1. Which of the following are value types in C#?

- [ ] `string`
- [x] `int`
- [x] `enum`
- [ ] `object`
- [x] `struct`

Answer: `int`, `enum`, `struct`

2. Which of the following methods can be used to prevent a class from being inherited?

- [x] Use the `sealed` keyword.
- [ ] Use the `abstract` keyword.
- [x] Use the `private` constructor.
- [ ] Use the `virtual` keyword.

Answer: `Use the sealed keyword.`, `Use the private constructor.`

3. Which of the following are true about delegates in C#?

- [x] Delegates are type-safe.
- [x] Delegates can point to multiple methods.
- [x] Delegates are reference types.
- [ ] Delegates can be used with `async` and `await`.

Answer: `Delegates are type-safe.`, `Delegates can point to multiple methods.`, `Delegates are reference types.`

4. Which of the following statements about `readonly` fields in C# are correct?

- [x] `readonly` fields can be assigned only in the constructor or at the declaration.
- [ ] `readonly` fields can be modified anywhere in the class.
- [x] `readonly` fields are evaluated at runtime.
- [ ] `readonly` fields can be initialized only at compile-time.

Answer: `readonly fields can be assigned only in the constructor or at the declaration.`, `readonly fields are evaluated at runtime.`

5. Which of the following are valid ways to pass parameters in C#?

- [x] `ref`
- [x] `out`
- [x] `params`
- [ ] `default`

Answer: `ref`, `out`, `params`

6. Which of the following are valid statements about `abstract` classes in C#?

- [x] An abstract class cannot be instantiated.
- [x] An abstract class can contain non-abstract methods.
- [ ] An abstract class cannot have a constructor.
- [x] An abstract class can contain fields and properties.

Answer: `An abstract class cannot be instantiated.`, `An abstract class can contain non-abstract methods.`, `An abstract class can contain fields and properties.`

7. Which of the following are true about the `lock` statement in C#?

- [x] The `lock` statement is used to prevent multiple threads from accessing a code block simultaneously.
- [x] The `lock` statement requires a reference type as a parameter.
- [ ] The `lock` statement is used to ensure the object is garbage collected.
- [x] The `lock` statement should be used with care to avoid deadlocks.

Answer: `The lock statement is used to prevent multiple threads from accessing a code block simultaneously.`, `The lock statement requires a reference type as a parameter.`, `The lock statement should be used with care to avoid deadlocks.`

8. Which of the following are true for exception handling in C#?

- [x] You can have multiple `catch` blocks for a single `try`.

- [x] You can rethrow an exception using the `throw` statement.

- [ ] All exceptions must be caught by a `catch` block.

- [x] You can have a `finally` block without a `catch` block.

Answer: `You can have multiple catch blocks for a single try.`, `You can rethrow an exception using the throw statement.`, `You can have a finally block without a catch block.`

9. Which of the following are true about the `dynamic` type in C#?

- [x] The `dynamic` type allows operations to bypass compile-time type checking.

- [x] The `dynamic` type can change type at runtime.

- [ ] The `dynamic` type is the same as `var`.

- [x] Operations on a `dynamic` type are resolved at runtime.

Answer: `The dynamic type allows operations to bypass compile-time type checking.`, `The dynamic type can change type at runtime.`, `Operations on a dynamic type are resolved at runtime.`

10. Which of the following are true about interfaces in C#?

- [x] An interface can have properties and methods.

- [ ] An interface can have fields.

- [x] An interface can be inherited by another interface.

- [x] A class can implement multiple interfaces.

Answer: `An interface can have properties and methods.`, `An interface can be inherited by another interface.`, `A class can implement multiple interfaces.`

11. Which of the following are true about `static` classes in C#?

- [x] A `static` class cannot be instantiated.

- [x] A `static` class can contain only `static` members.

- [ ] A `static` class can have instance constructors.

- [x] A `static` class can be inherited by another class.

Answer: `A static class cannot be instantiated.`, `A static class can contain only static members.`

12. Which of the following are valid access modifiers in C#?

- [x] `public`
- [x] `private`
- [x] `protected`
- [x] `internal`

Answer: `public`, `private`, `protected`, `internal`

13. Which of the following are true about the `override` keyword in C#?

- [x] It is used to provide a new implementation of a method inherited from a base class.
- [x] The method being overridden must be marked as `virtual`, `abstract`, or `override`.
- [ ] The `override` keyword can be used with constructors.
- [x] An overridden method in the derived class can have a different return type.

Answer: `It is used to provide a new implementation of a method inherited from a base class.`, `The method being overridden must be marked as virtual, abstract, or override.`

14. Which of the following are true for extension methods in C#?

- [x] Extension methods are defined as static methods.
- [x] Extension methods can be called on `null` objects.
- [ ] Extension methods can override existing methods of the type.
- [x] Extension methods must be in a static class.

