**Exercise 11: Implementing Dependency Injection**

**Scenario:**

You are developing a customer management application where the service class depends on a repository class. Use Dependency Injection to manage these dependencies.

**Steps:**

1. **Create a New Java Project:**
   * Create a new Java project named **DependencyInjectionExample**.
2. **Define Repository Interface:**
   * Create an interface **CustomerRepository** with methods like **findCustomerById()**.
3. **Implement Concrete Repository:**
   * Create a class **CustomerRepositoryImpl** that implements **CustomerRepository**.
4. **Define Service Class:**
   * Create a class **CustomerService** that depends on **CustomerRepository**.
5. **Implement Dependency Injection:**
   * Use constructor injection to inject **CustomerRepository** into **CustomerService**.
6. **Test the Dependency Injection Implementation:**
   * Create a main class to demonstrate creating a **CustomerService** with **CustomerRepositoryImpl** and using it to find a customer.

using System;

public interface ICustomerRepository

{

string FindCustomerById(int id);

}

public class CustomerRepositoryImpl : ICustomerRepository

{

public string FindCustomerById(int id)

{

return $"Customer ID: {id}, Name: Customer{id}";

}

}

public class CustomerService

{

private readonly ICustomerRepository \_repository;

public CustomerService(ICustomerRepository repository)

{

\_repository = repository;

}

public void DisplayCustomer(int id)

{

string customer = \_repository.FindCustomerById(id);

Console.WriteLine(customer);

}

}

public class DependencyInjectionExample

{

public static void Main(string[] args)

{

ICustomerRepository repository = new CustomerRepositoryImpl();

CustomerService service = new CustomerService(repository);

service.DisplayCustomer(101);

}

}

