**THEORY EXERCISE:**

**Module 1 – Overview of IT Industry**

**What is a Program?**

1. Explain in your own words what a program is and how it functions.

* A program is a set of instructions telling a computer what to do. It functions by these instructions being translated into a language the computer understands, and then the computer's central processing unit (CPU) executes them step-by-step to achieve a specific task.

**What is Programming?**

1. What are the key steps involved in the programming process?

* he key steps in the programming process are:

1) Understanding the problem and defining its requirements,

2) Designing a solution,

3) Coding the solution,

4) Testing and debugging the code,

5) Documenting the program, and

6) Maintaining the program.

**Types of Programming Languages**

1. What are the main differences between High level and Low level Programming Languages?

* High-level languages are designed for human readability and ease of use, while low-level languages are closer to machine code and offer greater control over hardware.

**World Wide Web & how Internet works**

1. Describe the role of the client and server in the Web communication.

* Client: This is typically your web browser (e.g., Chrome, Firefox) or a mobile app. Its role is to request information or services from the server and then display/present the received content to the user.
* Server: This is a powerful computer or program that stores and provides web content (like web pages, images, data) and services. Its role is to listen for client requests, process them, and then send back the appropriate response.

**Network Layers on Client and Server**

1. Explain the function of the TCP/IP model and it’s layers.

* The TCP/IP model is a framework that describes how data is transmitted across networks, including the internet. Its function is to provide a reliable, end-to-end communication system by breaking down the complex process into smaller, manageable layers.
* Here are its four main layers and their basic functions:

1. **Application Layer:** (Closest to the user) Deals with specific applications and services. It prepares user data for network transmission and handles protocols like HTTP (web Browse), FTP (file transfer), and SMTP (email).
2. **Transport Layer:** (Host-to-host communication) Responsible for reliable, end-to-end data delivery. It breaks data into segments and ensures they arrive correctly and in order (TCP) or provides faster, less reliable delivery (UDP).
3. **Internet Layer (Network Layer):** (Routing packets) Adds logical addresses (IP addresses) to data packets and determines the best path for them to travel across different networks to their destination.
4. **Network Access Layer (Link Layer):** (Physical transmission) Handles the actual physical transmission of data over the network medium (e.g., Ethernet cables, Wi-Fi). It deals with hardware addresses (MAC addresses) and converts data into signals for transmission.

**Client and Servers**

1. Explain Client Server Communication.

* Client-server communication is a fundamental networking concept where a client, like a web browser, sends a request to a server, which then processes the request and sends back a response.

**Types of Internet Connections**

1. How does broadband differ from fiber-optic internet?

* Broadband is a general term for high-speed internet access, while fiber optic internet is a specific type of broadband technology that uses thin glass or plastic fibers to transmit data. Fiber optic internet is generally faster and more reliable than other types of broadband like DSL or cable, which use copper wires.

**Protocols**

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

* HTTP is an insecure protocol, transmitting data as plain text, making it vulnerable to interception and tampering. HTTPS, on the other hand, is a secure protocol that encrypts data using SSL/TLS, protecting sensitive information from eavesdropping and ensuring data integrity.

**Application Security**

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

* Role of Encryption in Securing Applications: Encryption ensures no one can read communications or data except the intended recipient or data owner. This prevents attackers from intercepting and accessing sensitive data.

**Software Applications and It’s Types**

1. What is the difference between system software and application software?

* System software manages the computer's hardware and provides a platform for other software, while application software is designed to perform specific tasks for the user.

**Software Architecture**

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

* Modularity in software architecture is significant because it enables breaking down complex systems into smaller, manageable, and independent components (modules). This approach enhances various aspects of software development and maintenance, including improved organization, reusability, testing, scalability, and flexibility.

**Layers in Software Architecture**

1. Why are layers important in software architecture?

* Layers in software architecture are important because they separate concerns, making software modular, easier to maintain, scale, reuse, and test by assigning specific responsibilities to each distinct part.

**Software Environments**

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

* A development environment is a dedicated workspace (tools, software, configurations) where developers write, test, and debug code without affecting live users.

**Source Code**

1. What is the difference between source code and machine code?

* Source code is human-readable instructions written in a programming language, while machine code is the low-level binary code that a computer's processor directly executes.

**Github and Introductions**

1. Why is version control important in software development?

* Version control is crucial because it tracks all code changes, enabling team collaboration, allowing reversion to previous versions, helping resolve conflicts, supporting safe experimentation, and providing a change history for accountability.

**Student Account in Github**

1. What are the benefits of using Github for students?

* GitHub benefits students by offering free professional developer tools (via the Student Developer Pack), teaching essential version control skills, providing collaboration experience for group projects, enabling portfolio building through public repositories, and fostering community engagement for learning and networking.

