**C# has six combinations of access modifier keywords:**

* public: accessible from anywhere in the program.
* private: accessible only within the class where it is defined.
* protected: accessible within the class where it is defined, and in derived classes.
* internal: accessible only within the same assembly.
* protected internal: accessible within the same assembly, and in derived classes.
* private protected: accessible within the same class and in derived classes, but only within the same assembly.

The static keyword means that a member belongs to the type itself, rather than to instances of the type. The const keyword is used to define a compile-time constant value that cannot be changed. The readonly keyword is used to declare a value that can be set only once, typically in the constructor or initialization block of the class.

A constructor is a special method in a class that is used to initialize objects of that class. It typically has the same name as the class, and may take one or more parameters to initialize the object's fields.

The partial keyword is useful because it allows a class, struct, or interface to be defined in multiple files. This can make it easier to organize large or complex classes, and can allow different developers to work on different parts of a class without interfering with each other.

A tuple is a lightweight data structure in C# that allows multiple values of different types to be stored together in a single object. Tuples can be useful when a method needs to return multiple values, or when multiple values need to be passed as parameters to a method.

The C# record keyword is used to define classes that are intended primarily to store data, rather than to encapsulate behavior. Records can be used to define immutable data structures with automatically generated methods for comparison, hashing, and string representation.

Overloading means defining multiple methods or constructors with the same name but different parameter lists. Overloading allows the same method name to be used for different purposes. Overriding means providing a new implementation for a method that is already defined in a base class or interface. Overriding allows a subclass to provide a different implementation of a method that is inherited from its parent class or interface.

A field is a variable that belongs to an instance of a class or struct, and can be accessed directly by the class or struct. A property is a member of a class or struct that provides a way to access or modify a private field, while also allowing the class to enforce constraints or perform additional logic on the value.

To make a method parameter optional, you can give it a default value. This allows the method to be called with or without that parameter, and if the parameter is omitted, the default value is used.

An interface is a type that defines a set of methods and properties that a class must implement. An interface provides a way to define a contract between different classes, allowing them to communicate with each other in a standardized way. An abstract class is a class that cannot be instantiated, and is used as a base class for other classes. An abstract class can provide a partial implementation of a class, including abstract methods that must be implemented by subclasses. The main difference between an interface and an abstract class is that an interface defines only the methods and properties that must be implemented, while an abstract class can also provide some implementation.

Members of an interface are by default public and abstract.

True

True

True

True

True

True

True

False

False

False

False

True





