software*** |
| ***Device Driver*** | ***System Software*** |

***THEORY EXERCISE: Differences between open-source and proprietary software.***

* ***Open-source: Free to modify, access source code. Example: Linux.***
* ***Proprietary: Owned by companies, restrictions on modification. Example: Windows.***

***GIT and GITHUB Training***

***LAB EXERCISE: Follow a GIT tutorial to practice cloning, branching, and merging repositories.***

***Basic commands:***

* ***git clone <repo-url>***
* ***git checkout -b new-branch***
* ***git merge new-branch***

***THEORY EXERCISE: How does GIT improve collaboration in a software development team?***

* ***Enables multiple developers to work together, manage changes, merge code efficiently, and maintain history.***