Module 1 – Overview of IT Industry-Theory Excercise

**1.What is a Program?**

Ans-> A **program** is a set of instructions written in a programming language that tells a computer what to do. It’s like a recipe that the computer follows step-by-step to perform a task or solve a prob

Ex. **Executed by a computer** to do things like display a webpage, play music, calculate numbers, run a game, etc.

**2. What is Programming?**

Ans->**Programming** is the process of writing instructions that a computer can follow to perform specific tasks. These instructions are written in a **programming language** like Python, JavaScript, C++, or Java.

->Programming is like giving a set of step-by-step instructions to a computer to tell it what to do.

**3.what is key steps in programming process?**

Ans -> Understanding the Problem.

-> Planning the Solution.

-> Writing the Code (Implementation).

-> Testing and Debugging.

-> Deployment.

-> Maintenance and Updates.

4. **Differences Between High-Level and Low-Level Languages?**

Ans->

| **Feature** | **High-Level Language** | **Low-Level Language** |
| --- | --- | --- |
| **Definition** | Human-friendly programming language | Hardware-friendly programming language |
| **Abstraction** | High – hides hardware details | Low – close to hardware |
| **Ease of Use** | Easy to learn and understand | Difficult to understand |
| **Portability** | Portable – can run on different systems | Not portable – hardware specific |
| **Execution Speed** | Slower due to abstraction | Faster due to direct machine access |
| **Control over Hardware** | Limited | Full control |
| **Examples** | Python, Java, C++, etc. | Assembly, Machine Code |

5.Roles of the Client and Server in Web Communication?

Ans->

Client->The **client** is typically a web browser or an application that initiates communication by sending requests to a server.

Server->The server is a powerful computer or program designed to provide services to other computers. Its main responsibilities include:

6.Function of the TCP/IP Model and Its Layers?

Ans->The TCP/IP model is a foundational framework for internet and network communication that outlines how data is transmitted between devices.

-> Application Layer

-> Transport Layer

-> Internet Layer

-> Network Access Layer

**7. Explain Client Server Communication?**

Ans-> Client-server communication is a fundamental model in computer networking where a client (a device or program requesting a service) sends a request to a server (a program or device that provides a service), and the server sends a response back to the client. This model is the basis for how most of the internet works, from Browse websites to sending emails.

**8.How does broadband differ from fiber-optic internet?**

Ans->

| **Feature** | **Broadband Internet** | **Fiber-Optic Internet** |
| --- | --- | --- |
| **Definition** | A general term for high-speed internet using various technologies like DSL, cable, or satellite. | A type of broadband that uses fiber-optic cables to transmit data as light. |
| **Technology** | Uses copper wires, coaxial cables, or satellite signals. | Uses glass/plastic fiber-optic cables. |
| **Speed** | Moderate to high (up to 1 Gbps in some cases). | Very high (1 Gbps to 10 Gbps or more). |
| **Upload/Download** | Usually different speeds (asymmetrical). | Usually same speeds (symmetrical). |
| **Latency** | Higher latency (slower response time). | Very low latency (fast response time). |
| **Signal Quality** | Can weaken over long distances or during peak usage. | Maintains strong signal over long distances. |
| **Reliability** | Can be affected by interference or weather (esp. satellite). | Very reliable and stable. |
| **Availability** | Widely available, including rural areas. | Less available, mostly in urban or developed areas. |
| **Best For** | Basic use like browsing, video streaming, emails. | Heavy use like HD streaming, online gaming, remote work. |

**9. What are the differences between HTTP and HTTPS protocols?**

Ans->

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Point |  | HTTP |  | HTTPS |
| • | Security | • | Data is sent without encryption. | • | Data is encrypted using SSL/TLS. |
| • | Privacy | • | Vulnerable to eavesdropping. | • | Protects against interception and tampering. |
| • | Trust | • | No padlock icon in browser address bar. | • | Shows padlock icon for secure connection. |

**10. role of encryption in securing applications**

Ans-> Encryption secures applications by converting readable data into unreadable form, ensuring:

-> Confidentiality – Only authorized users can read data.

-> Data in Transit Protection – Secures communication (e.g., HTTPS).

-> Data at Rest Protection – Safeguards stored data (e.g., database, backups).

-> Integrity & Authentication – Prevents tampering and verifies sources.

-> Compliance – Meets legal/security standards.

