**EXPERIMENT NO: 01**

**Title**: Language Introduction (Includes console-based application, creation of DLL, running a program without IDE) calling a method from another program.

**Aim:** Study of Basic of C#.Net environment.

**Theory:**

**C# Programming:**

C# is a general-purpose, modern and object-oriented programming language pronounced as “C Sharp”. It was developed by Microsoft led by Anders Hejlsberg and his team within the .NET initiative and was approved by the European Computer Manufacturers Association (ECMA) and International Standards Organization (ISO). C# is among the languages for Common Language Infrastructure. C# is a lot similar to Java syntactically and is easy for users who have knowledge of C, C++ or Java.

**C# Features**

**Simple**

C# is a user-friendly language that offers a structured approach to problem-solving. it provides a wide range of library functions and data types to work.

**Modern Programming Language**

C# programming is a popular and powerful language that is for creating scalable, interoperable, and robust applications.

**Object Oriented**

C# is an object-oriented programming language, which makes development and maintenance easier. In contrast, with procedure-oriented programming languages, managing code becomes difficult as project size grows.

**Type Safe**

The code is type safe can only access memory locations that it has permission to execute. This feature significantly enhances program security.

**Scalable and Updateable**

C# is a programming language that is scalable and can be updated automatically. To update our application, we remove the old files and replace them with new ones.

**Component Oriented**

It is widely used as a software development methodology to create applications that are more strong and can easily scale.

**Structured Programming Language**

C# is a structured programming language that allows us to divide programs into parts using functions, making it easy to understand and modify.

**Fast Speed**

The compilation and execution time of C# language is fast.

**.Net Framework:**

**.Net Framework** is a software development platform developed by Microsoft for building and running Windows applications. The .Net framework consists of developer tools, programming languages, and libraries to build desktop and web applications. It is also used to build websites, web services, and games.

The Microsoft .Net framework can be used to create both – **Form-based**and**Web-based** applications.[Web services](https://www.guru99.com/web-services-tutorial.html)can also be developed using the .Net framework. The framework also supports various programming languages such as Visual Basic and C#. So developers can choose and select the language to develop the required application.

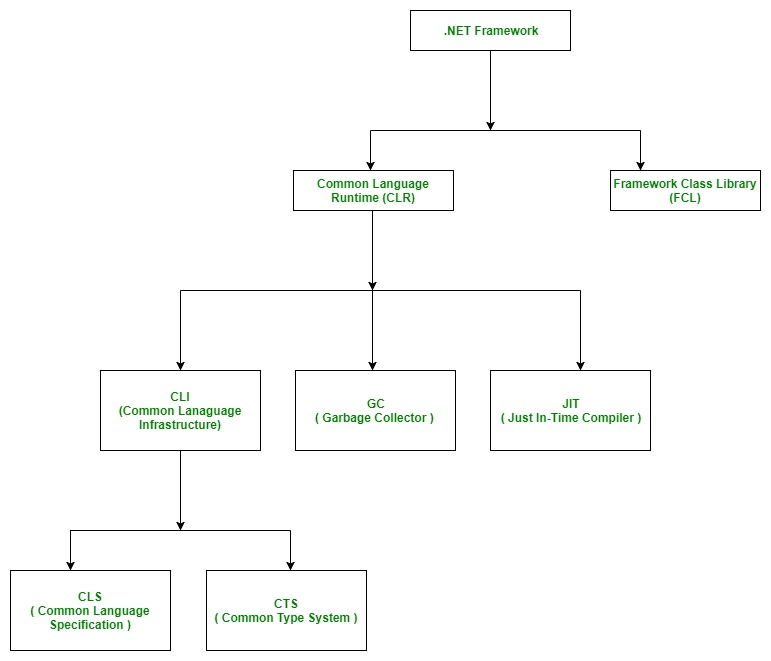
**11 Programming Languages which are designed and developed by Microsoft are:**

1. C#.NET
2. VB.NET
3. C++.NET
4. J#.NET
5. F#.NET
6. JSCRIPT.NET
7. WINDOWS POWERSHELL
8. IRON RUBY
9. IRON PYTHON
10. C OMEGA
11. ASML(Abstract State Machine Language)

**Main Components of .NET Framework:**

**Common Language Runtime(CLR):** CLR is the basic and Virtual Machine component of the .NET Framework. It is the run-time environment in the .NET Framework that runs the codes and helps in making the development process easier by providing various services such as remoting, thread management, type-safety, memory management, robustness, etc.. Basically, it is responsible for managing the execution of .NET programs regardless of any .NET programming language. It also helps in the management of code, as code that targets the runtime is known as the Managed Code, and code that doesn’t target to runtime is known as Unmanaged code.

