Problem1:

1. Compare Advantages and Disadvantages of IoT.

Advantages	Disadvantages
• Saves a lot of time because of automatic activities • It minimizes human effort because IoT devices connect and communicate with each other and tasks without human interact • Assists in smart homes and cities via mobile phone or any other devices that is connected to internet • Improved data collection and analysis because IoT devices generate vast amounts of data that can be analyzed for insights and decision-making.	• there is a risk that our information as it can be misused because hacker may access to system and gain personal information since a lot of devices are connected to internet • the complexity of systems, there are many ways for them to fail • It relies heavily on the Internet and cannot work efficiently without it

2. Define what's the Gateway and explains its role in IoT layers.

In the context of IoT, a gateway is a device or software application that serves as a bridge between IoT devices and the broader internet or cloud services.

The IoT architecture is typically divided into several layers, each serving specific functions to ensure efficient operation and management of IoT systems

For example:

Data Preprocessing: IoT devices can collect a massive amount of data, which requires a significant amount of bandwidth to send to cloud-based applications for processing. An IoT gateway performs data filtering and pre-processing to reduce the volume of data sent to the cloud.

3. Compare fog computing to cloud computing, then comment what is better for the IoT Application?

Feature	Fog Computing	Cloud Computing
Architecture	Decentralized	Centralized
Latency	Low	High
Data Processing	At the edge of the network, near data source	At centralized data centers
Bandwidth Usage	Lower, as data is processed locally	Higher, as all data is sent to the cloud
Response Time	Faster, real-time processing	Slower due to network delays
Scalability	Moderate	High
Deployment	Closer to the end devices	In remote data centers
Reliability	High, as it relies on multiple edge nodes	Depends on internet connectivity

fog computing is generally better for real-time processing and applications where immediate data analysis is critical.

Cloud computing, while highly scalable and resource-rich, may introduce higher latency and bandwidth usage, making it less optimal for time-sensitive IoT applications. However, a hybrid approach leveraging both fog and cloud computing can often provide the best of both worlds, combining the low latency and local processing of fog with the scalability and resource availability of the cloud.

4. State the difference between HTTP and MQTT in IoT systems and which is preferred in most cases?

HTTP (Hypertext Transfer Protocol) is a protocol used for transmitting hypertext (web pages) over the internet. It is the foundation of data communication on the World Wide Web

MQTT (Message Queuing Telemetry Transport) is a lightweight messaging protocol designed for constrained devices and low-bandwidth, high-latency, or unreliable networks

What Are the Main Differences?

HTTP is typically a transient interface in which each request is a short-lived session. MQTT sessions are long-lived.

Another important difference is that HTTP operates on a command-response basis. A command gets sent to the server and a response returns. The endpoint, browser or small-footprint device must initiate that transaction to the server every time.

However, MQTT uses a concept known as publish-subscribe. The messages can be initiated at either end. It also utilizes a broker (or an aggregator and a distributor of messages) that allows the client device to speak to thousands of endpoints instead of just one.

HTTP vs. MQTT for IoT Use Cases

For a cloud platform, IoT devices are typically going to communicate over MQTT because its bidirectional. Still, the other side of the cloud will need HTTPS to let users pull data for dashboarding and analytics. In essence, most of the interactions on the back end for data visualization are over HTTP. However, MQTT is better in the front end, where the transactional commands and raw data come in and out.

Picture a smart connected doorbell with a video camera: The device will activate and initiate a connection over HTTP when someone steps into view. Yet the video will establish an MQTT stream to move that data rather than pumping that information individually over HTTP messages.

5. Mention the three main kinds of program errors and compare them to each others.

1. Syntax Errors: Related to the structure and syntax of the code, detected at compile-time, easy to fix.
    - Common Causes: Typographical mistakes, missing symbols
    - Example: Missing semicolon, incorrect indentation

2. Runtime Errors: Occur during execution due to illegal operations or resource issues, can be complex to fix.
    - Common Causes: Illegal operations, resource limitations
    - Example: Division by zero, file not found

3. Logical Errors: Flaws in the program's logic, leading to incorrect output, can be difficult to identify and fix.
    - Common Causes: Incorrect algorithms, faulty logic
    - Example: Incorrect calculation, flawed condition

4. State difference between source code and object code.

Source code is the human-readable set of instructions written by a programmer using a high-level programming language (e.g., Python, Java, C++).
Object code is the machine-readable set of instructions generated by a compiler or assembler from the source code. It is usually in binary form.

Aspect	Source Code	Object Code
Readability	Human-readable	Machine-readable (binary)
Editability	Easily editable by programmers	Not easily editable
Creation	Written by programmers	Generated by compilers/assemblers
Language	High-level programming languages	Low-level machine code
Usage	Used for writing and maintaining programs	Used for execution by the computer's CPU
Form	Text-based	Binary or hexadecimal
Dependency	Requires a compiler/interpreter to execute	May require linking to form executable code

5. What is enum data type is used for?

Enumeration (or enum) is a user defined data type in C. It is mainly used to assign names to integral constants, the names make a program easy to read and maintain.