**11. system software vs application software**

Ans->

| **Aspect** | **System Software** | **Application Software** |
| --- | --- | --- |
| **Purpose** | Runs and manages computer hardware and system tasks | Helps users perform specific tasks |
| **User Interaction** | Works in the background | Used directly by the user |
| **Examples** | Operating System, Device Drivers, Utilities | MS Word, Web Browser, Media Player |
| **Installation** | Pre-installed or comes with the system | Installed by the user as needed |
| **Dependency** | Needed for the system to work | Needs system software to run |
| **Access to Hardware** | Direct access to hardware | Indirect, through system software |

**12.significance of modularity in software architecture**

Ans->

-> Easy to maintain.

-> Code reuse possible.

-> Easy to scale.

-> Simple testing.

-> Parallel team work.

-> Easy to understand.

-> More reliable system.

**13 important layer in software architecture?**

Ans->

-> Presentation Layer – User interface, jo user dekhte hain.

-> Business Logic Layer – Rules, calculations, and processing.

-> Data Access Layer – Database read/write handling.

-> Database Layer – Actual data storage.

**14. Importance of a Development Environment**

Ans->

-> Efficient coding – Provides tools, editors, and syntax highlighting.

-> Debugging support – Helps find and fix errors quickly.

-> Testing capabilities – Allows running and checking code before deployment.

-> Version control integration – Manages code history and teamwork.

-> Consistency – Keeps all developers on the same setup.

-> Productivity – Speeds up development with automation features.

-> Error prevention – Real-time error detection and suggestions.

**15.source code vs machine code**

Ans->

|  |  |  |
| --- | --- | --- |
| Point | Source Code | Machine Code |
| Format | Written in human-readable language. | Written in binary (0s and 1s). |
| Understandable By | Programmers. | Computer CPU. |
| Conversion | Needs compilation or interpretation. | Directly executed by CPU. |
| Example | printf("Hello"); | 10110000 01100001 |

**16. importance of version control**

Ans->

-> Track changes – Keeps history of all code modifications.

-> Collaboration – Multiple developers can work together without conflicts.

-> Backup & recovery – Restore previous versions if something breaks.

-> Conflict management – Handles simultaneous changes in code safely.

-> Accountability – Shows who made which changes and when.

-> Release management – Supports branching and merging for different versions.

-> Improved productivity – Reduces errors and streamlines development workflow.

**17. Benefits of Using GitHub for Students**

Ans->

-> Free hosting for projects – Students can host repositories at no cost.

-> Version control practice – Learn Git workflows and collaboration.

-> Portfolio building – Showcase projects to employers or colleges.

-> Collaboration skills – Work with peers on group projects.

-> Open-source contribution – Participate in real-world projects.

-> Backup of code – Keep all projects safe online.

-> Learning resources – Access tutorials, code examples, and community support.

**18.open source vs proprietary software**

Ans->

|  |  |  |
| --- | --- | --- |
| Point | Open-Source Software | Proprietary Software |
| Accessibility | Source code is freely available. | Source code is closed and restricted. |
| Cost | Usually free. | Usually paid. |
| Modification | Can be modified by anyone. | Cannot be modified without permission. |
| Examples | Linux, GIMP. | Windows, MS Office. |

**19. How Git Improves Collaboration**

Ans->

-> Multiple developers work simultaneously – Each can work on separate branches.

-> Version tracking – Every change is recorded with author and timestamp.

-> Merge management – Changes from different developers can be combined safely.

-> Conflict resolution – Detects and helps resolve code conflicts.

-> Code review support – Facilitates pull requests and peer reviews.

-> Backup & recovery – Team can revert to previous versions if needed.

-> Transparency – Everyone can see project history and progress.

**20. Role of Application Software in Businesses**

Ans->

-> Automates tasks – Reduces manual work and increases efficiency.

-> Improves productivity – Helps employees complete tasks faster.

-> Data management – Organizes, stores, and retrieves business data easily.

-> Communication – Supports email, messaging, and collaboration tools.

-> Decision making – Provides analytics and reporting for informed decisions.

-> Customer relationship management – Manages interactions with clients effectively.

-> Competitive advantage – Streamlines processes, reducing cost and improving service.

**21. Main Stages of the Software Development Process (SDLC)**

Ans->

-> Requirement Analysis – Understand and document what the users need.

-> Implementation / Coding – Write the actual program code.

-> System Design – Plan the architecture, modules, and data flow.

-> Testing – Check for errors, bugs, and ensure requirements are met.

-> Deployment – Install and release the software for use.

-> Maintenance – Update, fix, and improve the software over time.

**22. importance of requirement analysis**

Ans->

-> Clear understanding – Identifies exactly what the users need.

-> Reduces errors – Minimizes misunderstandings and mistakes in later stages.

-> Better planning – Helps design and development teams plan resources and timelines.

-> Cost-effective – Prevents costly changes during development.

