Module 1 – Overview of the IT Industry

LAB PRACTICE (Questions 1–32)

---

1. What is a Program?

Task: Write a “Hello World” program in two different languages (C and Python). Compare structure and syntax.

Python Code:

```python
print("Hello, World!")
```

C Code:

```c
#include <stdio.h>
int main() {
    printf("Hello, World!\n");
    return 0;
}
```

Diagram: Comparison Table

Feature	C	Python
Syntax	Verbose	Minimal
Semicolon	Required	Not required
Compilation	Required	Not required

---

2. What is Programming?

Task: Write pseudocode and flowchart for a program that calculates the average of 5 numbers.

Pseudocode:

```
Start
Input 5 numbers
```

```
Calculate sum and divide by 5
Print average
End
```

Diagram: Flowchart

```
[Start] → [Input numbers] → [Sum] → [Divide by 5] → [Output avera
ge] → [End]
```

---

3. Types of Programming Languages

Task: Create a classification chart of programming languages.

Diagram:

```
Programming Languages
|
|-- High-level (Python, Java)
|-- Low-level (Assembly)
|-- Scripting (PHP, JS)
|-- Object-Oriented (Java, C++)
```

---

4. World Wide Web & How Internet Works

Task: Diagram of how data transmits from client to server.

Diagram:

```
[Client Browser]
    |
    v
[DNS Lookup] → [Web Server]
    |
    v
[HTML Response Rendered in Browser]
```

---

5. Network Layers on Client and Server

Task: Design a simple HTTP client-server communication in Python.

Python Code:

```
from http.server import BaseHTTPRequestHandler, HTTPServer
class MyHandler(BaseHTTPRequestHandler):
    def do_GET(self):
        self.send_response(200)
        self.end_headers()
        self.wfile.write(b"Response from Server")
HTTPServer(("", 8080), MyHandler).serve_forever()
```

Diagram: TCP/IP Model

- Application Layer
- Transport Layer
- Network Layer
- Data Link Layer

---

6. Client and Servers

Task: Write client-server interaction diagram.

Diagram:

```
[Client App] ⇔ [Internet] ⇔ [Server App]
```

---

7. Types of Internet Connections

Task: Research and list types of connections with pros and cons.

Table:

Type	Pros	Cons
Broadband	Reliable, Available	Slower than fiber
Fiber	Very fast, Stable	Expensive
Satellite	Remote areas covered	Latency, Weather effect

---

8. Protocols

Task: Simulate HTTP and FTP using curl.

Command Line:

```
curl http://example.com
curl ftp://speedtest.tele2.net
```

---

9. Application Security

Task: Identify 3 vulnerabilities and give solutions.

Vulnerability	Solution
SQL Injection	Use Prepared Statements
XSS (Cross Site)	Sanitize User Input
CSRF	Token-based verification

Diagram: Web app security layers

---

10. Software Applications and Its Types

Task: Identify 5 apps and classify them.

Application	Type
MS Word	Application
Windows OS	System
WinRAR	Utility
Chrome	Application
Avast Antivirus	Utility

---

11. Software Architecture

Task: Design a 3-tier architecture diagram for a web application.

Diagram:

```
[Client Interface]
      |
[Application Server]
      |
[Database Layer]
```

---

12. Layers in Software Architecture

Task: Case study on presentation, business logic, and data access.

Example: Online Shopping App

- Presentation: HTML/CSS UI
- Logic: PHP or Python backend
- Data Layer: MySQL database

---

13. Software Environments

Task: Set up development environment in virtual machine (e.g. Ubuntu).

Steps:

1. Install VirtualBox
2. Install Ubuntu ISO
3. Setup Python and VS Code

---

14. Source Code

Task: Write a program and upload source code to GitHub.

Steps:

```bash
git init
git add .
git commit -m "Initial commit"
git remote add origin <repo-url>
git push -u origin main
```

---

15. GitHub and Introductions

Task: Create repo and document push steps.

Commands:

```
git clone <repo-url>
cd project-folder
git checkout -b dev
```

---

16. Student Account in GitHub

Task: Create account and collaborate on project.

Steps:

- Register at GitHub.com
- Create repository
- Invite collaborator by username/email

---

17. Types of Software

Task: Classify software used.

Software	Category
Windows OS	System
MS Word	Application
Chrome	Application
WinRAR	Utility
Ubuntu	System

---

18. Git and GitHub Training

Task: Practice Git: clone, branch, merge.

```
git clone <url>
git checkout -b new-feature
# make changes
```

```
git add .
git commit -m "feature added"
git merge new-feature
```

---

19. Application Software

Task: Report on software used to improve productivity.

Examples:

- MS Word → Documents
- Excel → Data Analysis
- Zoom → Virtual Meetings
- Canva → Graphic Design

---

20. Software Development Process

Task: Flowchart of SDLC.

Diagram:

```
[Requirements] → [Design] → [Development] → [Testing] → [Deployment] → [Maintenance]
```

---

21. Software Requirement

Task: Write requirement specification for library system.

Example Specs:

- Add/Search/Issue Books
- User Roles: Admin, Student
- Notifications for overdue books

---

22. Software Analysis

Task: Perform functional analysis of online shopping.

Functions:

- Browse products
- Cart & checkout
- Online payment
- Order tracking

---

23. System Design

Task: Design architecture for food delivery app.

Diagram:

```
[User App] → [Restaurant Panel] → [Delivery Panel] → [Admin Dashboard]
```

---

24. Software Testing

Task: Create test cases for calculator app.

Test Case	Input	Expected Output
Addition	2 + 3	5
Division by zero	4 / 0	Error

---

25. Maintenance

Task: Real-world example of software maintenance.

Example: Banking app update for two-factor authentication to meet new compliance.

---

26. Development

Task: Compare web and desktop applications.

Feature	Web App	Desktop App
Accessibility	Anywhere	Local only

Feature	Web App	Desktop App
Update	Server-side	Manual update

---

27. Web Application

Task: Explain advantages of web apps.

Benefits:

- Platform-independent
- Real-time updates
- Accessible from any device

---

28. Designing

Task: Explain role of UI/UX in development.

Explanation: Good UI/UX improves user satisfaction, usability, and product success.

---

29. Mobile Application

Task: Differences between native and hybrid apps.

Feature	Native App	Hybrid App
Platform	Specific	Cross-platform
Performance	High	Medium

---

30. DFD (Data Flow Diagram)

Task: DFD for hospital system.

DFD Example:

```
[Patient] → [Reception] → [Doctor] → [Lab/Pharmacy]
```

---

31. Desktop Application

Task: Build simple calculator using GUI.

Python (Tkinter):

```
from tkinter import *
root = Tk()
root.title("Calculator")
Entry(root).pack()
Button(root, text="Add").pack()
root.mainloop()
```

---

32. Flow Chart

Task: Draw flowchart for registration system.

Flowchart:

```
[Start] → [Input Data] → [Validate] → [Store] → [Success] → [End]
```