Answer: `Extension methods are defined as static methods.`, `Extension methods can be called on null objects.`, `Extension methods must be in a static class.`

15. Which of the following are valid loop constructs in C#?

- [x] `for`
- [x] `while`
- [x] `do-while`

- [x] `foreach`

Answer: `for`, `while`, `do-while`, `foreach`

16. Which of the following are true about the `out` parameter in C#?

- [x] The `out` parameter must be initialized inside the method before returning.
- [x] The `out` parameter can pass multiple values out of a method.
- [ ] The `out` parameter must be initialized before being passed into the method.
- [x] The `out` parameter can be used with asynchronous methods.

Answer: `The out parameter must be initialized inside the method before returning.`, `The out parameter can pass multiple values out of a method.`, `The out parameter can be used with asynchronous methods.`

17. Which of the following are true for anonymous types in C#?

- [x] Anonymous types are defined using the `var` keyword.
- [x] Anonymous types can have read-only properties.
- [x] Anonymous types cannot be cast to another type.
- [ ] Anonymous types can implement interfaces.

Answer: `Anonymous types are defined using the var keyword.`, `Anonymous types can have read-only properties.`, `Anonymous types cannot be cast to another type.`

18. Which of the following are true about the `foreach` loop in C#?

- [x] The `foreach` loop can be used to iterate over arrays and collections.
- [x] The `foreach` loop variable is read-only within the loop.
- [ ] The `foreach` loop requires an index variable.
- [x] The `foreach` loop can iterate over `IEnumerable` collections.

Answer: `The foreach loop can be used to iterate over arrays and collections.`, `The foreach loop variable is read-only within the loop.`, `The foreach loop can iterate over IEnumerable collections.`

19. Which of the following are true about events in C#?

- [x] Events are used to notify subscribers when something occurs.

- [x] Events can be declared using delegates.

- [x] Events can have multiple subscribers.

- [ ] Events can be invoked outside the class that declares them.

Answer: `Events are used to notify subscribers when something occurs.`, `Events can be declared using delegates.`, `Events can have multiple subscribers.`

20. Which of the following are valid statements about LINQ in C#?

- [x] LINQ can be used with arrays, collections, and databases.

- [x] LINQ queries can be written in query syntax or method syntax.

- [x] LINQ provides a unified query syntax for different data sources.

- [ ] LINQ requires a specific data source type.

Answer: `LINQ can be used with arrays, collections, and databases.`, `LINQ queries can be written in query syntax or method syntax.`, `LINQ provides a unified query syntax for different data sources.`

21. Which of the following are true about the `params` keyword in C#?

- [x] The `params` keyword allows a method to accept a variable number of arguments.

- [x] The `params` keyword must be the last parameter in the method signature.

- [ ] The `params` keyword can only be used with `int` types.

- [x] The `params` keyword can be used with an array.

Answer: `The params keyword allows a method to accept a variable number of arguments.`, `The params keyword must be the last parameter in the method signature.`, `The params keyword can be used with an array.`

22. Which of the following are true about generics in C#?

- [x] Generics allow you to define type-safe data structures.

- [x] Generics improve code reusability.

- [ ] Generics are limited to classes and cannot be used with methods.

- [x] Generics allow you to define classes and methods with placeholder types.

Answer: `Generics allow you to define type-safe data structures.`, `Generics improve code reusability.`, `Generics allow you to define classes and methods with placeholder types.`

23. Which of the following are valid statements about exception filters in C#?

- [x] Exception filters can be used to catch specific conditions in a `catch` block.

- [x] Exception filters use the `when` keyword.

- [ ] Exception filters can be applied to `finally` blocks.

- [x] Exception filters allow selective handling of exceptions based on a condition.

Answer: `Exception filters can be used to catch specific conditions in a catch block.`, `Exception filters use the when keyword.`, `Exception filters allow selective handling of exceptions based on a condition.`

24. Which of the following are true about properties in C#?

- [x] Properties provide a way to access fields with getter and setter methods.

- [x] Properties can be read-only or write-only.

- [ ] Properties must have both `get` and `set` accessors.

- [x] Properties can have different access modifiers for `get` and `set`.

Answer: `Properties provide a way to access fields with getter and setter methods.`, `Properties can be read-only or write-only.`, `Properties can have different access modifiers for get and set.`

25. Which of the following are valid ways to define a nullable type in C#?

- [x] `int?`

- [x] `Nullable<int>`

- [ ] `int[]`

- [x] `Nullable<bool>`

Answer: `int?`, `Nullable<int>`, `Nullable<bool>`

26. Which of the following are true about `indexers` in C#?

- [x] Indexers allow objects to be indexed like arrays.

- [x] Indexers can have different access modifiers for `get` and `set`.

- [ ] Indexers can only accept integer indices.

- [x] Indexers can be overloaded.

Answer: `Indexers allow objects to be indexed like arrays.`, `Indexers can have different access modifiers for get and set.`, `Indexers can be overloaded.`

27. Which of the following are true about `async` and `await` in C#?

- [x] The `async` keyword is used to mark a method as asynchronous.

- [x] The `await` keyword is used to wait for an asynchronous operation to complete.

- [x] An `async` method can return `void`, `Task`, or `Task<T>`.

