03 Object-Oriented Programming

Test your knowledge

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

--There are six different types of access modifiers.

--Public.

--Private.

--Protected.

--Internal.

--Protected Internal.

--Private Protected

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

a type member?

-- If you know the value will never, ever, ever change for any reason, use const.

--If you're unsure of whether or not the value will change, but you don't want other classes --or code to be able to change it, use readonly.

--If you need a field to be a property of a type, and not a property of an instance of that type, use static.

--A const value is also implicitly static.

3. What does a constructor do?

--In C#, constructor is a special method which is invoked automatically at the time of object creation. It is used to initialize the data members of new object generally. The constructor in C# has the same name as class or struct.

4. Why is the partial keyword useful?

-- With the help of partial classes, multiple developers can work simultaneously in the same class in different files.

--With the help of a partial class concept, you can split the UI of the design code and the business logic code to read and understand the code.

--When you were working with automatically generated code, the code can be added to the class without having to recreate the source file like in Visual studio.

--You can also maintain your application in an efficient manner by compressing large classes into small ones.

5. What is a tuple?

-- The word Tuple means “a data structure which consists of the multiple parts”. So tuple is a data structure which gives you the easiest way to represent a data set which has multiple values that may/may not be related to each other.

6. What does the C# record keyword do?

-- Beginning with C# 9, you use the record keyword to define a reference type that provides built-in functionality for encapsulating data.

--C# 10 allows the record class syntax as a synonym to clarify a reference type, and record struct to define a value type with similar functionality

7. What does overloading and overriding mean?

-- Overloading occurs when two or more methods in one class have the same method name but different parameters.

--Overriding occurs when two methods have the same method name and parameters. One of the methods is in the parent class, and the other is in the child class.

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

-- A field is a variable of any type that is declared directly in a class. A property is a member that provides a flexible mechanism to read, write or compute the value of a private field. A field can be used to explain the characteristics of an object or a class

9. How do you make a method parameter optional?

-- We can make a parameter optional by assigning default values for that parameter.

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

-- An abstract class permits you to make functionality that subclasses can implement or override whereas an interface only permits you to state functionality but not to implement it. A class can extend only one abstract class while a class can implement multiple interfaces.

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.

--False

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.

--True

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.

--False

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

--False

19. True/False. In a derived class, you can override a method that was neither virtual non abstract in the base class.

--True

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

--True

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.

--True

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

--False