Module 1 – Overview of IT Industry

THEORY EXERCISE

What is a Program?

- A **program** is a set of instructions that tells a computer what to do. It's like a recipe that guides the computer to perform tasks, such as adding numbers, showing a picture, or playing a video. Programs are written in special languages like Python or Java, and once created, the computer follows these instructions to do the job you want.

Programs can be:

1. **Executable**: Directly run by the operating system (like a software application).
2. **Scripts**: Shorter programs typically written to automate tasks (often executed by an interpreter).
3. **Systems**: Larger, more complex programs that manage and control computer hardware, operating systems, or networks.

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

- A **program** is a collection of instructions that a computer follows to perform specific tasks. It's like a set of directions telling the computer how to solve a problem or complete a job, such as displaying a webpage, calculating numbers, or running a game.

Here's how it functions:

1. Writing: A programmer writes the program using a programming language (like Python, Java, or C++).
2. Input: The program may take input, like data from a user or a file.
3. **Processing**: The computer follows the instructions in the program to process the input.
4. **Output**: The program gives results or performs actions based on the instructions, like showing information on a screen or saving data.

What is Programming?

- **Programming** is the process of writing instructions that a computer can follow to perform specific tasks. These instructions, called **code**, are written in a programming language, which is a language designed to communicate with computers.

When you write a program, you are telling the computer exactly what to do, step by step. This can involve tasks like performing calculations, storing information, creating websites, running applications, or even controlling robots.

In simple terms, programming is the act of creating software or applications by writing commands that computers can understand and execute. Some popular programming languages include **Python**, **Java**, **C++**, and **JavaScript**.

What are the main differences between high-level and low-level programming languages?

- The main differences between **high-level** and **low-level** programming languages are:

1. **Abstraction**:
   - **High-level languages** are more abstract and closer to human languages. They are easier to read and write because they use English-like syntax. Examples include Python, Java, and C++.
   - **Low-level languages** are closer to machine language (binary code) and are harder to read and write. They provide less abstraction and give more control over hardware. Examples include Assembly language and machine code.
2. **Ease of Use**:
   - **High-level languages** are designed to be user-friendly, making programming faster and easier. You don’t need to worry about hardware details.
   - **Low-level languages** require a deeper understanding of how the computer works. Programmers need to manage things like memory and CPU registers directly.
3. **Control**:
   - **Low-level languages** provide more control over the hardware, allowing precise manipulation of system resources (like memory).
   - **High-level languages** abstract away most hardware details, making it harder to access and control the system directly but easier to focus on the logic of the program.
4. **Portability**:
   - **High-level languages** are generally portable, meaning code written in one system can be easily run on another system with minimal changes.

   - **Low-level languages** are less portable because they are closely tied to a specific machine’s architecture.

5. **Speed**:
   - **Low-level languages** can produce faster and more efficient programs because they are closer to machine code.
   - **High-level languages** may be slower because they add extra layers of abstraction and require translation to machine code, often through compilers or interpreters.

World Wide Web & How Internet Works

World Wide Web (WWW):

The **World Wide Web** (often just called the web) is a system of websites and webpages that are connected to each other through links. It is a way of accessing information over the Internet using browsers like Google Chrome, Safari, or Firefox. The web allows you to browse websites, view multimedia content, and interact with various online services.

In simple terms, the WWW is the collection of websites and resources that you can access on the Internet using your browser.

How the Internet Works:

The **Internet** is a global network of computers connected to each other, allowing them to communicate and share information. Here's how it works in a simple way:

1. **Devices Connect**: Devices like computers, smartphones, and tablets connect to the Internet via wired connections (like Ethernet cables) or wireless connections (like Wi-Fi).

2. **Data Transmission**: When you send or receive information, the data is broken into smaller packets and travels across various network routes. These packets are sent through a complex system of routers and servers that help direct the data to the right destination.
3. **IP Addresses and DNS**:
    - Every device on the Internet has a unique **IP address** (like a home address for the device).
    - When you type a website address (like www.example.com), the **DNS (Domain Name System)** translates the human-readable domain name into an IP address so the computer knows where to send the request.
4. **Servers and Hosting**:
    - Websites are hosted on **servers**, which are powerful computers that store website files.
    - When you request a webpage, your browser contacts the server where the website is hosted, retrieves the data, and displays it on your screen.
5. **Protocols**:
    - The Internet uses specific rules or **protocols** to ensure smooth communication. The most common ones are:
        - **HTTP/HTTPS**: Used for transferring web pages (Hypertext Transfer Protocol).
        - **TCP/IP**: A set of protocols for reliable communication over the Internet.

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

The **TCP/IP model** is a set of rules that help computers and devices communicate over the Internet. It breaks down communication into **4 layers**, each with a specific job to make sure data gets from one place to another correctly.

Here's a simple breakdown of the **4 layers**:

1. Application Layer:

- **What it does**: This is where you interact with the Internet. When you use apps like a web browser or email, the application layer sends and receives your data.
- **Example**: Browsing a website (uses **HTTP**).

2. Transport Layer:

- **What it does**: It makes sure the data is delivered properly and in the right order.
- **Example**: **TCP** checks for errors and makes sure the data gets to the right place, while **UDP** is faster but doesn't check for errors.

3. Internet Layer:

- **What it does**: This layer is responsible for finding the best path to send the data to the correct destination, using an address (called an **IP address**).
- **Example**: It uses **IP** to find the destination device and send the data there.

4. Network Access Layer:

- **What it does**: This layer deals with the physical part of the connection, like cables or Wi-Fi, to actually send the data from one device to another.
- **Example**: Wi-Fi or Ethernet connections.