- [ ] The `await` keyword can only be used in `async` methods.

Answer: `The async keyword is used to mark a method as asynchronous.`, `The await keyword is used to wait for an asynchronous operation to complete.`, `An async method can return void, Task, or Task<T>.`, `The await keyword can only be used in async methods.`

28. Which of the following are true about reflection in C#?

- [x] Reflection allows you to inspect metadata about types at runtime.

- [x] Reflection can be used to dynamically invoke methods.

- [x] Reflection can access private members of a class.

- [ ] Reflection cannot be used with generics.

Answer: `Reflection allows you to inspect metadata about types at runtime.`, `Reflection can be used to dynamically invoke methods.`, `Reflection can access private members of a class.`

29. Which of the following are valid statements about tuples in C#?

- [x] Tuples can store multiple values in a

single object.

- [x] Tuples can have named elements.

- [ ] Tuples must contain elements of the same type.

- [x] Tuples can be deconstructed into individual variables.

Answer: `Tuples can store multiple values in a single object.`, `Tuples can have named elements.`, `Tuples can be deconstructed into individual variables.`

30. Which of the following are valid statements about the `StringBuilder` class in C#?

- [x] `StringBuilder` is mutable and can be modified without creating a new instance.

- [x] `StringBuilder` is more efficient than `String` for repeated string manipulations.

- [ ] `StringBuilder` is a value type.

- [x] `StringBuilder` provides methods like `Append`, `Insert`, and `Remove`.

Answer: `StringBuilder is mutable and can be modified without creating a new instance.`, `StringBuilder is more efficient than String for repeated string manipulations.`, `StringBuilder provides methods like Append, Insert, and Remove.`

31. Which of the following are true about partial classes in C#?

- [x] Partial classes allow a class definition to be split across multiple files.

- [x] Partial classes can contain methods, properties, and fields defined in separate files.

- [ ] Partial classes must be fully defined within a single file.

- [x] Partial classes are combined into a single class when the program is compiled.

Answer: `Partial classes allow a class definition to be split across multiple files.`, `Partial classes can contain methods, properties, and fields defined in separate files.`, `Partial classes are combined into a single class when the program is compiled.`

32. Which of the following are valid statements about the `params` keyword in C#?

- [x] The `params` keyword allows a method to accept a variable number of arguments.

- [x] The `params` parameter must be the last in the method signature.

- [ ] The `params` keyword can only be used with `int` types.

- [x] The `params` keyword can be used with an array.

Answer: `The params keyword allows a method to accept a variable number of arguments.`, `The params parameter must be the last in the method signature.`, `The params keyword can be used with an array.`

33. Which of the following are valid ways to implement polymorphism in C#?

- [x] Method overriding
- [x] Method overloading
- [x] Interfaces
- [ ] Abstract classes only

Answer: `Method overriding`, `Method overloading`, `Interfaces`

34. Which of the following are true about `readonly` fields in C#?

- [x] `readonly` fields can be initialized either at the point of declaration or in a constructor.
- [ ] `readonly` fields can be modified after they are initialized.
- [x] `readonly` fields are evaluated at runtime.
- [x] `readonly` fields cannot be assigned a value outside of a constructor.

Answer: `readonly fields can be initialized either at the point of declaration or in a constructor.`, `readonly fields are evaluated at runtime.`, `readonly fields cannot be assigned a value outside of a constructor.`

35. Which of the following are true about `foreach` loops in C#?

- [x] `foreach` can iterate over any collection that implements `IEnumerable`.
- [x] The loop variable in `foreach` is read-only within the loop.
- [ ] `foreach` loops cannot be nested.
- [x] `foreach` provides an easy way to iterate over collections without needing an index.

Answer: `foreach can iterate over any collection that implements IEnumerable.`, `The loop variable in foreach is read-only within the loop.`, `foreach provides an easy way to iterate over collections without needing an index.`

36. Which of the following are true about covariance and contravariance in C#?

- [x] Covariance allows a method to return a more derived type than specified by the generic parameter.

- [x] Contravariance allows a method to accept parameters of less derived types than specified by the generic parameter.

- [ ] Covariance and contravariance only apply to reference types.

- [x] Covariance and contravariance are useful in generic interfaces and delegates.

Answer: `Covariance allows a method to return a more derived type than specified by the generic parameter.`, `Contravariance allows a method to accept parameters of less derived types than specified by the generic parameter.`, `Covariance and contravariance are useful in generic interfaces and delegates.`

37. Which of the following are valid statements about the `sealed` keyword in C#?

- [x] The `sealed` keyword prevents a class from being inherited.

- [x] A `sealed` class can still inherit from another class.

- [ ] The `sealed` keyword can be applied to methods only.

- [x] A method can be sealed to prevent further overriding.

Answer: `The sealed keyword prevents a class from being inherited.`, `A sealed class can still inherit from another class.`, `A method can be sealed to prevent further overriding.`

38. Which of the following are valid statements about `TryParse` methods in C#?

- [x] `TryParse` methods attempt to convert a string to a specific type and return a boolean result.

