**Constructor Overloading**

* Declaring more than one constructor in a class is called constructor overloading.
* The process of overloading constructors is similar to overloading methods where every constructor has a signature similar to that of a method.
* Multiple constructors in a class can be declared wherein each constructor will have different signatures.
* Constructor overloading is used when different objects of the class might want to use different initialized values.
* Overloaded constructors reduce the task of assigning different values to member variables each time when needed by different objects of the class.

**The following code demonstrates the use of constructor overloading:**

using System;

public class Rectangle

{

double \_length;

double \_breadth;

public Rectangle()

{

}

\_length = 13.5;

\_breadth = 20.5;

}

public Rectangle(double len, double wide)

{

\_length = len;

\_breadth = wide;

}

public double Area()

{

return \_length \* \_breadth;

}

static void Main(string[] args)

{

Rectangle objRect1 = new Rectangle();

Console.WriteLine(“Area of rectangle = “ + objRect1.Area());

Rectangle objRect2 = new Rectangle(2.5, 6.9);

Console.WriteLine(“Area of rectangle = “ + objRect2.Area());

}

}

**In Above Code,**

* Two constructors are created having the same name, Rectangle.
* However, the signatures of these constructors are different. Hence, while calling the method Area()from the Main() method, the parameters passed to the calling method are identified.
* Then, the corresponding constructor is used to initialize the variables \_length and \_breadth. Finally, the multiplication operation is performed on these variables and the area values are displayed as the output.

**Output**

Area of rectangle1 = 276.75  
Area of rectangle2 = 17.25

**Source Code of Constructor Overloading**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace ConstructorOverloading

{

class Program

{

public Program()

{

Console.WriteLine("this is a first constructor !!");

}

public Program(int a, int b)

{

Console.WriteLine("this is a Second constructor !! {0}", (a+b));

}

public Program(int a, int b, int c)

{

Console.WriteLine("this is a third constructor !! {0}", (a + b +c));

}

public Program(string a, string b, string c)

{

Console.WriteLine("this is a fourth constructor !! {0}", (a + b + c));

}

static void Main(string[] args)

{

// CONSTRUCTOR OVERLOADING

Program p = new Program("A","B","C");

Console.ReadLine();

}

}

}