Types of Internet Connections

There are several types of **Internet connections**, and each one works a bit differently. Here's a simple breakdown:

1. Dial-Up Connection:

- **How it works**: Uses a phone line to connect to the Internet.
- **Speed**: Very slow (usually up to 56 Kbps).
- **Common use**: Rare today, but used in the past for basic web browsing and email.

2. DSL (Digital Subscriber Line):

- **How it works**: Uses a phone line but offers faster speeds than dial-up.
- **Speed**: Moderate (up to 100 Mbps, depending on the type of DSL).
- **Common use**: Home internet for browsing, streaming, and video calls.

3. Cable Internet:

- **How it works**: Uses cable TV lines to provide internet.
- **Speed**: Faster than DSL (up to 1 Gbps in some areas).
- **Common use**: Popular for home use, great for streaming and gaming.

4. Fiber-Optic Internet:

- **How it works**: Uses light signals traveling through fiber-optic cables for very fast internet.
- **Speed**: Very fast (up to 1 Gbps or more).
- **Common use**: Best for high-speed needs like streaming in HD or 4K, online gaming, and large file downloads.

5. Satellite Internet:

- **How it works**: Uses satellites to beam internet signals to and from your device.
- **Speed**: Moderate to slow (depends on the provider, usually 10-100 Mbps).
- **Common use**: Often used in rural or remote areas where other types of connections are not available.

6. Mobile Hotspot (4G/5G):

- **How it works**: Uses cellular networks (like 4G or 5G) to provide internet on the go.
- **Speed**: Fast (4G up to 100 Mbps, 5G much faster).
- **Common use**: Portable internet for when you're out and about or don't have a fixed internet connection.

7. Wireless (Wi-Fi):

- **How it works**: A wireless router connects to your internet service (like DSL, cable, or fiber) and lets you access the internet on your devices via Wi-Fi.
- **Speed**: Depends on the connection type (DSL, cable, fiber, etc.).
- **Common use**: Common for home and office internet.

Protocols

Protocols are like rules or instructions that help computers and devices communicate with each other over the Internet. They ensure that data is sent, received, and understood properly between devices, no matter what type of devices they are. Without these rules, devices wouldn't be able to understand each other.

Here are some common **protocols** explained simply:

1. **HTTP (Hypertext Transfer Protocol):**
   - **What it does**: This protocol is used for browsing websites. When you open a website in your browser, HTTP is used to request and send data (like the webpage you're viewing).
   - **Example**: When you type "www.example.com" in your browser, HTTP is used to get the page.
2. **HTTPS (Hypertext Transfer Protocol Secure):**
   - **What it does**: This is like HTTP, but with added security. It encrypts the data to keep your information safe when you're browsing websites (especially for things like online shopping or banking).
   - **Example**: Websites with "https://" in their URL use this protocol to keep your data private.
3. **FTP (File Transfer Protocol):**
   - **What it does**: FTP is used to transfer files between computers over the Internet.
   - **Example**: If you’re uploading a picture to a website or downloading a file from a server, FTP is the protocol being used.
   -
4. **IP (Internet Protocol):**
   - **What it does**: IP helps find the location of devices on the Internet and routes the data to the correct address.
   - **Example**: When you send a message or visit a website, IP makes sure it gets to the right destination.
5. **TCP (Transmission Control Protocol):**
   - **What it does**: TCP makes sure the data sent from one device to another is reliable and complete.

- **Example**: If you’re streaming a video, TCP ensures the video data arrives in the correct order and without errors.

What are the differences between HTTP and HTTPS protocols?

The main differences between **HTTP** and **HTTPS** are related to **security** and **data protection**. Here’s a simple breakdown:

1. Security:

- **HTTP**: Stands for **Hypertext Transfer Protocol**. It does **not encrypt** data. This means information (like passwords or credit card details) can be intercepted by hackers when transmitted over the internet.
- **HTTPS**: Stands for **Hypertext Transfer Protocol Secure**. It uses **SSL/TLS encryption** to secure the data being sent between your browser and the website. This makes it much harder for anyone to intercept or tamper with your data.

2. URL:

- **HTTP**: The website URL starts with **"http://"** (without the "S").
- **HTTPS**: The website URL starts with **"https://"** (with the "S" for secure).

3. Data Protection:

- **HTTP**: Data is **not encrypted**, so it can be seen by anyone who has access to the network.
- **HTTPS**: Data is **encrypted**, providing a secure connection to protect user privacy, especially when transmitting sensitive information.

4. Trustworthiness:

- **HTTP**: Websites using HTTP are **not verified** as secure. Users may see a warning from their browser when entering personal details.
- **HTTPS**: Websites using HTTPS are **trusted and verified** by certificate authorities (CAs). A small **padlock icon** appears next to the URL in the browser to show the connection is secure.

5. SEO Ranking:

- **HTTP**: Websites using HTTP may be ranked lower by search engines like Google.
- **HTTPS**: Websites with HTTPS are given a **ranking boost** by search engines because they are considered more secure.

Github and Introductions :

What is GitHub?

- GitHub is a web-based platform that provides version control using Git. It allows developers to collaborate on software projects, track changes to code, and manage repositories. It hosts both public and private repositories and provides tools for bug tracking, project management, and code review.

GIT Command :

- first time use

```
git config --global user.email "your-email@example.com"

git config --global user.name "Your Name"
```

> new respositry

```
1) git init
2) git add .
3) git status
4) git commit -m "first commit"
5) git branch -M main
6) git remote add origin https://github.com/harsh022006/xyz.git
7) git push -u origin main
```

> new file

```
1) git add .
2) git status
3) git commit -m "file name"
4) git push -u origin main
```