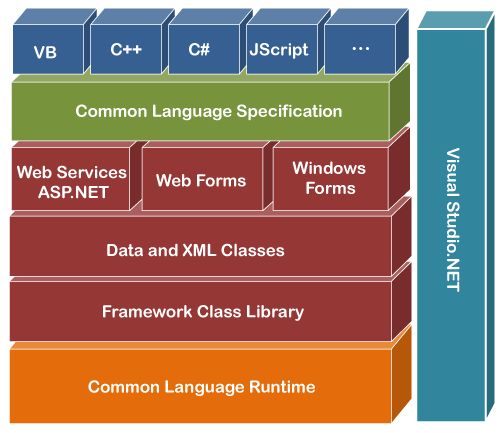
**Framework Class Library(FCL):** It is the collection of reusable, object-oriented class libraries and methods, etc that can be integrated with CLR. Also called the Assemblies. It is just like the header files in C/C++ and packages in java. Installing the .NET framework basically is the installation of CLR and FCL into the system. Below is the overview of the .NET Framework.



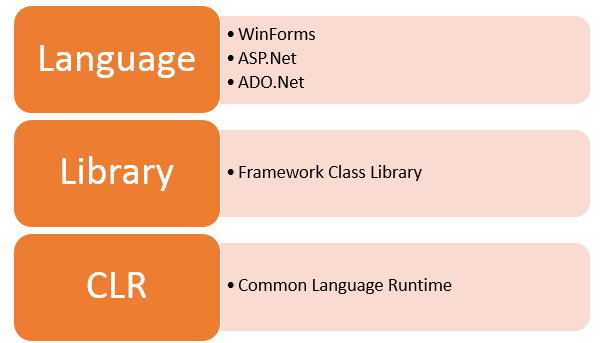
The combination of Operating System Architecture and CPU Architecture is known as the platform. Platform dependent means the programming language code will run only on particular Operating System. A .NET application is platform-dependent because of the .NET framework which is only able to run on the Windows-based operating system. The .Net application is platform-independent also because of the Mono framework. Using the Mono framework the .Net application can run on any Operating System including windows. Mono framework is a third-party software developed by Novell Company which is now a part of Micro Focus Company. It is a paid framework.

**Basic Architecture and Component Stack of .NET Framework:**

The first three components from bottom are considered as the basic architecture of .Net framework which came in the year 2005 and after this more components were added by Microsoft in the .Net Framework as following :



The basic architecture of the .Net framework is as shown below.

[](https://www.guru99.com/images/c-sharp-net/052416_1343_WhatisNETFr1.png)

**1. CLR (Common Language Runtime) :** It is a run-time environment which executes the code written in any .NET programming language. .Net framework provides the support for many languages like C#, F#, C++, Cobra, Jscript.Net, VB.Net, Oxygene etc

**2. FCL (Framework Class Library) :** A large number of class libraries are present in this framework which is known as FCL.

**3. Types of Applications :** Mainly the applications which are built in .Net framework is divided into the following three categories :

**WinForms :** Form – Based applications are considered under this category. In simple terms, we can say client based applications which read and writes the file system comes under this category.

**ASP .NET :** Web-Based applications come under this category. ASP.Net is a framework for web and it provides the awesome integration of HTML, CSS and JavaScript which makes it useful to develop the web applications, websites and web services. Web services were added in .Net Framework 2.0 and considered as a part of ASP.NET web applications.

**ADO .NET :** It includes the application which are developed to communicate with the database like MS SQL Server, Oracle etc. comes. It mainly consists of classes that can be used to connect, retrieve, insert and delete data.

**4. WPF (Windows Presentation Foundation) :** Windows Presentation Foundation (WPF) is a graphical subsystem given by Microsoft which uses DirectX and is used in Windows-based applications for rendering UI (User Interface). WPF was initially released as part of .NET Framework 3.0 in 2006 and previously known as “Avalon”.

**5. WCF (Windows Communication Foundation) :** It is a framework for building connected and service-oriented applications used to transmit the data as asynchronous from one service endpoint to another service point. It was previously known as the Indigo.

**6. WF (Windows Workflow Foundation) :** It is a technology given by Microsoft which provides a platform for building workflows within .Net applications.