**Types of Software**

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

* Open-source software provides publicly accessible source code and user freedom (to use, modify, distribute), often developed by a community. Proprietary software keeps its source code private, requires a paid license with restrictions, and is developed by a single company.

**GIT and GITHUB Training**

1. How does GIT improve collaboration in a software development team?

* Git improves team collaboration by enabling parallel development via branching, allowing seamless integration of code through merging, maintaining a complete change history for accountability, facilitating code review before integration, and supporting distributed workflows where developers can work independently before synchronizing.

**Application Software**

1. What is the role of application software in businesses?

* Application software plays a crucial role in businesses by streamlining operations, enhancing productivity, and enabling informed decision-making. It helps automate tasks, manage data, improve communication, and ultimately drive business growth.

**Software Development Process**

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

* The software development process, also known as the Software Development Life Cycle (SDLC), typically includes these main stages: Planning, Analysis, Design, Development, Testing, Deployment, and Maintenance.

**Software Requirement**

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

* The requirement analysis phase is critical in software development because it ensures that the software being built aligns with the needs and expectations of its users and stakeholders.

**Software Analysis**

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

* Software analysis plays a crucial role in defining and shaping the software development process. It involves understanding and documenting the specific needs of stakeholders, which is essential for creating a successful and functional software product.

**System Design**

1. What are the key elements of system design?

* Key elements of system design include architecture, data flow, scalability, reliability, security, performance, maintainability, APIs and interfaces, and database design.

**Software Testing**

1. Why is software testing important?

* Software testing is crucial because it ensures that applications work correctly, are reliable, and meet user expectations before release. Thorough testing helps identify and fix bugs early, reducing the risk of costly errors and reputational damage later on. It also improves customer satisfaction, enhances software performance, and ultimately leads to a more robust and trustworthy product.

**Maintenance**

1. What types of software maintenance are there?

* Software maintenance can be categorized into four main types: corrective, adaptive, perfective, and preventive.

**Development**

1. What are the key differences between web and desktop applications?

* Web applications are accessed through a web browser and typically require an internet connection, while desktop applications are installed directly on a user's computer and can often function offline.

**Web Application**

1. What are the advantages of using web applications over desktop applications?

* Web applications offer ubiquitous accessibility (any device, anywhere with internet), require no installation or manual updates, provide cross-platform compatibility, foster easier real-time collaboration, are generally lighter on local system resources, and can be more cost-effective to develop and maintain.

**Designing**

1. 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 user-friendly, visually appealing, and meets user needs, ultimately leading to increased user satisfaction, engagement, and success.

**Mobile Applications**

1. What are the differences between native and hybrid mobile apps?

* In a native app, your developers have to rewrite and redesign all the app functionality in the native development language. A hybrid app lets you write the app functionality in a single codebase. You can then wrap your code in a lightweight native app shell or container.

**DFD (Date Flow Diagram)**

1. What is the significance of DFD’s in system analysis?

* Data flow diagrams (DFDs) are essential in system analysis because they visually represent how data moves through a system, aiding in understanding, analysis, and communication.

**Desktop Application**

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

* Desktop applications generally offer better performance and offline functionality, but web applications are more accessible and easier to update.

**Flow Chart**

1. How do flow charts help in programming and system design?

* Flowcharts are visual diagrams that help in programming and system design by representing the step-by-step logic and flow of a process or algorithm.
* Clarity: A clear, graphical understanding of complex logic.
* Planning: Aid in planning the program's structure before coding begins.
* Communication: Serve as a common language for developers and stakeholders to discuss and understand the system.
* Debugging/Testing: Help identify logical errors and test cases by tracing the flow.
* Documentation: Provide easy-to-understand documentation of the system's operation.

**Module 15 – HTML in Full Stack**

**HTML Basics**

1. Define HTML. What is the purpose of HTML in web development?

* HTML (HyperText Markup Language) is the standard markup language used for creating web pages.
* Its purpose in web development is to define the structure and content of web pages. Think of it as the skeleton of a website, providing elements like headings, paragraphs, lists, links, images, and other multimedia, telling web browsers how to display information

1. Explain the basic structure of an HTML document. Identify the mandatory tags and their purposes.

* Basic Structure Example:

<!DOCTYPE html>

<html>

<head>

<title>My Web Page Title</title>

</head>

<body>

<h1>Welcome to my website!</h1>

</body>

</html>

* <!DOCTYPE html>: Tells the browser it's an HTML5 page.
* <html>: The root element containing all other HTML.
* <head>: Holds meta-information about the page (not visible content).
* <title>: Sets the title shown in the browser tab. (Mandatory for good practice/SEO)
* <body>: Contains all the visible content of the web page.

