Mini-project: C++ RAII (Resource Acquisition Is Initialization)

1. Project Goal:

In this mini-project, you will build a class that manages a resource (e.g., a file or socket) and automatically releases it when the object is destroyed. This helps you better understand C++ resource management, which is important for avoiding issues such as memory leaks.

2. Key Concepts:

1. Resource Acquisition Is Initialization (RAII):

- RAII is a C++ programming idiom where resource management (such as opening files or allocating memory) is tied to object lifetime. When an object acquires a resource (e.g., opening a file), the resource is released when the object's destructor is called (e.g., file is closed).
- This ensures that resources are automatically and safely cleaned up when the object goes out of scope, preventing resource leaks (e.g., memory leaks, file handles left open).

2. Destructors:

- In RAII, destructors play a critical role. When an object is destroyed (either because it goes out of scope or is explicitly deleted), the destructor is automatically called. This is where you release the resource (e.g., close a file, free memory).
- Destructors ensure resources are cleaned up, even if an exception is thrown.

3. Project Structure and Steps:

Step 1: Choose a resource to manage in your class. Examples include file handles, memory

allocations, or socket connections.

Step 2: Implement the constructor to acquire the resource when the object is created. For example, in the case of a file, open the file in the constructor.

Step 3: Implement the destructor to release the resource when the object is destroyed. For example, close the file in the destructor.

Step 4: Ensure that the class cannot be easily copied or moved in a way that would lead to resource issues (e.g., use the `delete` keyword for the copy constructor and assignment operator).

Step 5: Test the class to ensure that resources are acquired and released properly when objects go in and out of scope, especially in the presence of exceptions.