

Q1: Developing a basic console calculator

This program accepts two numbers as input via the ReadLine() function. ReadLine() only accepts input in string format, so to perform calculations with real numbers, we need to utilize data types like double or float.

To convert a string to a double, we can use the ToDouble() function, while the ToSingle() function is used for converting to a float.

Next, we need to request another input, which is the operator, to determine the type of operation we wish to execute.

If the operator input is anything other than +, -, *, or /, the program will display an error message: "Invalid Operator!"

Q2: Delegate in C#

Syntax:

```
delegate <return type> <delegate-name> <parameter list>;
```

A delegate can refer to a method, which have the same signature as that of the delegate.

Steps:

- Create reference for the delegate
- Create object for the delegate and the parameter is name of the desired methods
- Assign object to reference
- Call the method through the reference

```

namespace Demo
{
    0 references
    public class Lab1
    {
        // Create delegate
        delegate void Del(String s);
        // Create a method for a delegate.
        1 reference
        public static void DelegateMethod(string message)
        {
            System.Console.WriteLine(message);
        }
        0 references
        public static void Main()
        {
            // Instantiate the delegate.
            Del handler = DelegateMethod;
            // Call the delegate.
            handler("Hello World");
        }
    }
}

```

Events

Definition:

- An event is automatic notification when some action has occurred.
- An object that has an interest in an event, registers an event handler for that event.
- When the event occurs, **all registered handlers are called**.
- Event handlers are represented by delegates.
- Events are members of the class and are declared using the

keyword 'event'. General Form:

event *event-delegate* *object*;

event-delegate --> Name of the delegate used to support *the event*

object --> Name of the specific event object.

```

namespace SimpleEvent
{
    class Program
    {
        public delegate void SimpleDelegateHandler(string msg);
        public event SimpleDelegateHandler myEvent;

        public void run()
        {
            //Register with event
            myEvent += myMethod1;
            myEvent += myMethod2;

            //Make call to methods
            myEvent("Hello Event!");
        }


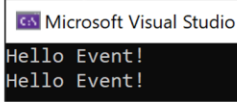
        static void myMethod1(String msg)
        {
            Console.WriteLine(msg);
        }
        static void myMethod2(String msg)
        {
            Console.WriteLine(msg);
        }
    }
}

```

```

static void Main(string[] args)
{
    new Program().run();
}

```

Q2.1:

Write a C# program to demonstrate three ways using Delegates in C#

Q2.2:

Write a C# program to find sum of 'n' numbers from 1 to 'n' using delegates

Steps:

1. Create a delegate
2. Create a class 'test' and use methods Read(), Calc(), and Show() to get the number, calculate sum 1 to n and to display the result.
3. Inside the main method class [use the delegate to call the method Calc\(\)](#) of the 'test' class

Q2.3:

Triggering multiple events using 'multicasting events' Steps:

1. Create a delegate 'mydel'
2. Create the class Class1 and inside that create an event handler method call1()
3. Create the class Class2 and inside that create an event handler method call2()
4. Create event "myEvent" then trigger handler method call1() and call2());