**7. Card Space :** It is a Microsoft .NET Framework software client which is designed to let users provide their digital identity to online services in a secure, simple and trusted way.

**8. LINQ (Language Integrated Query) :** It is introduced in .Net framework version 3.5. Basically, it is a query language used to make the query for data sources with VB or C# programming languages.

**9. Entity Framework :** It is open–source ORM (Object Relational Mapping) based framework which comes into .Net Framework version 3.5. It enables the .Net developer to work with database using .Net objects. Before entity framework, .Net developers have performed a lot of things related database. Like to open a connection to the database, developers have to create a Data Set to fetch or submit the data to the database, convert data from the Data Set to .NET objects or vice-versa. It creates the difficulties for developers and also it was the error-prone process, then “Entity Framework” comes to automate all these database related activities for the application. So, Entity Framework allows the developers to work at a higher level of abstraction.

**REST (Representational State Transfer) and AJAX** were added in .Net Framework 3.5 as an extension and services of ASP.NET for enhancing web services of .NET Framework.

**10. Parallel LINQ (Language Integrated Query) :** It comes in .Net Framework version 4.0 and also termed as PLINQ. It provides a concurrent query execution engine for LINQ. It executes the LINQ in parallel such that it tries to use as much processing power system on which it is executing.

**11. TPL (Task Parallel Library) :** It is a set of public types and APIs. It allows the developers to be more productive by simplifying the process of adding concurrency and parallelism to .Net applications.

**12. .NET API For Store/UWP Apps :** In 2012, Microsoft added some APIs for creating UWP(Universal Windows Platform) apps for Windows using C# or VB.

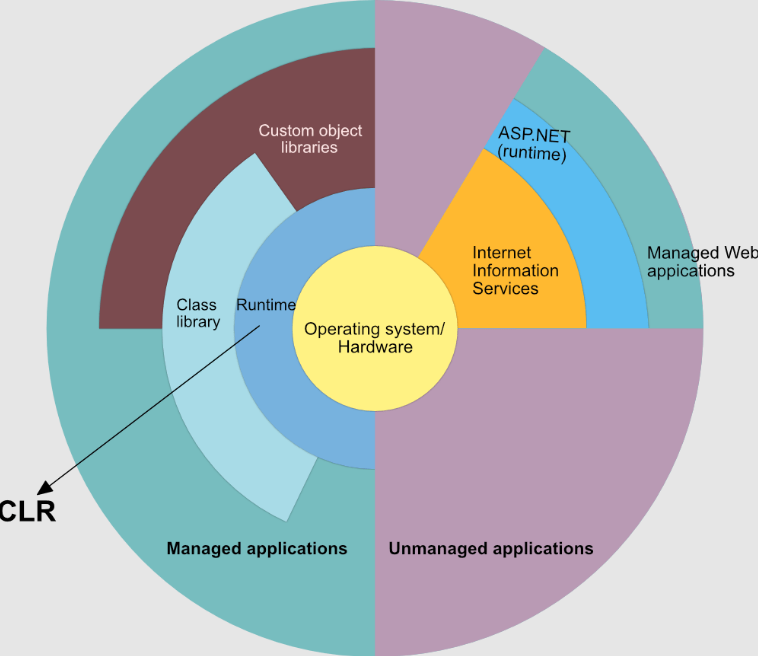
**13. Task-Based Asynchronous Model :** It is model used to describe the asynchronous operations and tasks in .Net Framework.

**CLR (Common Language Runtime):**

CLR is the basic and Virtual Machine component of the .NET Framework. It is the run-time environment in the .NET Framework that runs the codes and helps in making the development process easier by providing the various services. Basically, it is responsible for managing the execution of .NET programs regardless of any .NET programming language. Internally, CLR implements the VES(Virtual Execution System) which is defined in the Microsoft’s implementation of the CLI(Common Language Infrastructure).

The code that runs under the Common Language Runtime is termed as the Managed Code. In other words, you can say that CLR provides a managed execution environment for the .NET programs by improving the security, including the cross language integration and a rich set of class libraries, etc. CLR is present in every .NET framework version. Below table illustrate the CLR version in .NET framework.

Below diagram illustrate how CLR is associated with the operating system/hardware along with the class libraries. Here, the runtime is actually CLR.