- [x] `TryParse` methods do not throw an exception if the conversion fails.

- [ ] `TryParse` methods can only be used with numeric types.

- [x] `TryParse` methods provide the converted value through an `out` parameter.

Answer: `TryParse methods attempt to convert a string to a specific type and return a boolean result.`, `TryParse methods do not throw an exception if the conversion fails.`, `TryParse methods provide the converted value through an out parameter.`

39. Which of the following are true about attributes in C#?

- [x] Attributes provide a way to add metadata to code elements.

- [x] Attributes can be applied to classes, methods, properties, and more.

- [x] Custom attributes can be created by inheriting from the `Attribute` class.

- [ ] Attributes are required for all code elements.

Answer: `Attributes provide a way to add metadata to code elements.`, `Attributes can be applied to classes, methods, properties, and more.`, `Custom attributes can be created by inheriting from the Attribute class.`

40. Which of the following are true about thread safety in C#?

- [x] The `lock` statement ensures that a block of code is executed by only one thread at a time.

- [x] Immutable objects are inherently thread-safe.

- [x] Using `volatile` ensures visibility of changes to variables across threads.

- [ ] Thread safety issues only occur in multi-threaded applications.

Answer: `The lock statement ensures that a block of code is executed by only one thread at a time.`, `Immutable objects are inherently thread-safe.`, `Using volatile ensures visibility of changes to variables across threads.`

41. Which of the following are valid statements about the `is` keyword in C#?

- [x] The `is` keyword checks if an object is compatible with a specific type.

- [x] The `is` keyword can be used in pattern matching.

- [ ] The `is` keyword can be used to convert types.

- [x] The `is` keyword returns `true` if the object is of the specified type or a derived type.

Answer: `The is keyword checks if an object is compatible with a specific type.`, `The is keyword can be used in pattern matching.`, `The is keyword returns true if the object is of the specified type or a derived type.`

42. Which of the following are valid ways to handle exceptions in C#?

- [x] Using `try-catch` blocks

- [x] Using `throw` to rethrow an exception

- [x] Using `finally` blocks for cleanup

- [ ] Ignoring exceptions and letting them propagate automatically

Answer: `Using try-catch blocks`, `Using throw to rethrow an exception`, `Using finally blocks for cleanup`

43. Which of the following are true about the `nameof` operator in C#?

- [x] The `nameof` operator returns the name of a variable, type, or member as a string.
- [x] The `nameof` operator is evaluated at compile-time.
- [ ] The `nameof` operator can be used to get the value of a variable.
- [x] The `nameof` operator helps avoid magic strings in code.

Answer: `The nameof operator returns the name of a variable, type, or member as a string.`, `The nameof operator is evaluated at compile-time.`, `The nameof operator helps avoid magic strings in code.`

44. Which of the following are valid statements about `Task` in C#?

- [x] `Task` represents an asynchronous operation.
- [x] `Task` can be used to run code on a different thread.
- [x] `Task` can return a value of a specified type.
- [ ] `Task` is always executed on a background thread.

Answer: `Task represents an asynchronous operation.`, `Task can be used to run code on a different thread.`, `Task can return a value of a specified type.`

45. Which of the following are true about `readonly` properties in C#?

- [x] `readonly` properties can only be assigned a value in the constructor or at the point

of declaration.

- [ ] `readonly` properties can be modified anywhere in the class.
- [x] `readonly` properties cannot be set after the constructor has finished.
- [x] `readonly` properties are often used for immutable objects.

Answer: `readonly properties can only be assigned a value in the constructor or at the point of declaration.`, `readonly properties cannot be set after the constructor has finished.`, `readonly properties are often used for immutable objects.`

46. Which of the following are valid ways to declare an array in C#?

- [x] `int[] numbers = new int[5];`
- [x] `int[] numbers = { 1, 2, 3, 4, 5 };`
- [x] `int[] numbers = new int[] { 1, 2, 3, 4, 5 };`
- [ ] `int numbers[] = { 1, 2, 3, 4, 5 };`

Answer: `int[] numbers = new int[5];`, `int[] numbers = { 1, 2, 3, 4, 5 };`, `int[] numbers = new int[] { 1, 2, 3, 4, 5 };`

47. Which of the following are true about `async` methods in C#?

- [x] An `async` method can contain `await` expressions.
- [x] An `async` method can return `Task`, `Task<T>`, or `void`.
- [ ] An `async` method must have at least one `await` expression.
- [x] An `async` method can execute synchronously if there is no `await` expression.

Answer: `An async method can contain await expressions.`, `An async method can return Task, Task<T>, or void.`, `An async method can execute synchronously if there is no await expression.`

48. Which of the following are valid statements about the `this` keyword in C#?

- [x] The `this` keyword refers to the current instance of the class.
- [x] The `this` keyword can be used to differentiate between class fields and parameters with the same name.
- [ ] The `this` keyword can only be used in static methods.
- [x] The `this` keyword can be used to chain constructors.

