📓 PYTHON – BACKEND ASSIGNMENT

MODULE 1 – OVERVIEW OF IT INDUSTRY

(THEORY ONLY – WITHOUT LAB EXERCISES)

---

1. What is a Program?

A **program** is a collection of instructions written in a programming language that tells a computer how to perform a specific task. A program accepts input, processes it, and produces output.

---

2. What is Programming?

Programming is the process of designing, writing, testing, and maintaining code that enables a computer to solve problems or perform tasks efficiently.

---

3. Key Steps in the Programming Process

1. Problem definition
2. Requirement analysis
3. Algorithm design
4. Coding
5. Testing and debugging
6. Deployment
7. Maintenance

---

4. Types of Programming Languages

Programming languages are mainly classified into:

- **High-level languages**: Easy to understand and user-friendly (Python, Java)
- **Low-level languages**: Close to hardware and difficult to understand (Assembly, Machine code)

---

5. Difference Between High-Level and Low-Level Languages

High-level languages are easy to read, write, and maintain, whereas low-level languages provide better performance and direct hardware control.

---

6. World Wide Web & How Internet Works

The World Wide Web is a system of interconnected web pages accessed through the internet. Data is transferred between client and server using standard protocols like HTTP.

---

7. Role of Client and Server

- **Client**: Sends requests for data or services
- **Server**: Processes requests and sends responses back to the client

---

8. Network Layers on Client and Server

Network communication follows layered models such as TCP/IP to ensure proper data transmission and error handling.

---

9. TCP/IP Model and Its Layers

The TCP/IP model consists of four layers:

1. Application Layer
2. Transport Layer
3. Internet Layer
4. Network Access Layer

Each layer has a specific responsibility in data communication.

---

10. Client-Server Communication

Client-server communication is a request-response model where the client initiates a request and the server responds after processing it.

---

11. Types of Internet Connections

Common internet connections include broadband, fiber-optic, and satellite connections, each differing in speed, reliability, and cost.

---

12. Difference Between Broadband and Fiber Internet

Broadband uses copper cables and offers moderate speed, while fiber-optic internet uses light signals and provides very high-speed connectivity.

---

13. Protocols

Protocols are rules that define how data is transmitted over a network. Examples include HTTP, HTTPS, and FTP.

---

14. Difference Between HTTP and HTTPS

HTTP transfers data without encryption, while HTTPS uses encryption to secure data during transmission.

---

15. Application Security

Application security involves protecting software from threats such as unauthorized access, data breaches, and cyberattacks.

---

16. Role of Encryption in Application Security

Encryption converts data into unreadable form to protect sensitive information from unauthorized users.

---

17. Software Applications and Its Types

Software is classified into:

- System software
- Application software
- Utility software

---

18. Difference Between System Software and Application Software

System software manages hardware and system resources, while application software helps users perform specific tasks.

---

19. Software Architecture

Software architecture defines the overall structure of a software system and how its components interact.

---

20. Importance of Modularity in Software Architecture

Modularity divides software into smaller components, making it easier to develop, test, and maintain.

---

21. Layers in Software Architecture

Layers such as presentation, business logic, and data access help separate responsibilities in a software system.

---

22. Importance of Layers in Software Architecture

Layered architecture improves scalability, security, and maintainability.

---

23. Software Environments

Software environments include development, testing, and production environments used during the software lifecycle.

---

24. Importance of Development Environment

A development environment allows programmers to write, test, and debug code safely before deployment.

---

25. Source Code

Source code is the human-readable form of a program written by developers.

---

26. Difference Between Source Code and Machine Code

Source code is readable by humans, while machine code is a binary format understood by computers.

---

27. GitHub and Version Control

GitHub is a platform used for version control and collaboration in software development.

---

28. Importance of Version Control

Version control tracks changes, helps collaboration, and prevents data loss.

---

29. Benefits of GitHub for Students

GitHub helps students learn collaboration, manage projects, and build professional portfolios.

---

30. Types of Software

Software can be categorized as system software, application software, and utility software.

---

31. Open-Source vs Proprietary Software

Open-source software is free and modifiable, while proprietary software is paid and restricted.

---

32. Role of Application Software in Business

Application software improves efficiency, automation, and productivity in organizations.

---

33. Software Development Life Cycle (SDLC)

SDLC is a structured process used to develop high-quality software systematically.

---

34. Importance of Requirement Analysis

Requirement analysis ensures that the software meets user needs and reduces development errors.

---

35. Role of Software Analysis

Software analysis evaluates system feasibility and functionality before development.

---

36. System Design

System design defines architecture, components, interfaces, and data flow.

---

37. Importance of Software Testing

Software testing ensures reliability, quality, and performance of applications.

---

38. Types of Software Maintenance

- Corrective
- Adaptive

- Perfective
- Preventive

---

39. Difference Between Web and Desktop Applications

Web applications run in browsers, while desktop applications run directly on a computer.

---

40. Advantages of Web Applications

Web applications are platform-independent, easy to update, and accessible anywhere.

---

41. Role of UI/UX Design

UI/UX design improves usability, user satisfaction, and application efficiency.

---

42. Native vs Hybrid Mobile Applications

Native apps are platform-specific and fast, while hybrid apps are cross-platform.

---

43. Importance of Data Flow Diagrams (DFD)

DFDs visually represent data movement within a system.

---

44. Desktop Applications: Pros and Cons

Desktop applications offer high performance but are platform-dependent.

---

45. Importance of Flowcharts

Flowcharts help in understanding program logic and improving problem-solving skills.