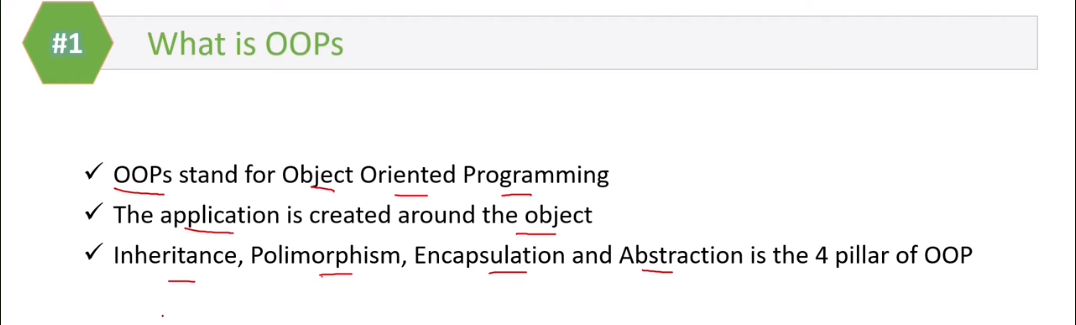
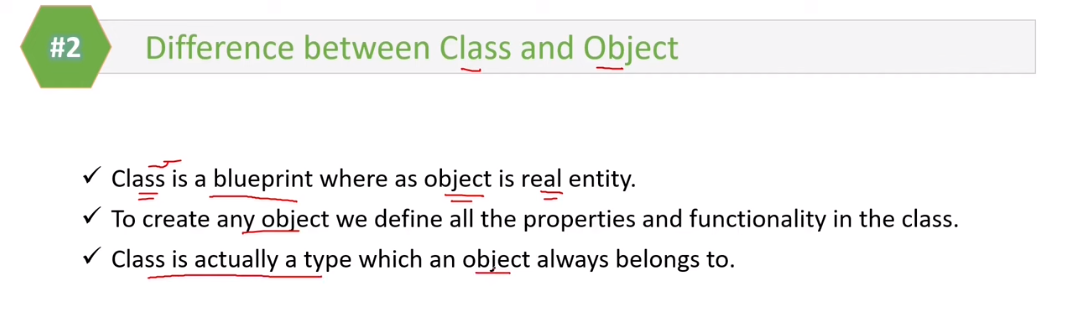
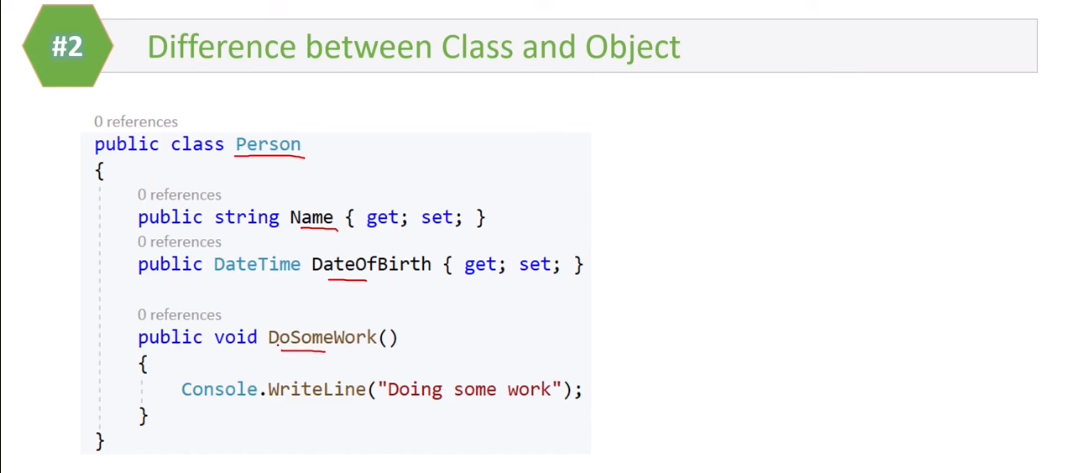
1. **What is Dynamic binding?**

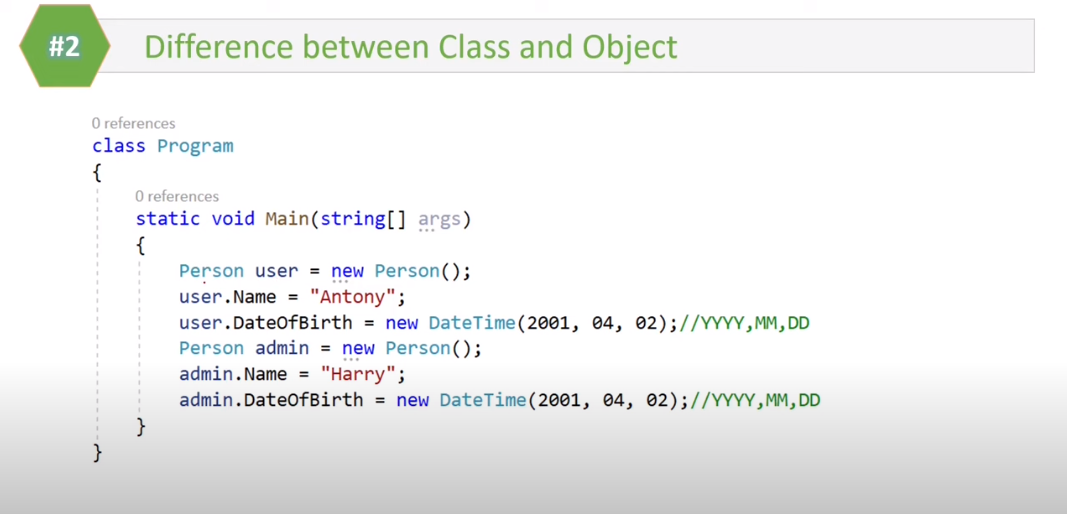
Dynamic binding in C# is a **process that delays binding until runtime, instead of compile time.** Binding is the process of resolving types, members, and operations. Dynamic binding is also known as late binding or runtime binding.