Answer: `The this keyword refers to the current instance of the class.`, `The this keyword can be used to differentiate between class fields and parameters with the same name.`, `The this keyword can be used to chain constructors.`

49. Which of the following are true about the `using` statement in C#?

- [x] The `using` statement is used to ensure that `IDisposable` objects are disposed of properly.
- [x] The `using` statement can be used with multiple resources.

- [ ] The `using` statement can only be used in asynchronous methods.

- [x] The `using` statement automatically disposes of the object when the block is exited.

Answer: `The using statement is used to ensure that IDisposable objects are disposed of properly.`, `The using statement can be used with multiple resources.`, `The using statement automatically disposes of the object when the block is exited.`

50. Which of the following are true about generics in C#?

- [x] Generics allow you to define type-safe data structures.

- [x] Generics improve code reusability.

- [ ] Generics are limited to classes and cannot be used with methods.

- [x] Generics allow you to define methods with placeholder types.

Answer: `Generics allow you to define type-safe data structures.`, `Generics improve code reusability.`, `Generics allow you to define methods with placeholder types.`

51. Which of the following are valid statements about lambda expressions in C#?

- [x] Lambda expressions provide a concise way to define anonymous methods.

- [x] Lambda expressions can capture variables from the surrounding scope.

- [ ] Lambda expressions cannot have parameters.

- [x] Lambda expressions can be used with LINQ queries.

Answer: `Lambda expressions provide a concise way to define anonymous methods.`, `Lambda expressions can capture variables from the surrounding scope.`, `Lambda expressions can be used with LINQ queries.`

52. Which of the following are true about `IDisposable` interface in C#?

- [x] `IDisposable` provides a mechanism for releasing unmanaged resources.

- [x] `IDisposable` is often implemented by classes that manage unmanaged resources like file handles.

- [ ] The `IDisposable` interface must be implemented by all classes.

- [x] The `Dispose` method should be called when the object is no longer needed.

Answer: `IDisposable provides a mechanism for releasing unmanaged resources.`, `IDisposable is often implemented by classes that manage unmanaged resources like file handles.`, `The Dispose method should be called when the object is no longer needed.`

53. Which of the following are true about events in C#?

- [x] Events are used to notify subscribers when something occurs.
- [x] Events can be declared using delegates.
- [x] Events can have multiple subscribers.
- [ ] Events can be invoked outside the class that declares them.

Answer: `Events are used to notify subscribers when something occurs.`, `Events can be declared using delegates.`, `Events can have multiple subscribers.`

54. Which of the following are valid statements about method overloading in C#?

- [x] Method overloading allows multiple methods with the same name but different parameters.
- [x] Method overloading improves code readability and usability.
- [ ] Method overloading allows different return types for the same method signature.
- [x] Method overloading cannot be achieved by changing the return type alone.

Answer: `Method overloading allows multiple methods with the same name but different parameters.`, `Method overloading improves code readability and usability.`, `Method overloading cannot be achieved by changing the return type alone.`

55. Which of the following are true about the `null-coalescing` operator (`??`) in C#?

- [x] The `null-coalescing` operator returns the left-hand operand if it is not `null`.
- [x] The `null-coalescing` operator returns the right-hand operand if the left-hand operand is `null`.
- [ ] The `null-coalescing` operator can be used with non-nullable types.
- [x] The `null-coalescing` operator is useful for providing default values.

Answer: `The null-coalescing operator returns the left-hand operand if it is not null.`, `The null-coalescing operator returns the right-hand operand if the left-hand operand is null.`, `The null-coalescing operator is useful for providing default values.`

56. Which of the following are true about the `dynamic` keyword in C#?

- [x] The `dynamic` keyword allows for late binding, where type checking occurs at runtime.

- [x] `dynamic` can be used to store any type of object.

- [x] Operations on `dynamic` objects are resolved at runtime.

- [ ] The `dynamic` keyword enforces strict compile-time type checking.

Answer: `The dynamic keyword allows for late binding, where type checking occurs at runtime.`, `dynamic can be used to store any type of object.`, `Operations on dynamic objects are resolved at runtime.`

57. Which of the following are true about the `out` keyword in C#?

- [x] The `out` keyword allows a method to return multiple values.

- [x] The `out` parameter must be initialized before the method returns.

- [ ] The `out` keyword can only be used with value types.

- [x] The `out` parameter is passed by reference.

Answer: `The out keyword allows a method to return multiple values.`, `The out parameter must be initialized before the method returns.`, `The out parameter is passed by reference.`

58. Which of the following are valid statements about the `new` keyword in C#?

- [x] The `new` keyword is used to create an instance of a type.

- [x] The `new` keyword can be used to hide a member inherited from a base class.

- [ ] The `new` keyword is only used with classes.

- [x] The `new` keyword can be used to allocate memory for arrays.

Answer: `The new keyword is used to create an instance of a type.`, `The new keyword can be used to hide a member inherited from a base class.`, `The new keyword can be used to allocate memory for arrays.`

59. Which of the following are true about the `Array` class in C#?

- [x] The `Array` class provides methods like `Sort`, `Reverse`, and `BinarySearch`.

