**03 Object-Oriented Programming**

**Test your knowledge**

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

* Public: accessible from everywhere
* Protected: accessible in the same class or derived class
* Internal: accessible in the same assembly
* Private: accessible only in current class
* protected internal: accessible in the same assembly, or from a derived class in another assembly
* private protected: Accessible in the same class or types derived from the same class within the current assembly

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

* static: member should belong to the type itself, rather than the objects from the type.
* Const: member must be assigned a value at the time of declaration and after that, it cannot be modified.
* readonly: member can be initialized either at the time of declaration or within the constructor of the same class. Therefore, readonly fields can be used for run-time constants.

3. What does a constructor do?

Constructor is used to create an object of the class and initialize class members

4. Why is the partial keyword useful?

It is possible to split the definition of a class, a struct, an interface or a method over two or more source files so multiple programmers can work on it at the same time.

5. What is a tuple?

a tuple is a lightweight data structures that you can use to group loosely related data elements.

6. What does the C# record keyword do?

A record in C# is a [class](https://docs.microsoft.com/en-us/dotnet/csharp/language-reference/keywords/class) or [struct](https://docs.microsoft.com/en-us/dotnet/csharp/language-reference/builtin-types/struct) that provides special syntax and behavior for working with data models. record keyword defines a reference type that provides built-in functionality for encapsulating data.

7. What does overloading and overriding mean?

* Method overriding — happens between base class and derived class; same method signature(access modifiers, method name, input/output); derived class can have different implementations for the same method
* Method overloading — multiple methods in the same class, same signature(access modifiers, method name) but different input/output

8. What is the difference between a field and a property?

## Fields are normal variable members of a class. Properties are special methods to get and set fields’ values. Generally, we should declare your fields as private, then use *Properties* to get and set their values.

9. How do you make a method parameter optional?

By assigning a default value to the optional parameter.

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

Interface defines the a contract which is implemented by the derived classes.

## Interface vs Abstract Class

* Interface supports multiple inheritance but abstract class does not
* Interface can not have instanced constructor but abstract class can have
* Interface has by default all members abstract and public, but abstract class can have abstract and concrete members
* Interface can not have fields but abstract class can have fields.

11. What accessibility level are members of an interface?

public

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

True

13. 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

14. 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

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

False

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

True

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

True

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

True

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

False

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

False

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

True

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

False

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

True