**Beyond Syllabus**

Practical:

Create and run console application using C#

Steps:

1. Start Visual Studio.
2. On the menu bar, choose **File**, **New**, **Project**.

The **New Project** dialog box opens.

1. Expand **Installed**, expand **Templates**, expand **Visual C#**, and then choose **Console Application**.
2. In the **Name** box, specify a name for your project, and then choose the **OK** button.

The new project appears in **Solution Explorer**.

1. If Program.cs isn't open in the **Code Editor**, open the shortcut menu for **Program.cs** in **Solution Explorer**, and then choose **View Code**.
2. Replace the contents of Program.cs with the following code.

**Program**

// A Hello World! program in C#.

using System;

namespace HelloWorld

{

class Hello

{

static void Main()

{

Console.WriteLine("Hello World!");

// Keep the console window open in debug mode.

Console.WriteLine("Press any key to exit.");

Console.ReadKey();

}

}

}

1. Choose the F5 key to run the project. A Command Prompt window appears that contains the line Hello World!

Next, the important parts of this program are examined.

**Comments**

**The first line contains a comment. The characters // convert the** rest of the line to a comment.

C#Copy

// A Hello World! program in C#.

You can also comment out a block of text by enclosing it between the /\* and \*/ characters. This is shown in the following example.

C#Copy

/\* A "Hello World!" program in C#.

This program displays the string "Hello World!" on the screen. \*/

**Main Method**

A C# console application must contain a Main method, in which control starts and ends. The Main method is where you create objects and execute other methods.

The Main method is a [static](https://docs.microsoft.com/en-us/dotnet/csharp/language-reference/keywords/static) method that resides inside a class or a struct. In the previous "Hello World!" example, it resides in a class named Hello. You can declare the Main method in one of the following ways:

* It can return void.

C#Copy

static void Main()

{

//...

}

* It can also return an integer.

C#Copy

static int Main()

{

//...

return 0;

}

* With either of the return types, it can take arguments.

C#Copy

static void Main(string[] args)

{

//...

}

-or-

C#Copy

static int Main(string[] args)

{

//...

return 0;

}

**Input and Output**

C# programs generally use the input/output services provided by the run-time library of the .NET Framework. The statement System.Console.WriteLine("Hello World!"); uses the [WriteLine](https://docs.microsoft.com/en-us/dotnet/api/system.console.writeline) method. This is one of the output methods of the [Console](https://docs.microsoft.com/en-us/dotnet/api/system.console) class in the run-time library. It displays its string parameter on the standard output stream followed by a new line. Other [Console](https://docs.microsoft.com/en-us/dotnet/api/system.console) methods are available for different input and output operations. If you include the using System; directive at the beginning of the program, you can directly use the [System](https://docs.microsoft.com/en-us/dotnet/api/system) classes and methods without fully qualifying them. For example, you can call Console.WriteLine instead of System.Console.WriteLine:

C#Copy

using System;

C#Copy

Console.WriteLine("Hello World!");