- [x] The `Array` class is the base class for all arrays in C#.

- [ ] Arrays in C# are always zero-based.

- [x] The `Array` class provides properties like `Length` and `Rank`.

Answer: `The Array class provides methods like Sort, Reverse, and BinarySearch.`, `The Array class is the base class for all arrays in C#.`, `The Array class provides properties like Length and Rank.`

60. Which of the following are valid statements about access modifiers in C#?

- [x] `public` allows access to the member from any code.

- [x] `private` restricts access to the member within its own class.

- [x] `protected` allows access to the member within its own class and derived classes.

- [x] `internal` allows access to the member within the same assembly.

Answer: `public allows access to the member from any code.`, `private restricts access to the member within its own class.`, `protected allows access to the member within its own class and derived classes.`, `internal allows access to the member within the same assembly.`

61. Which of the following are true about `readonly` fields in C#?

- [x] `readonly` fields can be assigned a value only in the constructor or at the point of declaration.

- [x] `readonly` fields cannot be modified after they are initialized.

- [ ] `readonly` fields can be assigned a value multiple times.

- [x] `readonly` fields are often used for constants that need to be initialized at runtime.

Answer: `readonly fields can be assigned a value only in the constructor or at the point of declaration.`, `readonly fields cannot be modified after they are initialized.`, `readonly fields are often used for constants that need to be initialized at runtime.`

62. Which of the following are valid statements about properties in C#?

- [x] Properties provide a way to encapsulate fields.

- [x] Properties can have `get` and `set` accessors.

- [x] Properties can be read-only or write-only.

- [ ] Properties must always have both `get` and `set` accessors.

Answer: `Properties provide a way to encapsulate fields.`, `Properties can have get and set accessors.`, `Properties can be read-only or write-only.`

63. Which of the following are true about the `foreach` loop in C#?

- [x] The `foreach` loop is used to iterate over collections.

- [x] The loop variable in `foreach` is read-only within the loop.

- [ ] The `foreach` loop requires an index variable.

- [x] The `foreach` loop can iterate over any collection that implements `IEnumerable`.

Answer: `The foreach loop is used to iterate over collections.`, `The loop variable in foreach is read-only within the loop.`, `The foreach loop can iterate over any collection that implements IEnumerable.`

64. Which of the following are valid ways to declare an enum in C#?

- [x] `enum Colors { Red, Green, Blue }`

- [x] `enum Colors : byte { Red, Green, Blue }`

- [ ] `enum Colors = { Red, Green, Blue }`

- [x] `enum Colors { Red = 1, Green = 2, Blue = 3 }`

Answer: `enum Colors { Red, Green, Blue }`, `enum Colors : byte { Red, Green, Blue }`, `enum Colors { Red = 1, Green = 2, Blue = 3 }`

65. Which of the following are true about `finally` blocks in C#?

- [x] The `finally` block is executed after the `try` and `catch` blocks, regardless of whether an exception was thrown.

- [x] The `finally` block is used to clean up resources.

- [x] The `finally` block is always executed, even if a `return` statement is encountered in the `try` block.

- [ ] The `finally` block is optional and not always needed.

Answer: `The finally block is executed after the try and catch blocks, regardless of whether an exception was thrown.`, `The finally block is used to clean up resources.`, `The finally block is always executed, even if a return statement is encountered in the try block.`, `The finally block is optional and not always needed.`

66. Which of the following are true about anonymous methods in C#?

- [x] Anonymous methods provide a way to define inline methods.
- [x] Anonymous methods can capture variables from the surrounding scope.
- [ ] Anonymous methods must have a name.
- [x] Anonymous methods can be assigned to delegates.

Answer: `Anonymous methods provide a way to define inline methods.`, `Anonymous methods can capture variables from the surrounding scope.`, `Anonymous methods can be assigned to delegates.`

67. Which of the following are valid statements about abstract classes in C#?

- [x] Abstract classes cannot be instantiated directly.
- [x] Abstract classes can contain both abstract and non-abstract methods.
- [x] Abstract methods must be implemented by derived classes.
- [ ] Abstract classes cannot have constructors.

Answer: `Abstract classes cannot be instantiated directly.`, `Abstract classes can contain both abstract and non-abstract methods.`, `Abstract methods must be implemented by derived classes.`

68. Which of the following are true about the `lock` statement in C#?

- [x] The `lock` statement ensures that a block of code is executed by only one thread at a time.
- [x] The `lock` statement requires a reference type as a parameter.
- [ ] The `lock` statement can only be used with `static` methods.
- [x] The `lock` statement should be used carefully to avoid deadlocks.

Answer: `The lock statement ensures that a block of code is executed by only one thread at a time.`, `The lock statement requires a reference type as a parameter.`, `The lock statement should be used carefully to avoid deadlocks.`

69. Which of the following are valid statements about constructors in C#?

- [x] Constructors are special methods used to initialize objects.
- [x] Constructors can be overloaded with different parameter lists.

- [x] A constructor can be `private`.