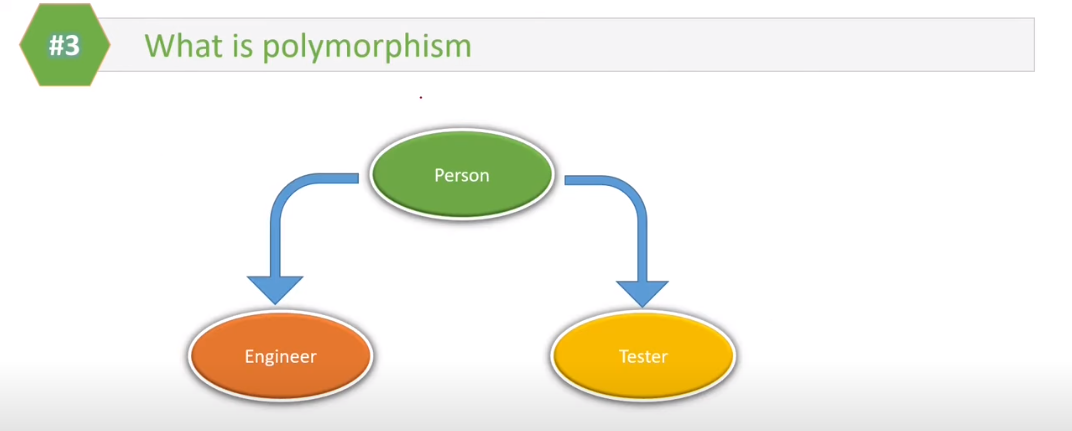


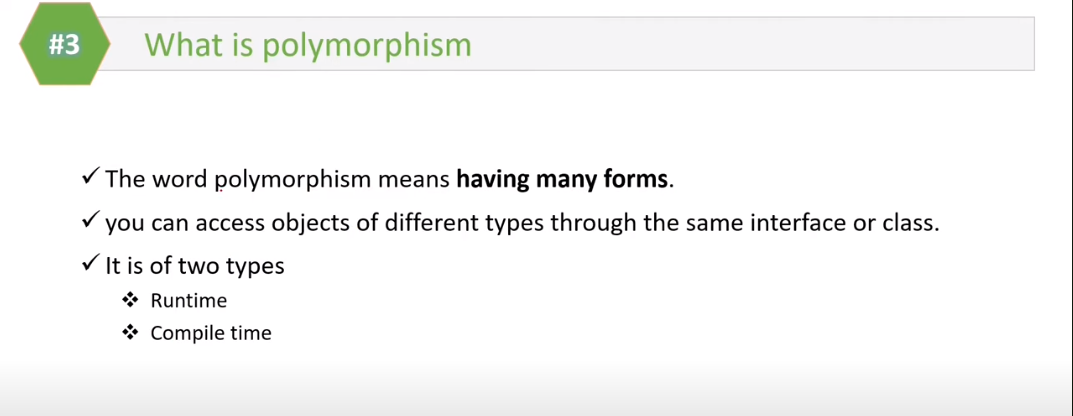


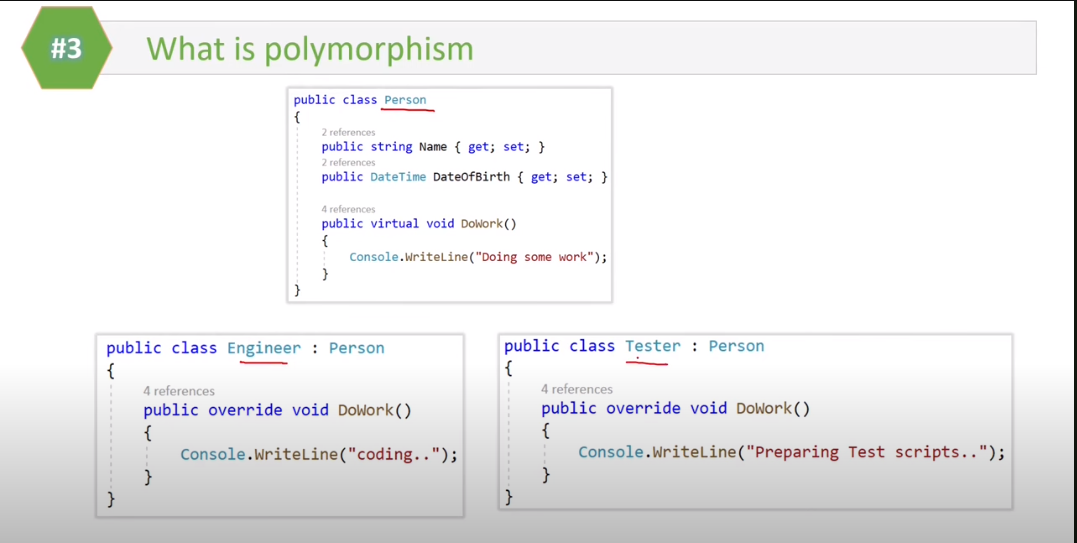


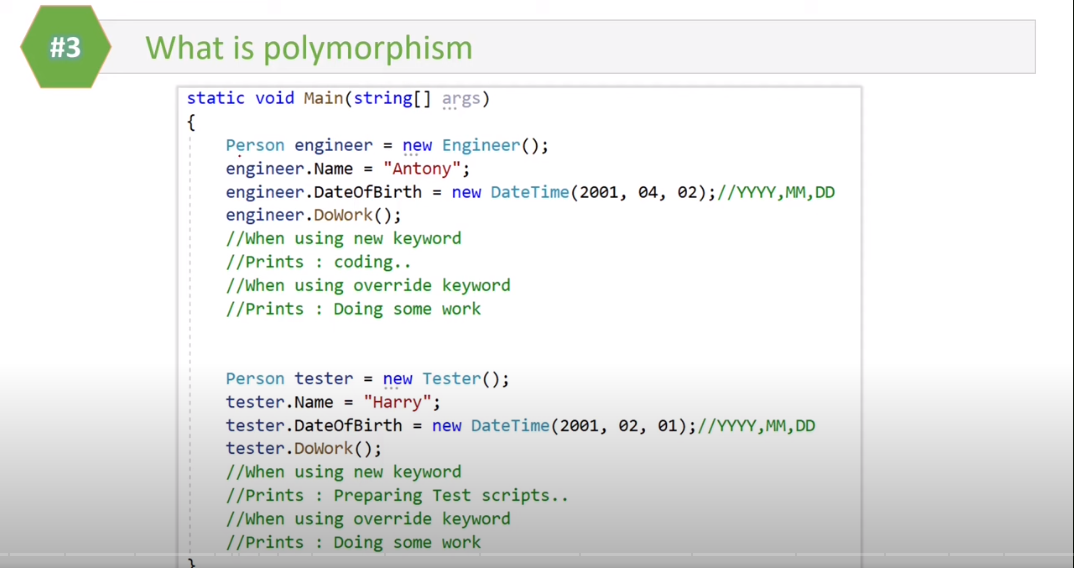


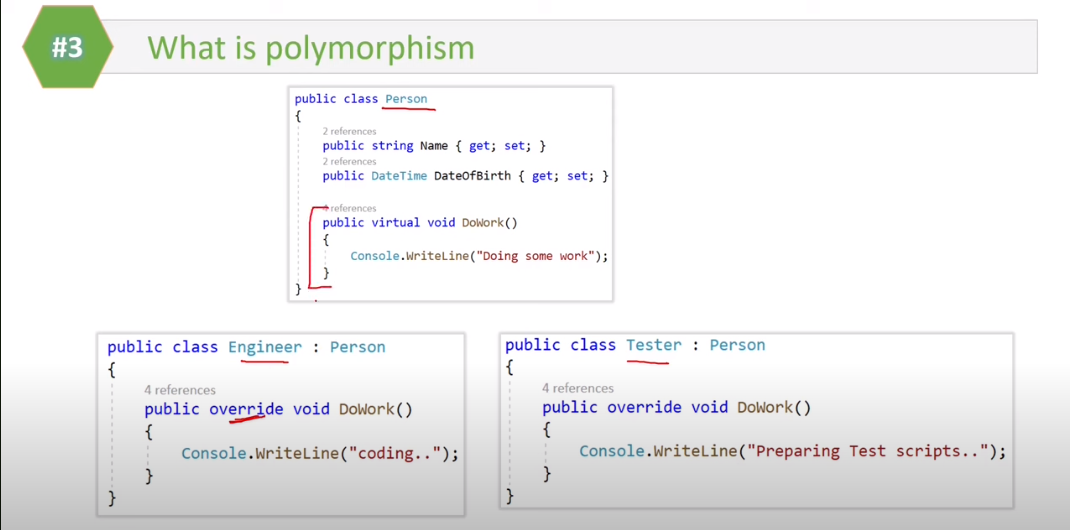


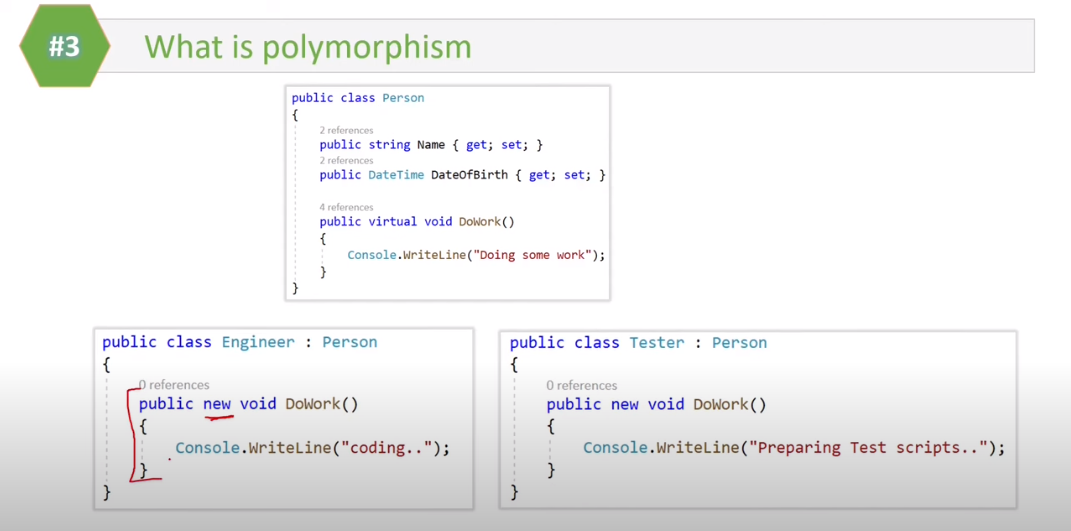


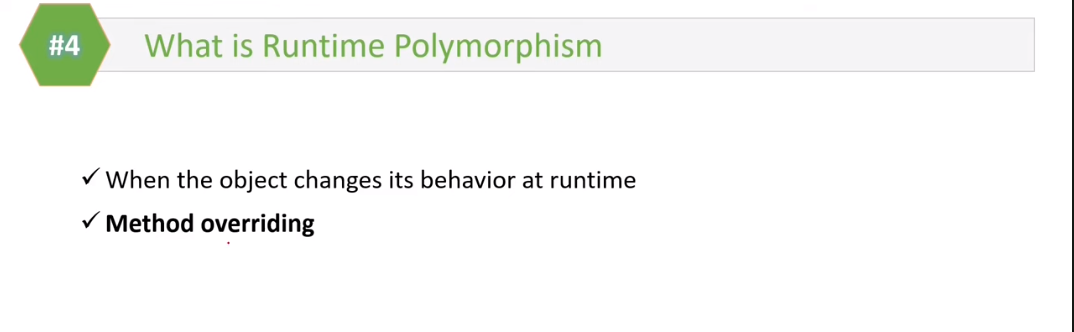


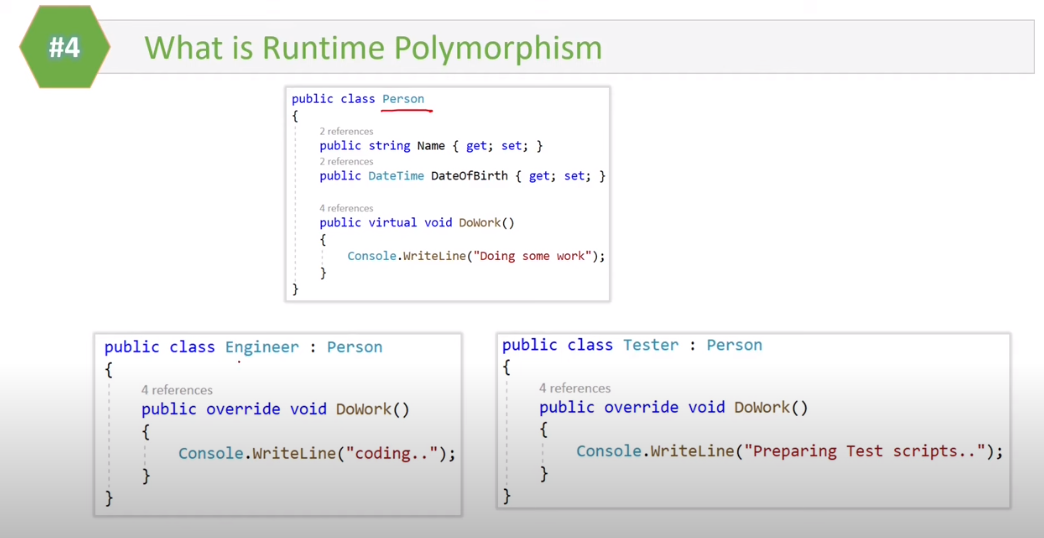


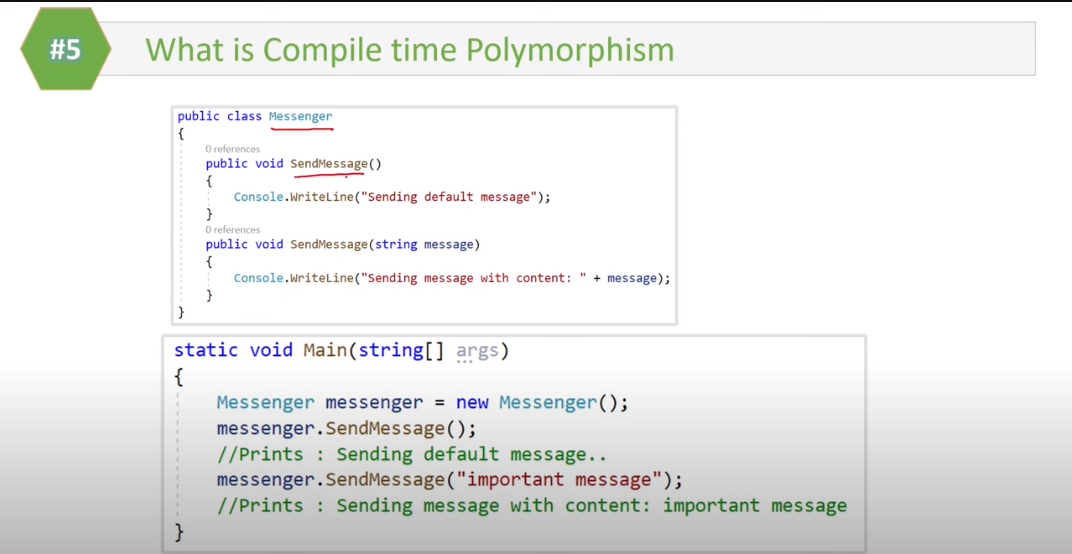


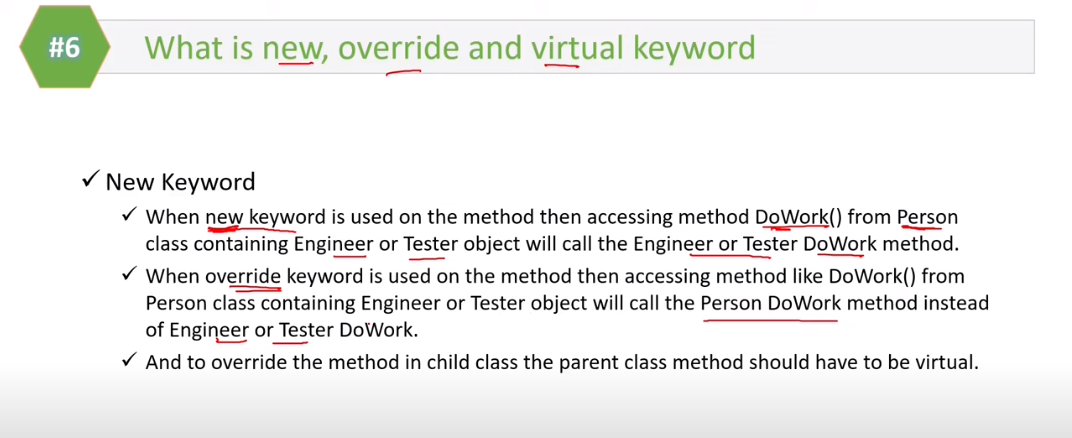