1. What is the difference between block-level elements and inline elements in HTML? Provide examples of each.

* Block-level Elements:
* Always start on a new line.
* By default, they take up the full available width of their parent container, even if their content doesn't fill that space.
* You can set their width, height, margin (all sides), and padding (all sides).
* Can contain both other block-level elements and inline elements.
* Examples:
* <div>: A generic container for grouping other elements.
* <p>: A paragraph of text.
* <h1> to <h6>: Headings of different levels.
* <ul>, <ol>, <li>: Lists and list items.
* <header>, <footer>, <nav>, <section>, <article>: Semantic elements for page structure.
* Inline Elements:
* Do not start on a new line; they flow horizontally within the current line of content.
* Only take as much width as necessary for their content.
* You cannot set their width or height directly (they are determined by their content).
* Only respect horizontal margin and padding (left and right); vertical margin and padding often have no visual effect or behave unpredictably.
* Can generally only contain data or other inline elements. They should not contain block-level elements.
* Examples:
* <a>: A hyperlink (link).
* <span>: A generic inline container for styling small sections of text.
* <strong> / <b>: For bold text.
* <img>: An image.
* <em> / <i>: For italicized/emphasized text.

1. Discuss the role of semantic HTML. Why is it important for accessibility and SEO? Provide examples of semantic elements.

* Semantic HTML enhances web page structure and meaning, improving both accessibility and SEO.
* This leads to better user experience and higher search engine rankings. Examples include <header>, <nav>, <article>, <aside>, <section>, and <footer>.
* Role of Semantic HTML: Semantic HTML involves using HTML elements that clearly describe the meaning of the content they contain, rather than just its presentation. For example, instead of using a <div> with a class like "navigation" to indicate a navigation section, a <nav> element is used.
* Importance for Accessibility:
* Semantic HTML is crucial for accessibility because it provides structure and meaning to web content, which is essential for assistive technologies like screen readers. Screen readers rely on semantic HTML to understand the organization of a page and present it to users with disabilities. By using semantic elements, developers ensure that screen readers can accurately convey the content, including headings, paragraphs, navigation, and other essential parts of the page.

* Importance for SEO:
* Semantic HTML also plays a vital role in SEO (Search Engine Optimization). Search engines use the semantic meaning of HTML elements to understand the structure and content of a webpage. By using appropriate semantic tags, websites can help search engines better index and rank their pages, making it easier for users to find the content through search queries. For instance, using <article> for blog posts and <header> for the top part of a page helps search engines understand the hierarchy and context of the content.
* Examples of Semantic Elements:
* <header>: Defines a header for a document or section.
* <nav>: Defines a navigation section.
* <article>: Defines an independent, self-contained piece of content.
* <aside>: Defines content that is tangentially related to the main content.
* <section>: Defines a section within a document.
* <footer>: Defines a footer for a document or section.
* <main>: Defines the main content of a document.
* <figure> and <figcaption>: Used for embedding images and providing captions.
* <h1> to <h6>: Headings that indicate the hierarchy of content on the page.
* <time>: Represents a specific period in time, such as a date or time.

**HTML Forms**

1. What are HTML forms used for? Describe the purpose of the input, textarea, select, and button elements.

* HTML forms collect user input and send it to a server for processing.
* <input>: Takes single-line input like text, numbers, emails.
* <textarea>: Takes multi-line text input.
* <select>: Creates a drop-down list of options.
* <button>: Submits the form or performs an action.

1. Explain the difference between the GET and POST methods in form submission. When should each be used?

* GET: Adds form data to the URL. Good for simple, safe requests like search or filters. Data is visible in the URL.
* POST: Sends data in the request body (not visible in URL). Better for sensitive info (like passwords) or large data (file uploads, forms).
* When to use:
* GET: When data doesn’t change server data (e.g., search).
* POST: When data changes or adds to server data (e.g., login, register, upload).

1. What is the purpose of the label element in a form, and how does it improve accessibility?

* Purpose:
* It describes what an input field is for by linking text to a form control.
* How it improves accessibility:
* It helps screen readers read the label with the input, and makes the text clickable to focus the input — easier for all users to interact with the form.

**HTML Tables**

1. Explain the structure of an HTML table and the purpose of each of the following elements: <table>, <tr>, <th>, <td>, and <thead>.

* <table>: Defines the table.
* <tr>: Table row, groups cells in a row.
* <th>: Table header cell, bold and centered by default.
* <td>: Table data cell, holds normal data.
* <thead>: Groups header rows, helps with styling and readability.

1. What is the difference between colspan and rowspan in tables? Provide examples. colspan:

* Merges cells across columns.

Example: <td colspan="2"> → spans 2 columns.

* rowspan: Merges cells across rows.

Example: <td rowspan="3"> → spans 3 rows.

1. Why should tables be used sparingly for layout purposes? What is a better alternative?

* Tables make layouts harder to maintain and less responsive. They also confuse screen readers.
* Better alternative: Use CSS with <div> and <section> for flexible, accessible layouts.