- [ ] Constructors must always have parameters.

Answer: `Constructors are special methods used to initialize objects.`, `Constructors can be overloaded with different parameter lists.`, `A constructor can be private.`

70. Which of the following are true about interfaces in C#?

- [x] An interface can contain method signatures and properties.

- [ ] An interface can contain fields.

- [x] A class can implement multiple interfaces.

- [x] An interface can inherit from another interface.

Answer: `An interface can contain method signatures and properties.`, `A class can implement multiple interfaces.`, `An interface can inherit from another interface.`

71. Which of the following are true about the `null` keyword in C#?

- [x] The `null` keyword represents the absence of a value.

- [x] Reference types can be assigned `null`.

- [ ] Value types can be assigned `null` without being nullable.

- [x] `null` can be checked using the `==` operator.

Answer: `The null keyword represents the absence of a value.`, `Reference types can be assigned null.`, `null can be checked using the == operator.`

72. Which of the following are valid statements about `sealed` classes in C#?

- [x] A `sealed` class cannot be inherited.

- [x] A `sealed` class can still inherit from another class.

- [ ] A `sealed` class must be a `static` class.

- [x] A `sealed` method in a class cannot be overridden in derived classes.

Answer: `A sealed class cannot be inherited.`, `A sealed class can still inherit from another class.`, `A sealed method in a class cannot be overridden in derived classes.`

73. Which of the following are true about the `default` keyword in C#?

- [x] The `default` keyword can be used to initialize variables to their default values.

- [x] The `default` keyword can be used in `switch` statements to specify a default case.

- [ ] The `default` keyword can only be used with reference types.

- [x] The `default` keyword can be used in generic methods to specify the default value for the type parameter.

Answer: `The default keyword can be used to initialize variables to their default values.`, `The default keyword can be used in switch statements to specify a default case.`, `The default keyword can be used in generic methods to specify the default value for the type parameter.`

74. Which of the following are valid ways to implement inheritance in C#?

- [x] Using the `:` symbol to inherit from a base class

- [x] Implementing an interface

- [ ] Inheriting from multiple classes

- [x] Using the `virtual` keyword to allow method overriding

Answer: `Using the : symbol to inherit from a base class`, `Implementing an interface`, `Using the virtual keyword to allow method overriding`

75. Which of the following are true about `async` methods in C#?

- [x] `async` methods can use the `await` keyword to wait for asynchronous operations.

- [x] `async` methods can return `void`, `Task`, or `Task<T>`.

- [ ] `async` methods are automatically executed on a separate thread.

- [x] `async` methods can execute synchronously if no `await` expressions are present.

Answer: `async methods can use the await keyword to wait for asynchronous operations.`, `async methods can return void, Task, or Task<T>.`, `async methods can execute synchronously if no await expressions are present.`

76. Which of the following are valid statements about the `volatile` keyword in C#?

- [x] The `volatile` keyword indicates that a field might be

modified by multiple threads.

- [x] The `volatile` keyword ensures that the most recent value of the field is always read.

- [ ] The `volatile` keyword can only be used with reference types.

- [x] The `volatile` keyword can be applied to fields of primitive types like `int` and `bool`.

Answer: `The volatile keyword indicates that a field might be modified by multiple threads.`, `The volatile keyword ensures that the most recent value of the field is always read.`, `The volatile keyword can be applied to fields of primitive types like int and bool.`

77. Which of the following are true about `try-catch` blocks in C#?

- [x] `try-catch` blocks are used to handle exceptions.

- [x] Multiple `catch` blocks can be used to handle different types of exceptions.

- [x] A `catch` block can rethrow an exception using the `throw` keyword.

- [ ] A `try-catch` block must always have a `finally` block.

Answer: `try-catch blocks are used to handle exceptions.`, `Multiple catch blocks can be used to handle different types of exceptions.`, `A catch block can rethrow an exception using the throw keyword.`

78. Which of the following are true about the `is` operator in C#?

- [x] The `is` operator checks if an object is compatible with a specific type.

- [x] The `is` operator can be used in pattern matching.

- [ ] The `is` operator can be used to convert types.

- [x] The `is` operator returns `true` if the object is of the specified type or a derived type.

Answer: `The is operator checks if an object is compatible with a specific type.`, `The is operator can be used in pattern matching.`, `The is operator returns true if the object is of the specified type or a derived type.`

79. Which of the following are valid statements about enumerations in C#?

- [x] Enumerations provide a way to define a set of named integral constants.

- [x] Enumerations are value types.

- [x] Enumerations can be cast to their underlying type.

- [ ] Enumerations must have unique values for each member.

Answer: `Enumerations provide a way to define a set of named integral constants.`, `Enumerations are value types.`, `Enumerations can be cast to their underlying type.`

80. Which of the following are true about the `readonly` keyword in C#?

- [x] The `readonly` keyword is used to declare fields that can only be assigned once.

- [x] `readonly` fields can be assigned a value at the point of declaration or in a constructor.

- [ ] `readonly` fields can be modified after they are assigned.