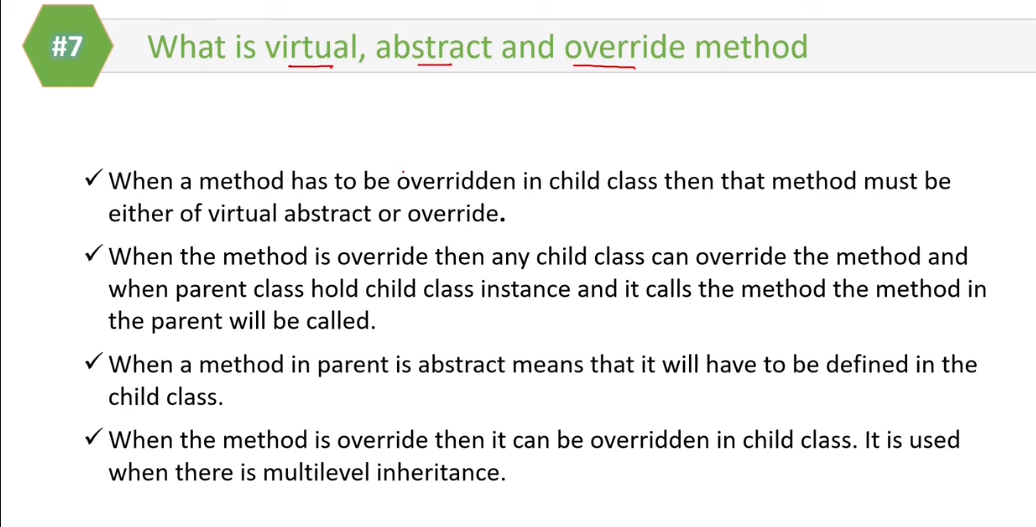


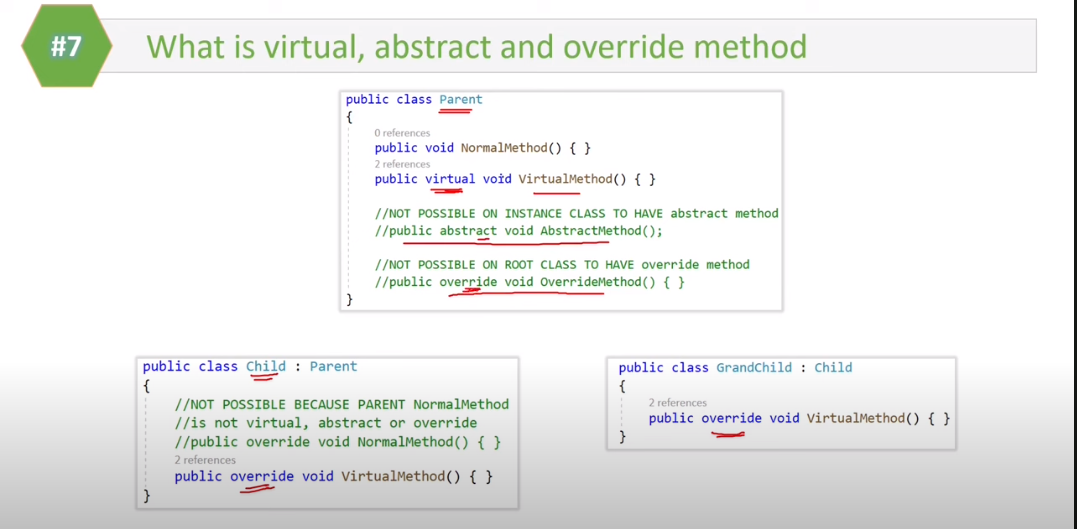


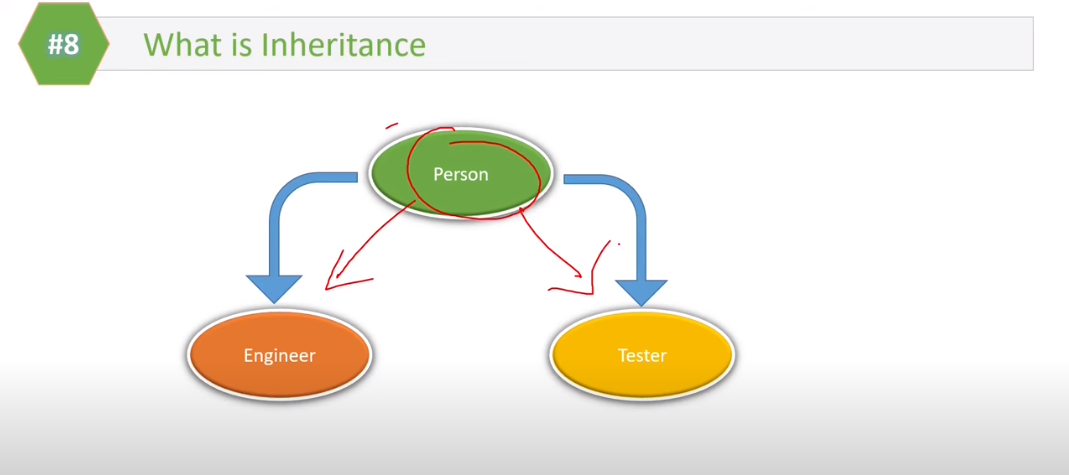


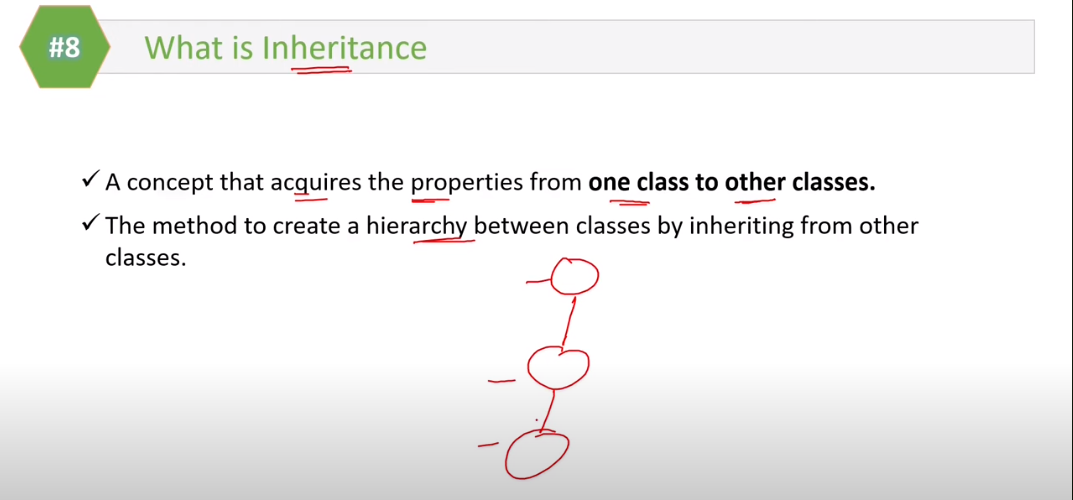


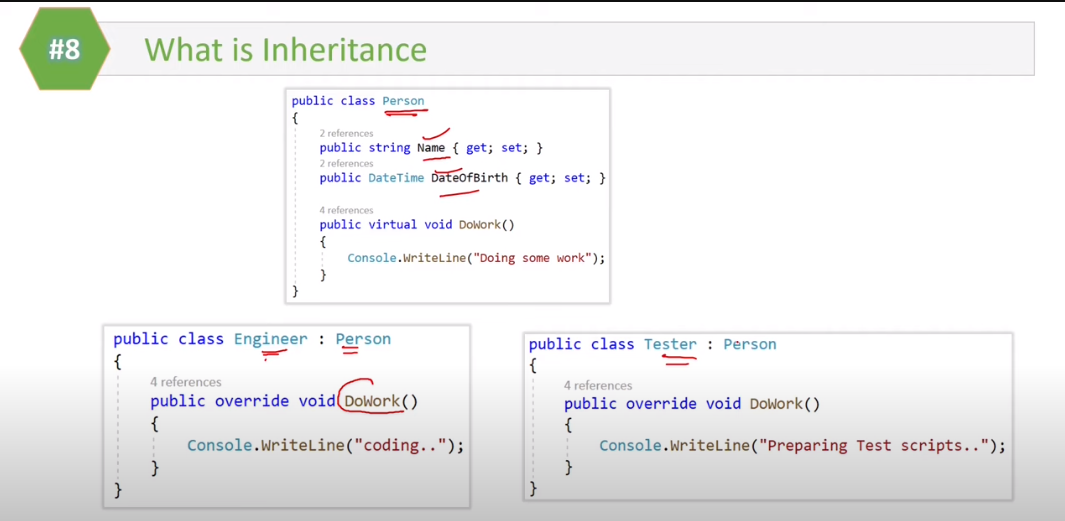


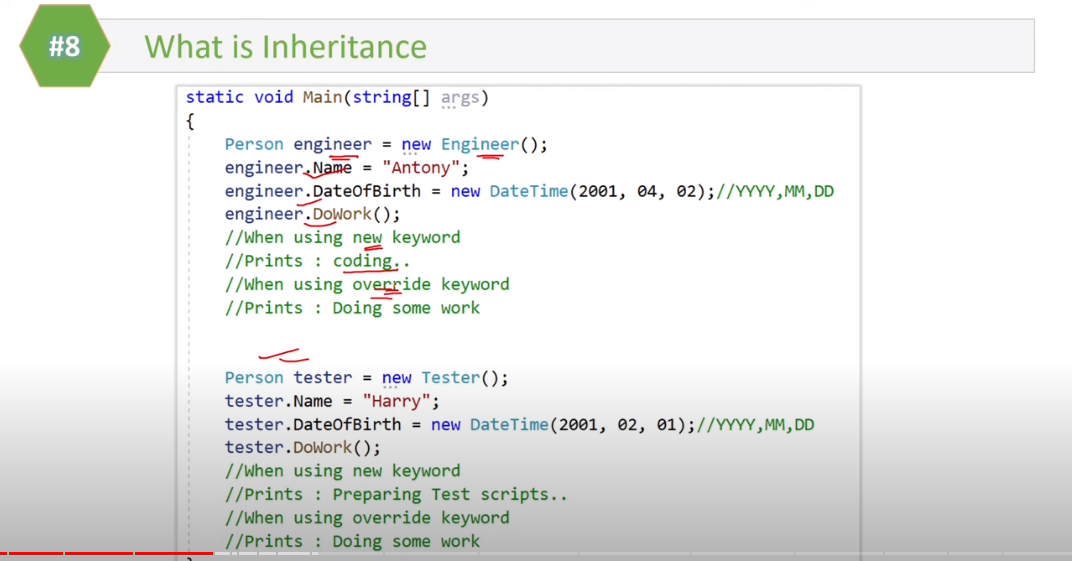


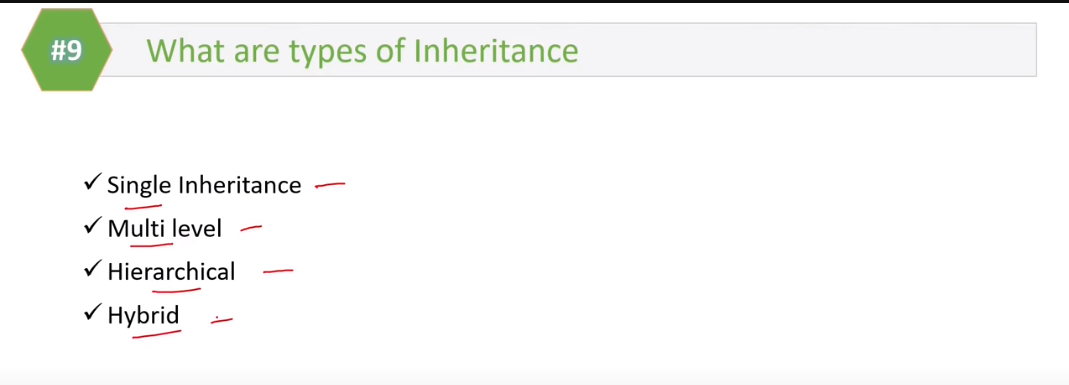


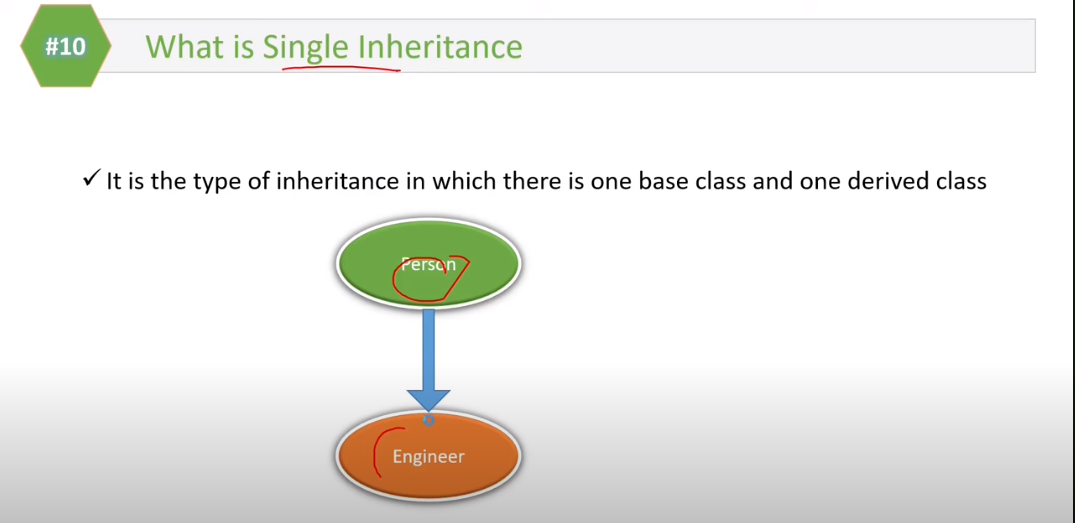


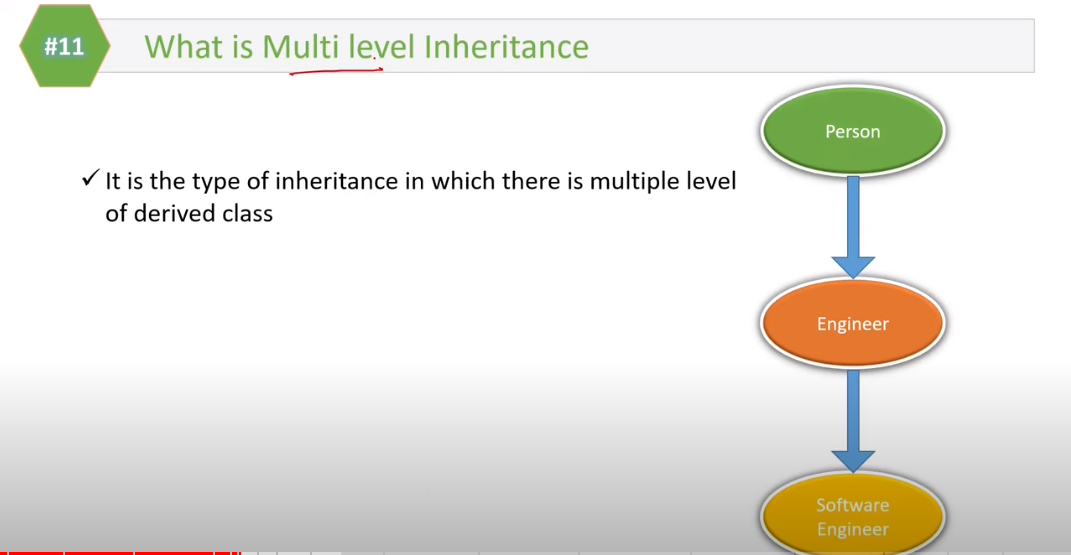


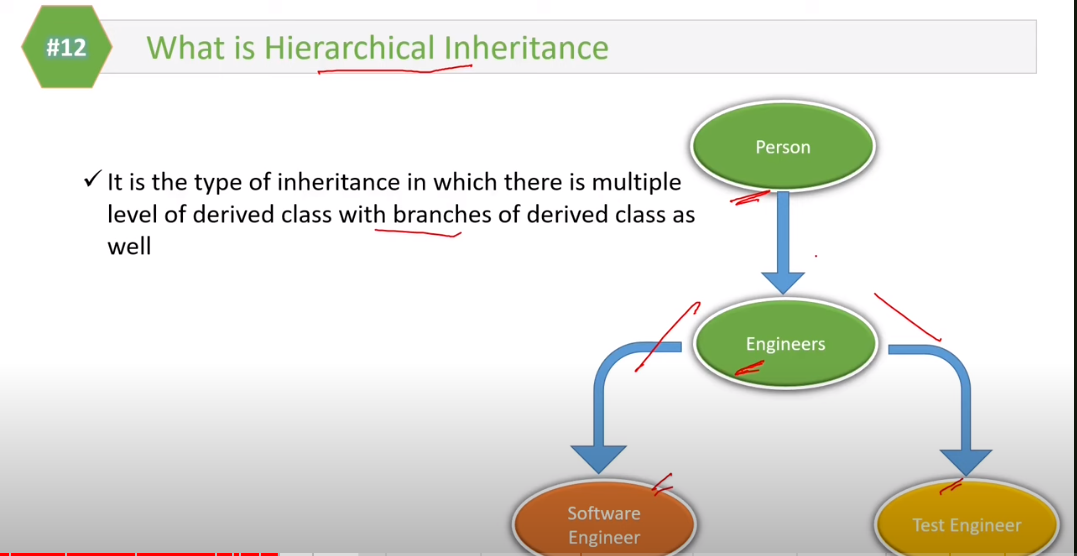


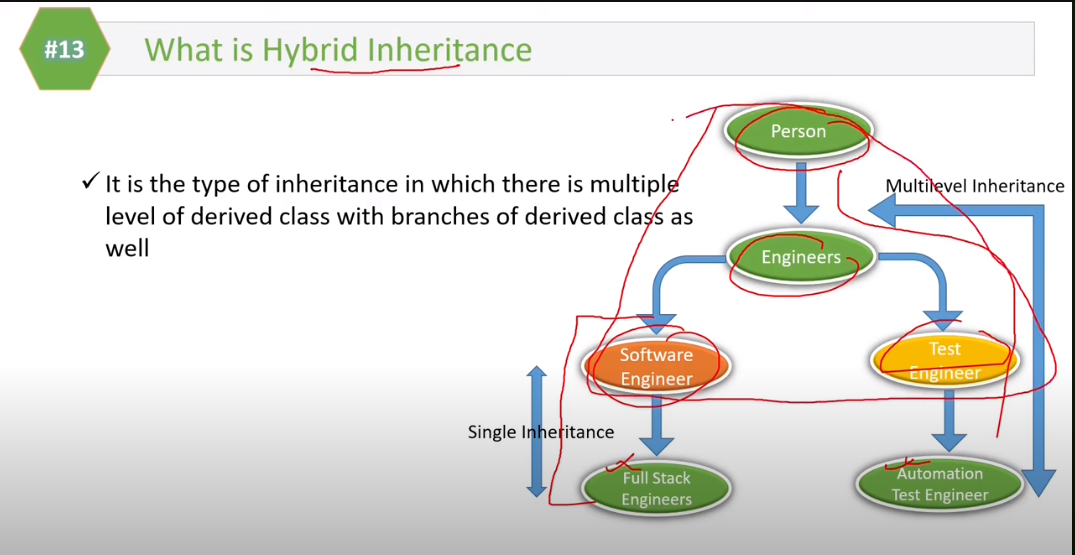