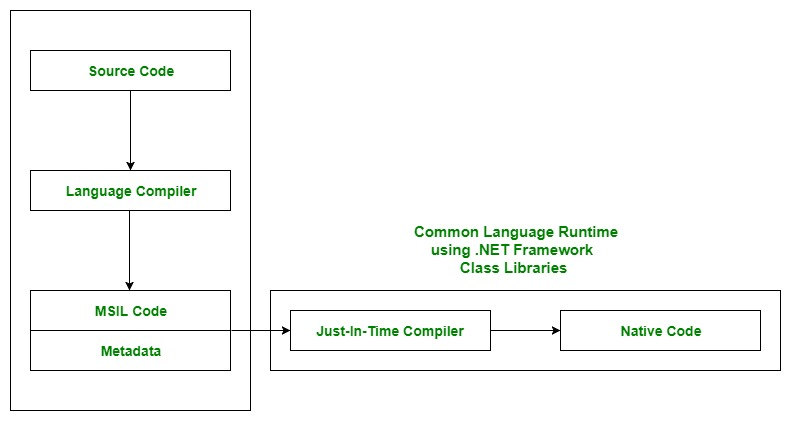


**Role of CLR in the execution of a C# program:**

Suppose you have written a C# program and save it in a file which is known as the Source Code.

Language specific compiler compiles the source code into the MSIL(Microsoft Intermediate Language) which is also known as the CIL(Common Intermediate Language) or IL(Intermediate Language) along with its metadata. Metadata includes all the types, actual implementation of each function of the program. MSIL is machine-independent code.

Now CLR comes into existence. CLR provides the services and runtime environment to the MSIL code. Internally CLR includes the JIT(Just-In-Time) compiler which converts the MSIL code to machine code which further executed by CPU. CLR also uses the .NET Framework class libraries. Metadata provides information about the programming language, environment, version, and class libraries to the CLR by which CLR handles the MSIL code. As CLR is common so it allows an instance of a class that written in a different language to call a method of the class which written in another language.



**Main Components of CLR:**

As the word specify, Common means CLR provides a common runtime or execution environment as there are more than 60 .NET programming languages.

* **Common Language Specification (CLS):**

It is responsible for converting the different .NET programming language syntactical rules and regulations into CLR understandable format. Basically, it provides Language Interoperability. Language Interoperability means providing execution support to other programming languages also in .NET framework.

Language Interoperability can be achieved in two ways :

**Managed Code:** The MSIL code which is managed by the CLR is known as the Managed Code. For managed code CLR provides three .NET facilities:

**Unmanaged Code:** Before .NET development, programming languages like.COM Components & Win32 API do not generate the MSIL code. So these are not managed by CLR rather managed by Operating System.

* **Common Type System (CTS):**

Every programming language has its own data type system, so CTS is responsible for understanding all the data type systems of .NET programming languages and converting them into CLR understandable format which will be a common format.

There are 2 Types of CTS that every .NET programming language have :

**Value Types:** Value Types will store the value directly into the memory location. These types work with stack mechanisms only. CLR allows memory for these at Compile Time.

**Reference Types:** Reference Types will contain a memory address of value because the reference types won’t store the variable value directly in memory. These types work with Heap mechanism. CLR allot memory for these at Runtime.

* **Garbage Collector:**

It is used to provide the Automatic Memory Management feature. If there was no garbage collector, programmers would have to write the memory management codes which will be a kind of overhead on programmers.

* **JIT(Just In Time Compiler):**

It is responsible for converting the CIL(Common Intermediate Language) into machine code or native code using the Common Language Runtime environment.

**Benefits of CLR:**

* It improves the performance by providing a rich interact between programs at run time.
* Enhance portability by removing the need of recompiling a program on any operating system that supports it.
* Security also increases as it analyzes the MSIL instructions whether they are safe or unsafe. Also, the use of delegates in place of function pointers enhance the type safety and security.
* Support automatic memory management with the help of Garbage Collector.
* Provides cross-language integration because CTS inside CLR provides a common standard that activates the different languages to extend and share each other’s libraries.
* Provides support to use the components that developed in other .NET programming languages.
* Provide language, platform, and architecture independence.
* It allows easy creation of scalable and multithreaded applications, as the developer has no need to think about memory management and security issues.

**Sample Program.cs:**

// C# program to print Hello World!

using System;

// namespace declaration

namespace HelloWorldApp {

// Class declaration

class SampleProgram {

// Main Method

static void Main(string[] args) {

// statement

// printing Hello World!

Console.WriteLine("Hello World!");

// To prevents the screen from

// running and closing quickly

Console.ReadKey();

}

}

}

**Output:**

Hello World!