- [x] `readonly` fields are often used for constants that are initialized at runtime.

Answer: `The readonly keyword is used to declare fields that can only be assigned once.`, `readonly fields can be assigned a value at the point of declaration or in a constructor.`, `readonly fields are often used for constants that are initialized at runtime.`

81. Which of the following are valid statements about the `switch` statement in C#?

- [x] The `switch` statement can be used to evaluate multiple cases based on a single expression.

- [x] The `switch` statement can handle string values.

- [x] The `switch` statement can have a `default` case that handles unmatched cases.

- [ ] The `switch` statement cannot be used with `enum` types.

Answer: `The switch statement can be used to evaluate multiple cases based on a single expression.`, `The switch statement can handle string values.`, `The switch statement can have a default case that handles unmatched cases.`

82. Which of the following are true about the `using` directive in C#?

- [x] The `using` directive is used to import namespaces.

- [x] The `using` directive can simplify code by removing the need for fully qualified names.

- [x] The `using` directive can be used with `IDisposable` objects to ensure they are disposed of properly.

- [ ] The `using` directive is required for all C# files.

Answer: `The using directive is used to import namespaces.`, `The using directive can simplify code by removing the need for fully qualified names.`, `The using directive can be used with IDisposable objects to ensure they are disposed of properly.`

83. Which of the following are valid ways to declare a delegate in C#?

- [x] `delegate void MyDelegate();`
- [x] `delegate int MyDelegate(int x);`
- [ ] `delegate void MyDelegate = () => {};`
- [x] `public delegate void MyDelegate(string message);`

Answer: `delegate void MyDelegate();`, `delegate int MyDelegate(int x);`, `public delegate void MyDelegate(string message);`

84. Which of the following are true about the `object` class in C#?

- [x] The `object` class is the base class for all types in C#.
- [x] The `object` class provides methods like `ToString`, `Equals`, and `GetHashCode`.
- [ ] The `object` class cannot be overridden.
- [x] The `object` class can be used to hold any type of data.

Answer: `The object class is the base class for all types in C#.`, `The object class provides methods like ToString, Equals, and GetHashCode.`, `The object class can be used to hold any type of data.`

85. Which of the following are valid statements about operator overloading in C#?

- [x] Operator overloading allows you to define custom behavior for operators with user-defined types.
- [x] Operator overloading can be applied to both unary and binary operators.
- [ ] Operator overloading can only be used with numeric types.
- [x] Operator overloading requires the use of the `operator` keyword.

Answer: `Operator overloading allows you to define custom behavior for operators with user-defined types.`, `Operator overloading can be applied to both unary and binary operators.`, `Operator overloading requires the use of the operator keyword.`

86. Which of the following are true about `LINQ` in C#?

- [x] `LINQ` provides a uniform query syntax to query different data sources.
- [x] `LINQ` queries can be written in both query syntax and method syntax.
- [x] `LINQ` can be used with collections, databases, XML, and more.
- [ ] `LINQ` queries are always executed immediately when they are defined.

Answer: `LINQ provides a uniform query syntax to query different data sources.`, `LINQ queries can be written in both query syntax and method syntax.`, `LINQ can be used with collections, databases, XML, and more.`

87. Which of the following are valid statements about `Task` in C#?

- [x] `Task` represents an asynchronous operation.
- [x] `Task` can be used to run code on a different thread.
- [x] `Task` can return a value of a specified type.
- [ ] `Task` is always executed on a background thread.

Answer: `Task represents an asynchronous operation.`, `Task can be used to run code on a different thread.`, `Task can return a value of a specified type.`

88. Which of the following are true about `partial` classes in C#?

- [x] `partial` classes allow a class definition to be split across multiple files.
- [x] `partial` classes can contain methods, properties, and fields defined in separate files.
- [ ] `partial` classes must be fully defined within a single file.
- [x] `partial` classes are combined into a single class when the program is compiled.

Answer: `partial classes allow a class definition to be split across multiple files.`, `partial classes can contain methods, properties, and fields defined in separate files.`, `partial classes are combined into a single class when the program is compiled.`

89. Which of the following are valid ways to declare an array in C#?

- [x] `int[] numbers = new int[5];`
- [x] `int[] numbers = { 1, 2, 3, 4, 5 };`
- [x] `int[] numbers = new int[] { 1, 2, 3, 4, 5 };`

- [ ] `int numbers[] = { 1, 2, 3, 4, 5 };`

Answer: `int[] numbers = new int[5];`, `int[] numbers = { 1, 2, 3, 4, 5 };`, `int[] numbers = new int[] { 1, 2, 3, 4, 5 };`

90. Which of the following are true about method overloading in C#?

- [x] Method overloading allows multiple methods with the same name but different parameters.
- [x] Method overloading improves code readability and usability.
- [ ] Method overloading allows different return types for the same method signature.
- [x] Method overloading cannot be achieved by changing the return type alone.

Answer: `Method overloading allows multiple methods with the same name but different parameters.`, `Method overloading improves code readability and usability.`, `Method overloading cannot be achieved by changing the return type alone.`