**ADO.NET Assignment**

What is the output of below program?

using System;  
namespace Test  
{  
    class Program  
    {  
        static void Main(string[] args)  
        {  
            Base obj = new Derive();  
            obj.F1();  
            obj.F2();  
            obj.F3();  
            Console.WriteLine("Hello World!");  
        }  
    }  
  
    public class Base  
    {  
        public void F1() { Console.WriteLine("Base F1"); }  
        public virtual void F2() { Console.WriteLine("Base F2"); }  
        public virtual void F3() { Console.WriteLine("Base F3"); }  
    }  
  
    public class Derive : Base  
    {  
        new public void F1()  
        {  
            Console.WriteLine("Derive F1");  
        }  
  
        public override void F2()  
        {  
            Console.WriteLine("Derive F2");  
        }  
  
        public new void F3()  
        {  
            Console.WriteLine("Derive F3");  
        }  
    }  
}

Output of the given code is as follows:

Base F1 // Calls base class because it is not overridden  
Derive F2 //Calling the Derive class because it is override the method  
Base F3  
Hello World!

1. Why IDisposable is used?

* IDisposable is an interface in C# which is used to release unmanaged resources and perform cleanup operations when an object is no longer needed. Not all the objects need to use IDisposable because .NET manages the objects and it is collected by the Garbage Collector. We can use the 'Dispose' method or 'using' statement to ensure that resources are released in a timely and efficient manner. It will prevent the resource leaks and improving the performance of the application.

1. What is Finalizer is used for?

* Finalizer is also called as destructor, and it is a method used for cleanup and releasing resources held by an object before it actually destroyed by the GC(Garbage Collector).
* Finalizer is defined by using the destructor syntax. It has the same name as the class name but it has tilde (~) as prefix.

1. How GC works in .net?

* GC(Garbage Collector) manages the memory in a managed application. .NET Garbage Collector is designed to manage memory, minimize the risk of memory leaks, and provide automatic memory management for developers.
* GC has some phases such as, A marking phase that finds and creates a list of all live objects. A relocating phase that updates the references to the objects that will be compacted. A compacting phase that reclaims the space occupied by the dead objects and compact the surviving objects.
* Garbage Collector uses the information to check whether objects are live or not. Stack root, Garbage collection handle and static data.

1. What is difference between .Net and .Net Core?

* .NET core is the latest version of .NET framework and it is Cross-platform, and open-source. On the other hand, .NET framework is a platform which is used to develop ASP.NET applications that executes based only on windows.
* .NET core has lot more benefits over the .NET framework such as, installation, performance, scalability, compatibility, security and so on. .Net core offers high performance and scalability and it is compatible with various operating systems such as windows, mac, linux.

1. What is difference between managed v/s unmanaged memory in .Net?

* In managed memory, there are some built in functionalities like Automatic memory management, Garbage Collection, Memory safety, Automatic Cleanup.
* On the other hand, Unmanaged memory has manual memory management, No automatic cleanup, direct memory access, and platform dependent.
* So managed memory provide automatic mamory management through garbage collector, ensuring memory safety, reducing the risk of memory related issues.

1. If program is developed in .net core in Windows then can we run the same program in Linux

* Yes, we can run a program in .NET core on both windows and Linux. One of the main advantage of the .NET core is it is a cross-platform compatibility. So we can use .NET core application on various operating systems such as Windows, Linux, MacOs.

1. If program is developed in .net core in Windows then can we run the same program in Linux without changing anything?

* Yes, we can run a program in .NET core on both windows and Linux. One of the main advantage of the .NET core is it is a cross-platform compatibility. So we can use .NET core application on various operating systems such as Windows, Linux, MacOs.

1. .Net core deployment modes?

* In .NET core there are several deployment modes available such as:
  + Framework-Dependent Deployment (FDD)
  + Self-Contained Deployment (SCD)
  + Portable-Deployment
  + Container Deployment
  + Azure functions and AWS Lambda
  + NuGet packages and Shared Libraries

1. What is extension methods and when to use?

* Extension methods in C# allow us to add new methods to existing classes without modifying their source code. These methods are defined in static classes and can be called as if they were instance methods of the extended class.
* We can use extension methods when we want to add functionality to existing types, to encapsulates behavior, for fluent and declarative APIs and to keep code clean.

public static class DemoExtensions

{

public static string CustomMethod(this string str)

{

// custom functionality here

}

}