**Explanation:**

**using System:** System is a namespace which contains the commonly used types. It is specified with a using System directive.

**namespace HelloWorldApp:** Here namespace is the keyword which is used to define the namespace. HelloWorldApp is the user-defined name given to namespace.

**class SampleProgram:** Here class is the keyword which is used for the declaration of classes. SampleProgram is the user-defined name of the class.

**static void Main(string[] args):** Here static keyword tells us that this method is accessible without instantiating the class. void keyword tells that this method will not return anything. Main() method is the entry point of our application. In our program, Main() method specifies its behavior with the statement Console.WriteLine(“Hello World!”);.

**Console.WriteLine():** Here WriteLine() is a method of the Console class defined in the System namespace.

**Console.ReadKey():** This is for the VS.NET Users. This makes the program wait for a key press and prevents the screen from running and closing quickly.

**Using Command-Line to run C# Program:** Below steps demonstrate how to run a C# program on Command line in Windows Operating System:

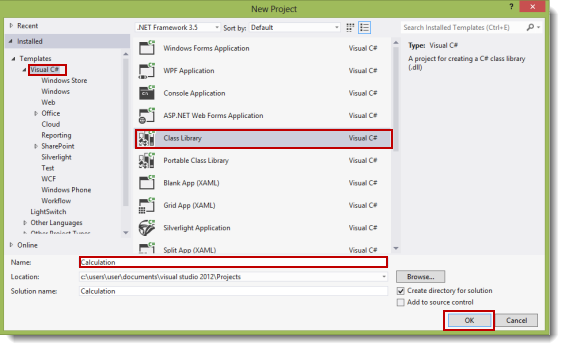
* First, open a text editor like Notepad or Notepad++.
* Write the code in the text editor and save the file with .cs extension.
* Open the cmd(Command Prompt) and run the command csc to check for the compiler version. It specifies whether you have installed a valid compiler or not. You can avoid this step if you confirmed that compiler is installed.
* To compile the code type csc filename.cs on cmd. If your program has no error then it will create a filename.exe file in the same directory where you have saved your program. Suppose you saved the above program as hello.cs. So you will write csc hello.cs on cmd. This will create a hello.exe.
* Now you have to ways to execute the hello.exe. First, you have to simply type the filename i.e hello on the cmd and it will give the output. Second, you can go to the directory where you saved your program and there you find filename.exe. You have to simply double-click that file and it will give the output.

**Dynamic Link library (DLL):**

A Dynamic Link library (DLL) is a library that contains functions and codes that can be used by more than one program at a time. Once we have created a DLL file, we can use it in many applications. The only thing we need to do is to add the reference/import the DLL File. Both DLL and .exe files are executable program modules but the difference is that we cannot execute DLL files directly.

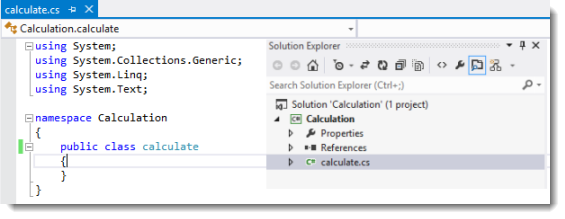
**Creating DLL File:**

**Step 1 -** Open Visual Studio then select "File" -> "New" -> "Project..." then seelct "Visual C#" -> "Class library".

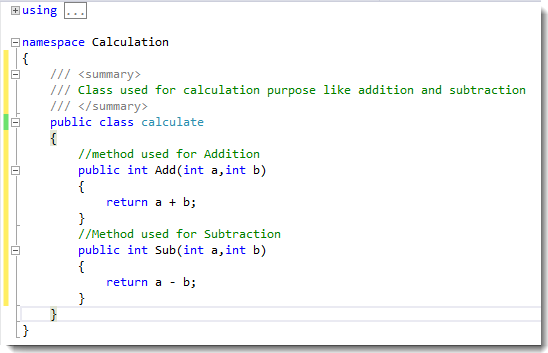


(I give it the name "Calculation".)

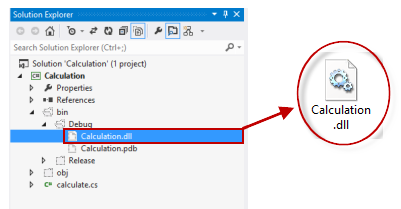
**Step 2 -** Change the class name ("class1.cs") to "calculate.cs".