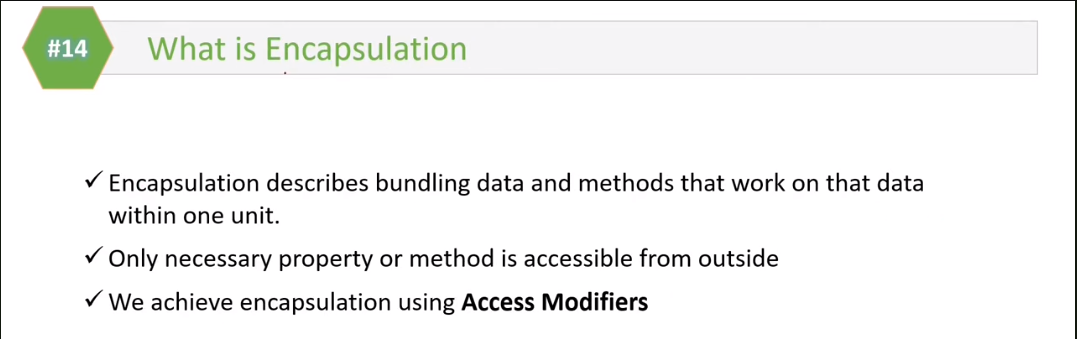


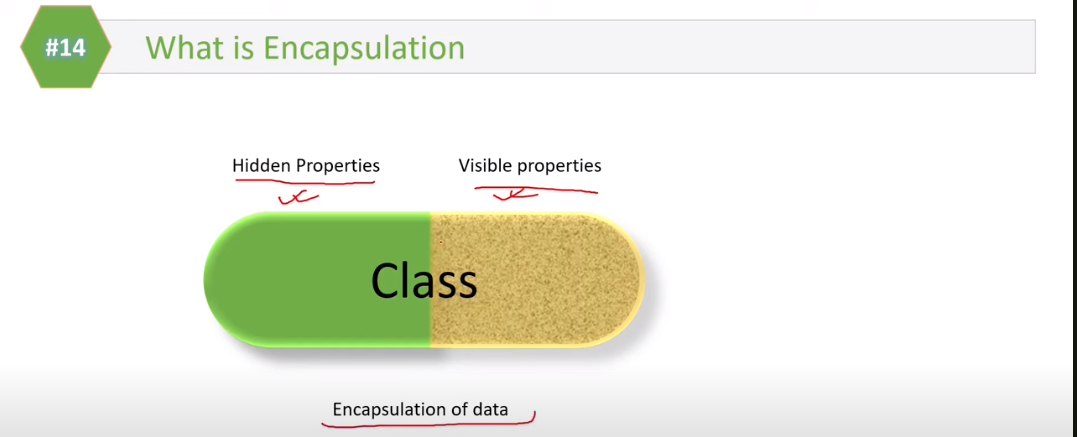


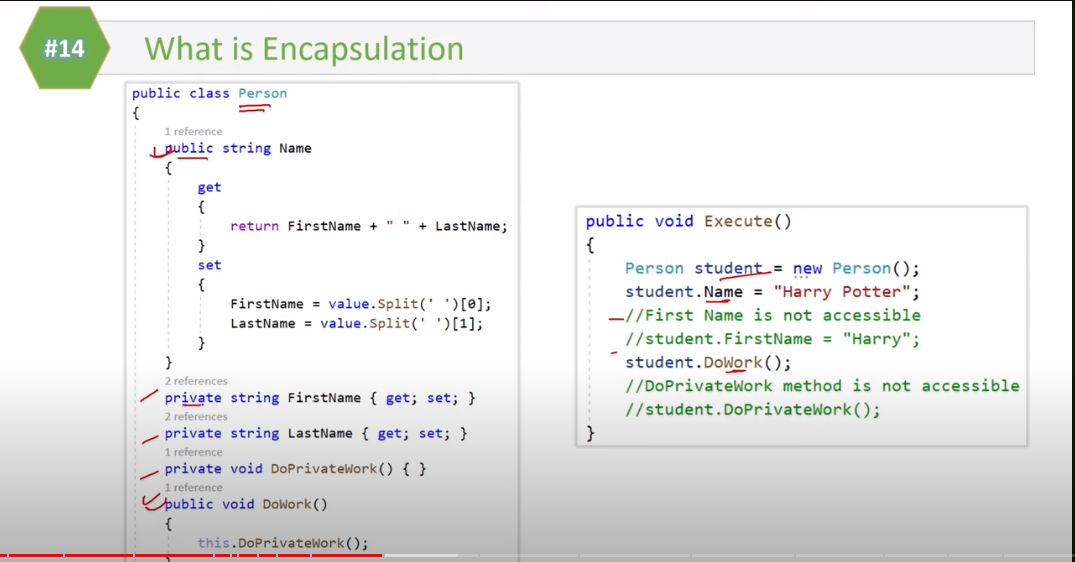


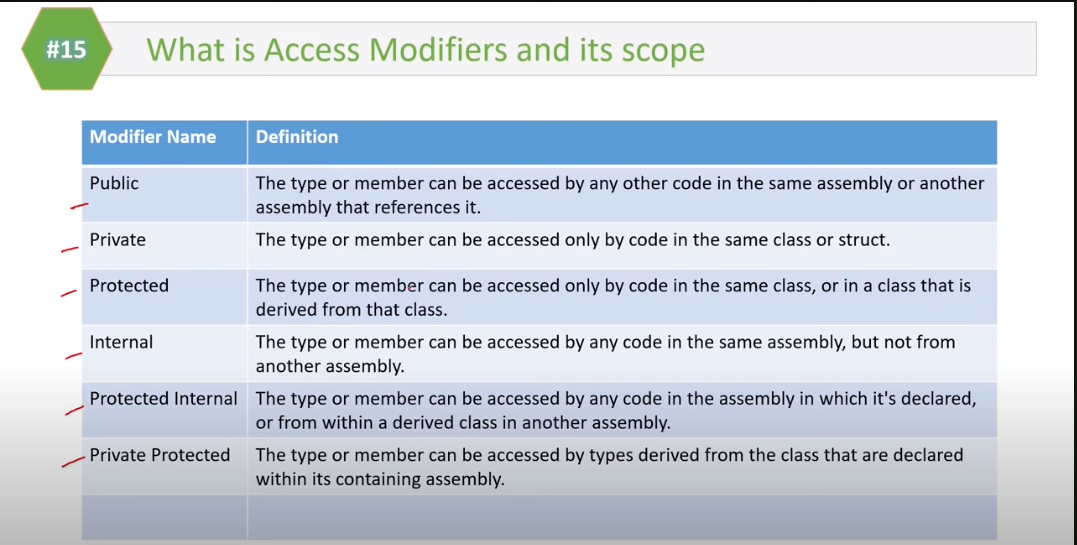


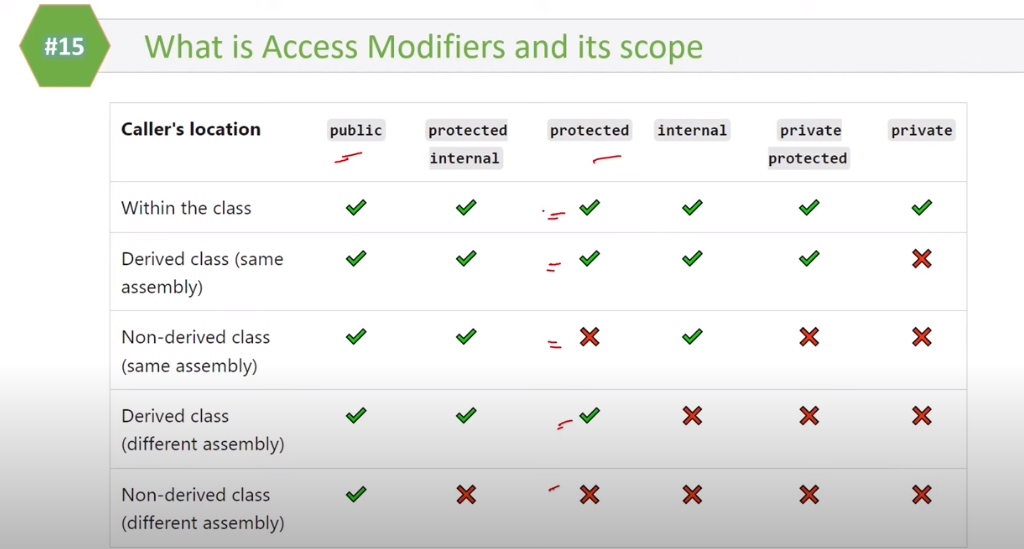


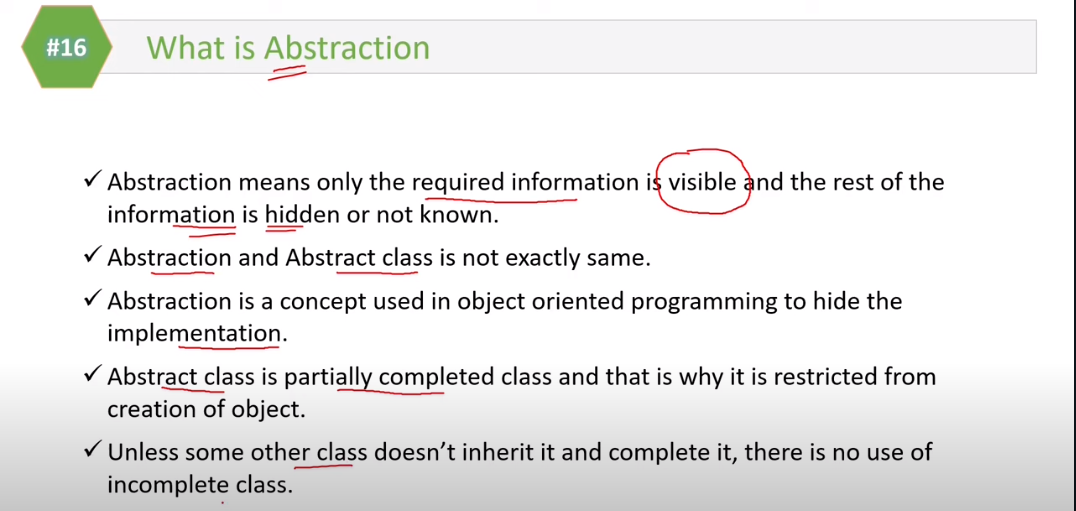


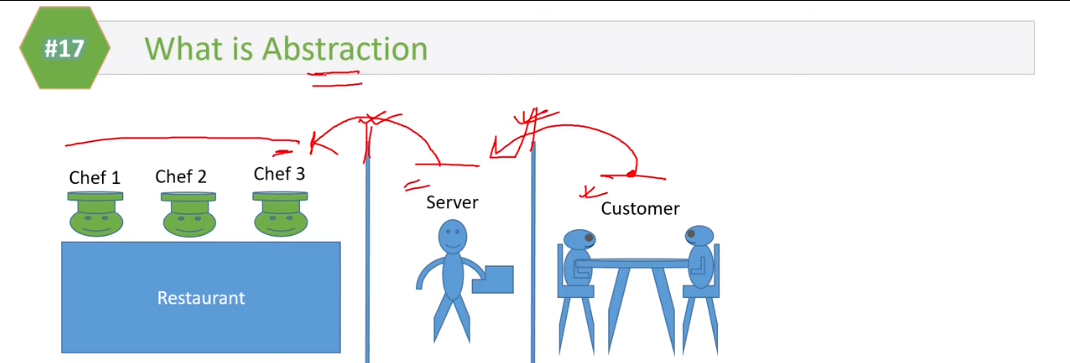


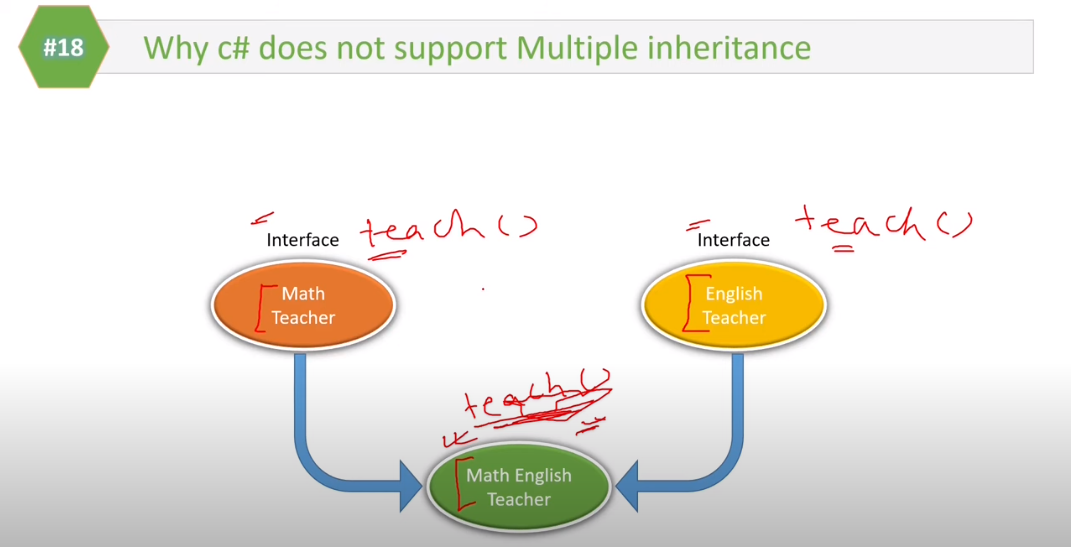


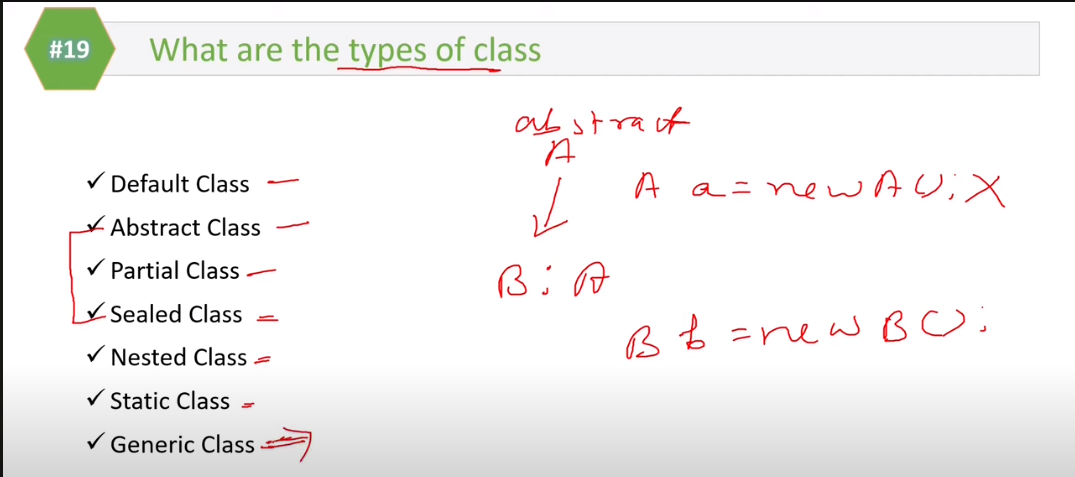


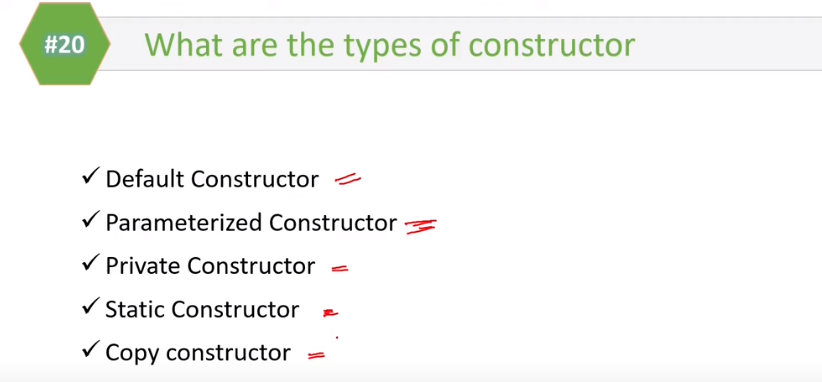


