DAY17 ASSIGNMENT
BY
PAVAN KUMAR (15-02-2022)

Q1). Research and write what is assembly in C#.

ASSEMBLY:

An Assembly is a basic building block of .Net Framework applications. It is unit of deployment like EXE or a DLL. It is completely self-describing and it is reusable.

- Assemblies are only loaded into memory if they are required. If they aren't used, they aren't loaded. This means that assemblies can be an efficient way to manage resources in larger projects.
- Assemblies are implemented as .exe or .dll files.
- There are few kinds of Assemblies. They are.

a). **PRIVATE ASSEMBLY**: (.dll and .exe are at the same place)
b). **SHARED ASSEMBLY**: (.dll and .exe are at different place)

Q2). In a tabular format write the access modifiers and explain.

Access Modifiers:

Access modifiers are used to specify the scope of accessibility of a class or type of the class itself. For example, let's consider a public class, a public class is accessible to everyone without any restrictions. Access Modifiers are of 4 types. They are.

1). PUBLIC CLASS
2). PRIVATE CLASS
3). PROTECTED CLASS
4). INTERNAL CLASS
5). DEFAULT CLASS

- In derived class we cannot access Private and Default class due to compilation error.
- Whereas in the Base class we can access all types of classes.
- Private class and Default class are same.

TABULAR FORMAT:

CLASS NAME	INSIDE ASSEMBLY			OUTSIDE ASSEMBLY	
	WITHIN CLASS	DERIVED CLASS	OTHER CLASS	DERIVED CLASS	OTHER CLASS
PUBLIC CLASS	YES	YES	YES	YES	YES
PRIVATE CLASS	YES	NO	NO	NO	NO
PROTECTED CLASS	YES	YES	NO	YES	NO
INTERNAL CLASS	YES	YES	YES	NO	NO
DEFAULT CLASS	YES	NO	NO	NO	NO
PROTECTED INTERNAL CLASS	YES	YES	NO	YES	NO

CODE:

```
using System;
using System.Collections.Generic;
namespace PavanLibrary
{
   /// <summary>
   /// DONE.BY: PAVAN
   /// PURPOSE: CREATING BASECLASS AND DERIVED CLASS
   /// </summary>
   public class MyBaseClass
   {
      public int a;
      private int b;
      protected int c;
      internal int d;
     protected internal int e;
```

```
    public void MyBaseClassMethod()
    {
      a = 10;
      b = 15;
      c = 20;
      d = 25;
      e = 30;
    }
  }
  public class MyDerivedClass : MyBaseClass
  {
    public void MyDerivedClassMethod()
    {
      a = 10;
     // b = 15;   In this case Private class cannot be accessed//
      c = 20;
      d = 25;
      e = 30;
    }
  }
  public class MyOtherClass
  {
    public void MyOtherClassMethod()
    {
      MyBaseClass bc = new MyBaseClass();
      bc.a = 10;
     //  bc.b = 15;    Private Class is not accessed
     //  bc.c = 20;    Protected Class is not accessed
      bc.d = 25;
      bc.e = 30;
    }
  }
}

using System;
using System.Collections.Generic;
using PavanLibrary;

namespace PublicLibrary
{
  /// <summary>
  /// DONE BY: PAVAN
  /// PURPOSE: INHERITING BASECLASS IN DERIVEDCLASS
  /// </summary>
```

```
    public class MyDerivedClass : MyBaseClass
    {
        public void MyDerivedClassMethod()
        {
            a = 10;
          //  b = 15; Private Class cannot be accessed//
            c = 20;
           // d = 25;  internal class can't get access//
            e = 30;
        }
    }
    public class MyDerivedOtherClass
    {
        public void MyDerivedOtherClassMethod()
        {
            MyBaseClass bc = new MyBaseClass();
            bc.a = 10;
            //  bc.b = 15; Private class can't get access
            //   bc.c = 20; Protected class can't get access
            //  bc.d = 25; internal class can't get access
            //  bc.e = 30; Protected internal class can't get access
        }
    }
}
```