**Step 3 -** In the calculate class, write methods for the addition and subtraction of two integers (for example purposes).



**Step 4 -** Build the solution (F6). If the build is successful then you will see a "calculation.dll" file in the "bin/debug" directory of your project.

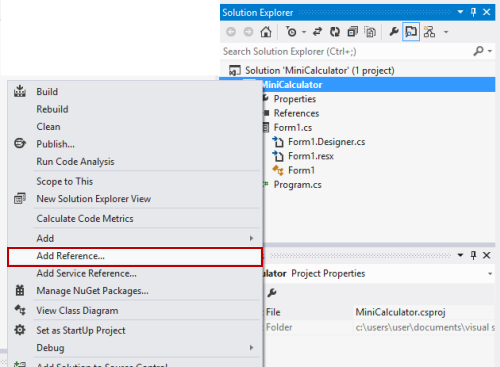


We have created our DLL file. Now we will use it in another application.

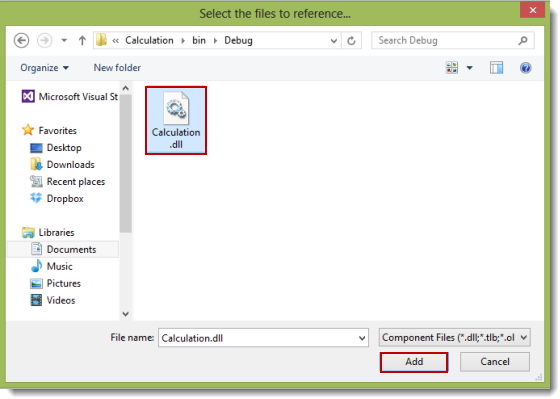
**Using DLL File:**

**Step 1 -** Open Visual Studio then select "File" -> "New" -> "Project..." then select "Visual C#" -> "Console Based application".

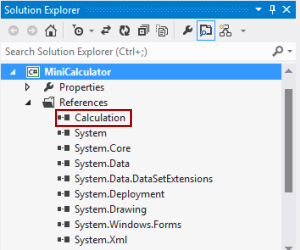
**Step 2 -** Add a reference for the dll file, "calculation.dll", that we created earlier. Right-click on the project and then click on "Add reference".



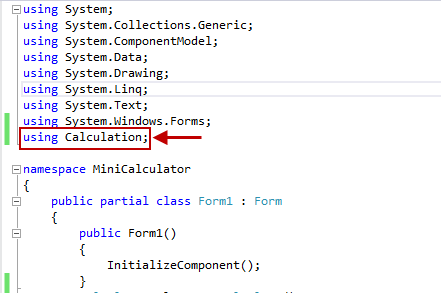
**Step 3 -** Select the DLL file and add it to the project.



After adding the file, you will see that the calculation namespace has been added (in references) as in the following:



**Step 4 -** Add the namespace ("using calculation;") as in the following:



**Step 5-Sample Program use DLL File:**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace MathLibrary

{

public class Class1

{

public float Substract(float a, float b)

{

return a - b;

}

public float Multiply(float a, float b)

{

return a \* b;

}

public float Devide(float a, float b)

{

return a / b;

}

public float Power(float a)

{

return a \* a;

}

}

}

Create Another Projecct and import the MathLibrary

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using MathLibrary;

namespace ConsoleApplication1

{

class Program

{

static void Main(string[] args)

{

// Declare class from MathLibrary.dll

Class1 math = new Class1();

// Declare variables and use methods from MathLibrary

Console.WriteLine("Enter First Number");

int a = Convert.ToInt32(Console.ReadLine());

Console.WriteLine("Enter second Number");

int b= Convert.ToInt32(Console.ReadLine());

float substract = math.Substract(a,b);

float multiply = math.Multiply(a,b);

float devide = math.Devide(a,b);

float power = math.Power(a);

Console.WriteLine("This Application uses functions from MathLibrary.dll todo simple calculations");

// Print results on screen

Console.WriteLine("Substraction:"+substract);

Console.WriteLine("Multilication:"+multiply);

Console.WriteLine("Dividation:"+devide);

Console.WriteLine("Power of a given Number:"+power);

Console.ReadLine();

}

}

}

**Problem Statement:**

Sport Competitions Slot

**Conclusion:**

**Sample Questions:**

1) What is C#?

2) Explain .Net Architecture?

3) Explain CLR?

4) What is DLL?