**Assignment-3**

**What are the six combinations of access modifier keywords and what do they do?**

public: Accessible from anywhere.

private: Accessible only within the containing class.

protected: Accessible within the containing class and derived classes.

internal: Accessible within the same assembly.

protected internal: Accessible within the same assembly or any derived classes.

private protected: Accessible within the containing class or derived classes within the same assembly.

**What is the difference between the static, const, and readonly keywords when applied to a type member?**

**static:** The member belongs to the type itself rather than to a specific object.

**const:** The value is set at compile time and cannot be changed; implicitly static.

**readonly:** The value can be set only during declaration or in the constructor; cannot be changed afterwards.

**What does a constructor do?**

A constructor initializes a new instance of a class, setting initial values for fields and performing any setup required.

**Why is the partial keyword useful?**

The partial keyword allows a class, struct, or interface to be defined across multiple files, useful for separating code logically or when using code generation tools.

**What is a tuple?**

A tuple is a data structure that can hold multiple elements of different types, allowing for the grouping of related values without creating a separate class.

**What does the C# record keyword do?**

The record keyword defines a reference type that provides built-in functionality for value equality, immutability, and concise syntax for creating data objects.

**What does overloading and overriding mean?**

Overloading: Defining multiple methods with the same name but different parameters.

Overriding: Providing a new implementation for a virtual or abstract method in a derived class.

**What is the difference between a field and a property?**

Field: A variable that holds data directly.

Property: Provides a way to get or set values, with optional logic in accessors (get and set).

**How do you make a method parameter optional?**

By providing a default value in the method signature:

void MyMethod(int x = 0) { }

**What is an interface and how is it different from an abstract class?**

Interface: Defines a contract with methods and properties that implementing classes must provide. Interfaces cannot have implementation.

Abstract class: Can have both abstract methods (without implementation) and regular methods (with implementation). Classes can inherit from only one abstract class but can implement multiple interfaces.

**What accessibility level are members of an interface?**

Members of an interface are implicitly public and cannot have any other access modifier.

**True/False. Polymorphism allows derived classes to provide different implementations of the same method.**

True

**True/False. The override keyword is used to indicate that a method in a derived class is providing its own implementation of a method.**

True

**True/False. The new keyword is used to indicate that a method in a derived class is providing its own implementation of a method.**

False (The new keyword is used to hide a method in the base class, not to override it).

**True/False. Abstract methods can be used in a normal (non-abstract) class.**

False

**True/False. Normal (non-abstract) methods can be used in an abstract class.**

True

**True/False. Derived classes can override methods that were virtual in the base class.**

True

**True/False. Derived classes can override methods that were abstract in the base class.**

True

**True/False. In a derived class, you can override a method that was neither virtual nor abstract in the base class.**

False (Only virtual or abstract methods can be overridden).

**True/False. A class that implements an interface does not have to provide an implementation for all of the members of the interface.**

False.

**True/False. A class that implements an interface is allowed to have other members that aren’t defined in the interface.**

True

**True/False. A class can have more than one base class.**

False.

**True/False. A class can implement more than one interface.**

True