C4: C# Object-Oriented Programming

This course bonus helps you learn in a more meaningful manner.

Section 2: Understanding C# Class

QUIZ: C# Classes and Objects

1. **True or False:** It's recommended for a class to have a single responsibility.
    - True
    - False
2. Which keyword defines a class in C#?
    - struct
    - public
    - class
    - define
3. **True or False:** Each class object can have its own values.
    - True
    - False
4. Which keyword is used to create an object in C#?
    - new
    - define
    - create
    - class
5. **True or False:** Class and Object are the same.
    - True
    - False

Section 3: Understanding C# Class Fields

QUIZ: C# Class Fields

1. Which is the **correct** way to access a field in C#?
    - obj->Name

- obj.Name
- obj[Name]
- obj[“Name”]

2. **True or False:** The program can’t change the following field value.

    - const int MaxStudents = 120;

    - True
    - False
3. Which statement among the below is **incorrect** concerning const and read-only keywords?
    - const variables can’t be changed during runtime.
    - read-only variables can be assigned a value during initialization.
    - We can assign only literal values to a read-only variable.
4. **True or False**: Static fields are used when we want to share the same value across all the instances.
    - True
    - False
5. Which statement regarding the reference type is **incorrect**?
    - Objects are reference types in C#.
    - The value of instances is independent of each other.
    - The value of both instances gets changed.
6. **True or False**: The default access modifier for the class is internal.
    - True
    - False

Section 4: Understanding C# Class Methods

QUIZ: C# Class Methods

1. **True or False**: A method in C# can return only one type.
    - True
    - False
2. Select the method signature that accepts **DateTime** as a parameter, calculates the age, and returns it as an integer.
    - public CalculateAge(DateTime date, int age);
    - int CalculateAge(DateTime date);
    - public CalculateAge(DateTime date);

3. Which among the below is a **valid** constructor for the class Rectangle?
    - ~Rectangle() { }
    - public bool Rectangle() { return true; }
    - Constructors can't have a return value.
    - private Rectangle() { }
    - public int Rectangle(int x, int y) { return x * y; }
    - Constructors can't have a return value.
4. Which of the following is **correct** about properties?
    - It must be implemented with a private variable to hold the data.
    - We can define a read-only property with only a getter.
    - Properties can't have a default value.
5. What is the value of the variable **area** after calling the method?

```
int area = 0;
CalculateArea(10, 5, ref area);
...
public void CalculateArea(int x, int y, ref int result)
{
  result = x * y;
}
```

    - 0
    - 15
    - 50
    - Compilation error.
6. Select the option that assigns a default value of 0 to the parameter *y*.

```
…
public void CalculateArea(int x, int y)
{
  Console.WriteLine(x * y);
}
...
```

    - public void CalculateArea(int x, int y = 0)

    {
        Console.WriteLine(x * y);
    }

- public void CalculateArea(int x, int y)

```
{
    y = 0;
    Console.WriteLine(x * y);
}
```

- public void CalculateArea(int x, int y) : y(0)

```
{
    Console.WriteLine(x * y);
}
```

7. Which statement is **correct** about static classes in C#?
    - Static class usage requires more testing efforts.
    - Static classes can't be instantiated.
    - A static class can have only methods; fields aren't allowed.

Section 5: Understanding C# Interfaces

QUIZ: C# Interfaces

1. **True or False:** If a type implements an interface, it promises that this type supports certain features.
    - True
    - False
2. Which among the following is the **recommended way** to define an interface?
    - interface IShape { ... }
    - private interface IShape { … }
    - public interface IShape { … }
    - public class IShape { … }
3. Which class member among the below **cannot** be included inside an interface?
    - Methods
    - Fields
    - Properties
    - Events

4. **True or False:** Interfaces must always be implemented explicitly.
    - True
    - False
5. Which one is among the following **incorrect way** to implement the CalculateArea() method of an interface IShape?
    - decimal IShape.CalculateArea() { … }
    - public decimal CalculateArea() { … }
    - public decimal IShape.CalculateArea() { … }

    [Access modifier isn't applicable for explicitly implemented interface method]

6. **True or False:** The default implementation is supported for members, including properties.
    - True
    - False

Section 6: Understanding C# Inheritance

QUIZ: C# Inheritance

1. Which is the **right way** to derive class D from the base class B?
    - public class D: class B
    - public class D: B
    - public class B: class D
    - public class B: D
2. Which statement regarding inheritance is **correct**?
    - The class whose members are inherited is called the derived class.
    - The class that inherits the members is called the base class.
    - Inheritance enables us to create new classes that extend the behavior defined in other classes.
    - The derived class can't add methods of its own.
3. **True or False**: Use the keyword 'base' to refer to the base class in a derived class.
    - True
    - False
4. Which among the following is the **right** way to call a base class B constructor in a derived class D constructor?
    - D()

    {
        B();
    }

- D()

```
{
      new B();
}
```

- D() : base()

```
{
}
```

- D()

```
{
      base();
}
```

5. **True or False:** Access modifiers related to **‘protected’** are applicable only for the base class and derived class relationships.
    - True
    - False
6. Which among the following is **incorrect** concerning multiple inheritances in C#?
    - A class can inherit implementation from one base class only.
    - A class can implement more than one interface.
    - A class can inherit one base class and implement any number of interfaces.
    - None of the above.

Section 7: Understanding C# Polymorphism

QUIZ: C# Polymorphism

1. Which statement regarding Polymorphism is **incorrect**?
    - It refers to the ability to take multiple forms.
    - It refers to the behavior at compile time.
    - It is about allowing a derived class to override an inherited action to provide custom behavior.
2. Which of the following keywords is used for polymorphism?
    - interface.
    - base.

- virtual.
- as.

3. Which keyword is used to hide a method in polymorphism?
    - virtual.
    - new.
    - override.
4. What is the problem with the below code snippet to hide **Method1** in derived class D from the base class B?

```
public class B
{
  public void Method1() {}
}
public class D: B
{
  public new void Method1() {}
}
```

    - Class D can’t have the same method, Method1, defined in Class B.
    - The body of the method implementation is blank.
    - Method1 in Class B isn’t marked virtual.
    - The method signature must be different.
5. Which statement regarding abstract class is **incorrect**?
    - It provides a common definition that other classes must implement.
    - Methods inside it can’t be marked abstract.
    - We must override the method that is declared as abstract.
    - Class fields can’t be marked as abstract.