-> Improves quality – Ensures the final software meets user expectations.

-> Facilitates communication – Bridges gap between stakeholders and developers.

-> Foundation for testing – Provides clear criteria to test the software against.

**23.Role of Software Analysis**

Ans->

-> Understanding requirements – Converts user needs into detailed functional specifications.

-> Identifying problems – Detects gaps, inconsistencies, and potential issues early.

-> Defining system scope – Determines what the software will and will not do.

-> Improves design – Provides a solid foundation for system architecture and design.

-> Risk management – Helps anticipate and mitigate potential project risks.

-> Resource planning – Guides allocation of time, budget, and manpower effectively.

**24.Key Elements of System Design**

Ans->

-> Architecture Design – Defines overall system structure and interaction between components.

-> Interface Design – Specifies how different modules and users interact with the system.

-> Data Design – Organizes how data is stored, accessed, and managed.

-> Component Design – Details individual modules and their functionality.

-> Database Design – Plans the database structure, relationships, and constraints.

-> Security Design – Ensures data and system protection against threats.

-> Performance Design – Optimizes system for speed, efficiency, and scalability.

**25 Importance of Software Testing**

Ans->

-> Detects bugs.

-> Ensures quality.

-> Improves reliability.

-> Saves cost.

-> Enhances user experience.

-> Verifies performance.

**26. Types of Software Maintenance**

Ans->

-> Corrective Maintenance – Fixing bugs and errors in the software.

-> Adaptive Maintenance – Updating software to work with new environments or platforms.

-> Perfective Maintenance – Improving performance, features, or usability.

-> Preventive Maintenance – Making changes to prevent future issues or failures.

**27.Web Application vs Desktop Application**

Ans->

|  |  |  |
| --- | --- | --- |
| Point | Web Applications | Desktop Applications |
| Location | Runs inside a web browser. | Installed locally on a device. |
| Point | Web Applications | Desktop Applications |
| Internet  Requirement | Needs internet to function. | Can work offline. |
| Updates | Updated on server, instantly available to all users. | Requires manual updates on each device. |
| Accessibility | Accessible from any device with a browser. | Accessible only on the installed device. |

**28.Advantages of web Application Over Desktop Application**

Ans->

-> Accessible anywhere.

-> No installation needed.

-> Works on multiple platforms.

-> Easy updates.

-> Cost-effective.

-> Supports collaboration.

-> Easily scalable.

**29.Role of UI/UX Design in Application Development.**

Ans->

|  |  |  |
| --- | --- | --- |
| Point | Native Mobile Apps | Hybrid Mobile Apps |
| Development | Built for a specific OS using platformspecific languages. | Built using web technologies for multiple OS. |
| Performance | High performance, optimized for platform. | Slightly lower due to compatibility layers. |
| Maintenance | Requires separate code for each OS. | One codebase works for multiple platforms. |
| Examples | Android apps in Kotlin/Java, iOS apps in Swift. | Apps built with Flutter, React Native. |

**30.Native vs Hybrid Mobile Apps**

Ans->

->Native Apps: Built for a specific platform (iOS or Android) using platform-specific languages; offer high performance, better UI/UX, full access to device features, but costlier and need separate development for each platform.

->Hybrid Apps: Built using web technologies (HTML, CSS, JS) wrapped in a native container; cost-effective, single codebase for multiple platforms, easier maintenance, but slightly lower performance and limited access to device features.

**31. Significance of DFDs in System Analysis**

Ans->

|  |  |  |
| --- | --- | --- |
| Point | Desktop Applications | Web Applications |
| Installation | Requires installation on the system. | No installation required. |
| Platform  Dependency | Usually works only on specific OS. | Works on multiple platforms via browser. |
| Performance | Often faster and can use full system resources. | May be slower, limited by browser capabilities. |
| Accessibility | Accessible only on installed device. | Accessible anywhere with internet. |

**32. Pros and Cons of Desktop Applications**

Ans->

|  |  |
| --- | --- |
| Pros | Cons |
| Works offline without internet. | Requires installation on each device. |
| Often faster and more responsive than web apps. | Platform-dependent, may not work on all OS. |
| Can utilize full system resources (CPU, GPU, storage). | Updating requires manual effort on each device. |
| Generally offers better performance for heavy tasks (e.g., video editing, gaming). | Not accessible remotely unless additional setup is done. |

**33. How Flowcharts Help in Programming and System Design**

Ans->

->Visualizes logic.

-> Simplifies complex processes.

-> Aids planning.

-> Helps detect errors.

-> Improves communication.

-> Acts as documentation.

-> Clarifies decision paths