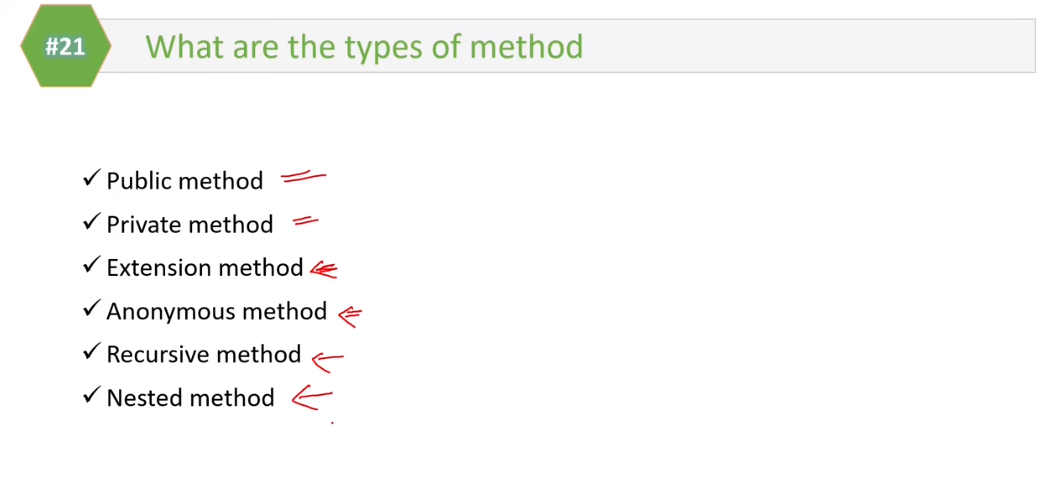






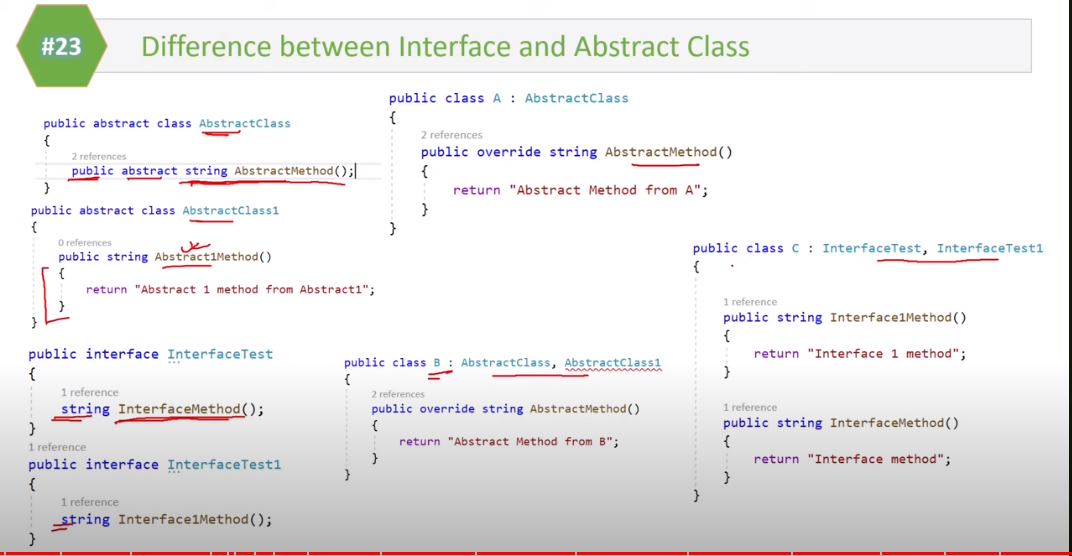


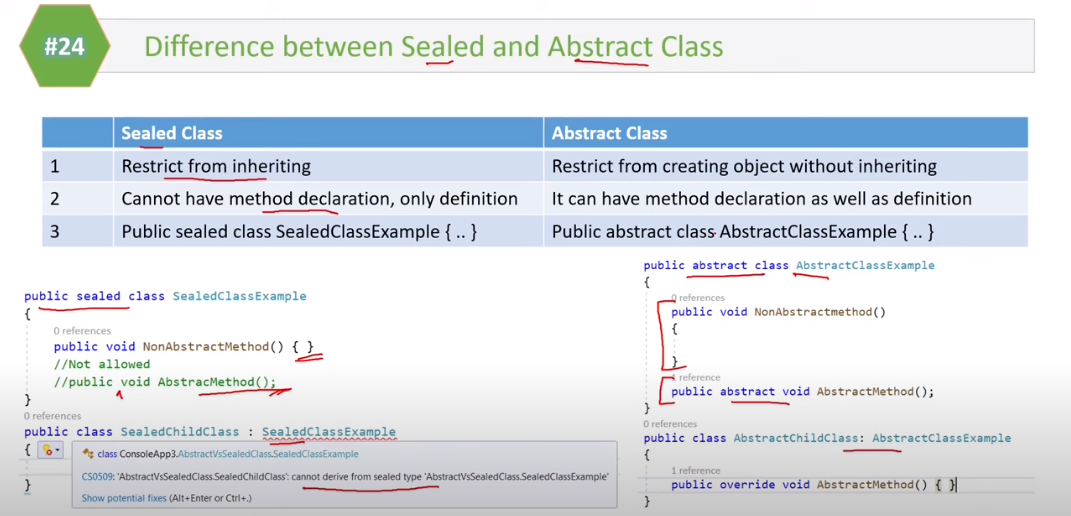


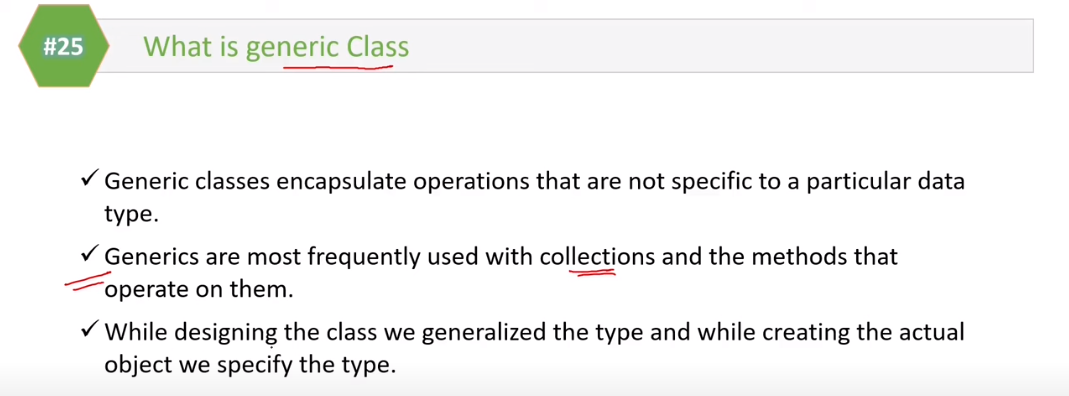


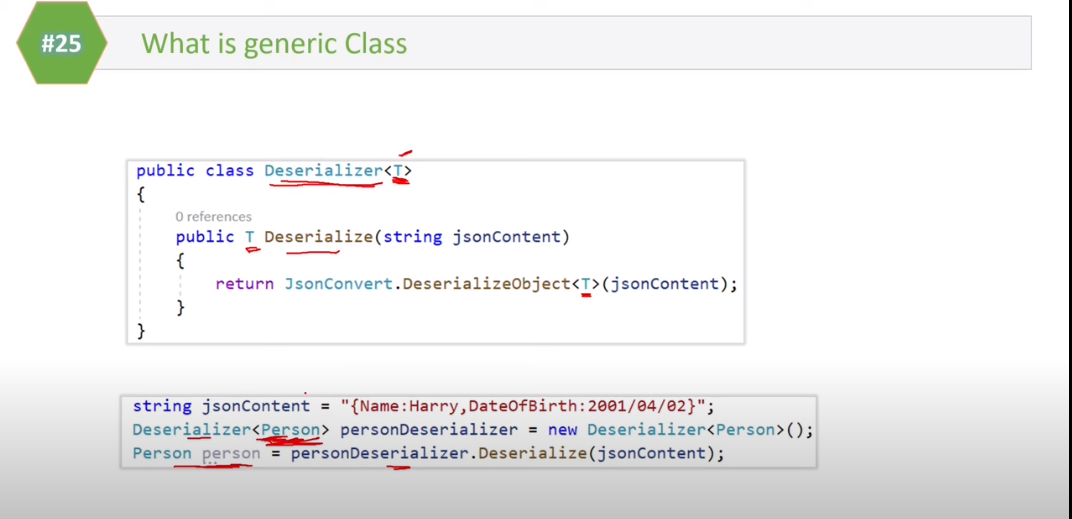












Solid Principles:

SOLID Acronym

S : Single Responsibility Principle (SRP)

O : Open closed Principle (OSP)

L :  Liskov substitution Principle (LSP)

I  :  Interface Segregation Principle (ISP)

D : Dependency Inversion Principle (DIP)

**Single Responsibility Principle**

1. "A class should have only one reason to change”

2. Every module or class should have responsibility over a single part of the functionality provided by the software, and that responsibility should be entirely encapsulated by the class

3) The simplest way to apply OCP is to implement the new functionality on new derived class

4) Allow client to access the original class with abstract interface



Q1. What is Datatype?

Q2. What do you mean by value type and reference type?

Q3. What are Nullable types?

How to use Nullable types in .NET Framework?

Q4. How to check that a nullable variable is having value

Q5. What are the differences between int, Int16, Int32 and Int64?

Q6. What is ?? Operator in C#?

Abstract Class and Interface

Q1. What is an abstract Class?

Q2. How to create an abstract Class in C#?

Q3. When to create an abstract class? Q4. What are common design guidelines for an abstract classQ5. What is an interface and how to use it

Q6. When to use an interface?

Q7. What are the disadvantages of an interface?

Q8. What are common design guidelines for the interface

Page13

Q9. How can you achieve multiple inheritances (implementation) using interfaces?

Q10. What is the difference between an interface and an abstract class?