Concept Questions:

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

Access modifiers keywords are used to specify the declared accessibility of a member or a type. There are six combinations of AMK which are:

1. Public: the type or member can be accessed by any other code in the same assembly or another assembly that references it.
2. Private: the type or member can be accessed only by code in the same class or struct.
3. Protected: the type or member can be accessed only by code in the same class, or in a class that is derived from that class.
4. Internal: the type or member can be accessed by any code in the same assembly, but not from another assembly.
5. protected internal: the type or member can be accessed by any code in the assembly in which it is declared, or from within a derived class in another assembly.
6. private protected: the type or member can be accessed by types derived from the class that are declared within its containing assembly.

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

a. Static is used to define a member that belongs to the type itself, rather than to any specific instance of that type. E.g. a static variable can be accessed using its class name without creating an instance of the class.

b. Const is used to define a constant value that cannot be changed once it has been set, which means we cannot assign a new value to this var. A const var must be initialized at the time of declaration, and its value must be a compile-time constant.

c. Readonly is used to define a var whose value can be set only once, either at the time of declaration or in the constructor of the class. Unlike const, a readonly variable can have a different value for different objects of the same class.

Their differences contains mutability, scope, memory allocation, and initialization. Static can be mutable or immutable, while const and readonly are immutable cannot be changed. Static var belongs to the type itself and can be accessed from any instance of the class or from outside the class. Const and readonly vars are instance-level vars and can only be accessed from within an instance of the class. Static and readonly vars can be initialized at the time of declaration or in a static constructor / constructor of the class.

3. What does a constructor do?

A constructor is special method that is used to initialize an object of a class. It is called automatically when an instance of the class is created using the “new” keyword. The main purpose of a constructor is to ensure that an object is initialized with valid data and in a consistent state.

4. Why is the partial keyword useful?

For easier maintenance and manipulation by splitting class, struct, or interface into multiple files. The definition of the type will be split across multiple files.

5. What is a tuple?

A tuple is a data structure that can be hold multiple values of different types in a single object. It can return multiple values from a method or pass multiple values to a method. It can also store multiple values temporarily.

6. What does the C# record keyword do?

The keyword is used to define a new type of data structure called “records”. It is particularly useful for modeling immutable data.

7. What does overloading and overriding mean?

Overloading: can define multiple methods with the same name but different parameter lists in a class. This allows a method to be used with different argument to perform different operations.

Overriding: can define a method in a subclass that has the same name, return type, and parameter list as a method in its superclass. This allows the subclass to provide its own implementation of the method, while still maintaining the same signature as the superclass method.

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

The main difference between a field and a property is that a field is a simple variable that holds a value directly, while a property provides a way to access and modify the value of a private field in a controlled way. Using properties can help enforce encapsulation and provide additional functionality, such as validation or computation, when accessing and modifying data within a class or struct.

9. How do you make a method parameter optional?

By assigning default values to the parameter when creating a method. Then it can be omitted when calling the method.

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

An interface is a contract that defines a set of methods, properties, and events that a class must implement. Like a requirement of the project that must be followed. It will not provide any implementation of its members. The main difference is that abstract class can provide some implementation and can have both abstract and non-abstract members, while an interface does not provide any implementation. A class can implement multiple interfaces, but can only inherit from one abstract class.

Interfaces are commonly used to define common functionality across different types of classes, while abstract classes are commonly used to provide a base class for related classes with common functionality.

11. What accessibility level are members of an interface?

By default they are public. They can not have access modifiers such as private, protected, or internal. This is because an interface only defines a contract that a class must follow, and the access level of its members should be defined by the implementing class.

12. True/False. Polymorphism allows derived classes to provide different implementations

of the same method. T

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. T

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. T

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

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

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

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

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

20. True/False. A class that implements an interface does not have to provide an

implementation for all of the members of the interface. F

21. True/False. A class that implements an interface is allowed to have other members that

aren’t defined in the interface. T

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

23. True/False. A class can implement more than one interface. What is meant by the terms managed resource and unmanaged resource in .NET T

24. What's the purpose of Garbage Collector in .NET?

The purpose of the Garbage Collector (GC) in .NET is to automatically manage memory allocation and deallocation for objects on the managed heap. When an object is no longer being used by the program, the GC identifies it as garbage and frees up the memory that was allocated for it.