CONSTRUCTORS

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Intermediate\_7\_Constructors

{

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* NOTES

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

//================================

// WHAT IS A CONSTRUCTOR?

//================================

// A Constructor is a method that is called when an instance of a class is created.

//================================

// WHY DO We NEED A CONSTRUCTOR?

//================================

// A Constructor is used to place an Object in an 'EARLY STATE'. That means to initialize

// fields or perform methods during the creation of an instance of a Class.

//================================

// SYNTAX OF A CONSTRUCTOR

//================================

// The Constructor Name needs to have the same name as the Class.

// Constructors do not have a Return type.

// A PARAMETERLESS CONSTRUCTOR

//================================

/\*

public class Customer

{

public Customer()

{

// Code goes in here

}

}

\*/

// A CONSTRUCTOR WITH PARAMETER

//================================

/\*

public class Customer

{

public string Name;

public Customer(string name)

{

// Code goes in here

this.Name = name

}

}

\*/

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// "THIS" KEY WORD

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// The 'This' keyword that references the current object.

// The 'This' keyword helps developers to easily identify

// a property that belongs to the current object.

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// CONSTRUCTOR OVERLOADING

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// A Constructor can be Overloaded.

// 'Overloading' is when you have a class Method with the same name, but with a different SIGNATURE.

// A SIGNATURE is what uniquely identifies a method. That is it's:

// - Return Type

// = It's Name

// - Type and Number of its parameters

class Program

{

static void Main(string[] args)

{

var customer = new Customer(1, "John");

Console.WriteLine(customer.Id);

Console.WriteLine(customer.Name);

}

}

}

#######################################################################################################################################

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Intermediate\_7\_Constructors

{

public class Customer

{

// Declare some fields/properties

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// IMPORTANT NOTE

//

// You only create Constructors when it is

// really important to initialize a property

// that needs to be initialized, because if

// you overload too many Constructors then

// the Code could look messy and cluttered.

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

//===============

// EXAMPLE

//===============

// An Example of a Class that needs to be set to an Early State

// upon Initialization.

// In this example, whenever you have a Class that has fields/properties

// that are a List of objects (such as the List<Order> property that belongs to the

// Customer Class) you need to always initialize the List Object to an EMPTY

// Property Value (Not a NULL value. There is a difference between an Empty

// List value and a NULL value).

public int Id;

public string Name;

public List<Order> Orders;

public Customer()

{

this.Orders = new List<Order>();

}

public Customer(int id)

: this() //<---- Include : this() means that before this Constructor is called

// the Constructor with no parameters is called first

{

this.Id = id;

}

public Customer(int id, string name)

: this(id) //<---- Include : this(id) means that before this Constructor is called

// the Constructor with one Parameter of 'int id' is called first

{

this.Id = id;

this.Name = name;

}

}

}

#######################################################################################################################################

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Intermediate\_7\_Constructors

{

public class Order

{

}

}