
C# INTERMEDIATE
Constructors

What?

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

Why?

To put an object in an early state.

How?

```
public class Customer
{
    public Customer()
    {
    }
}
```

Constructors

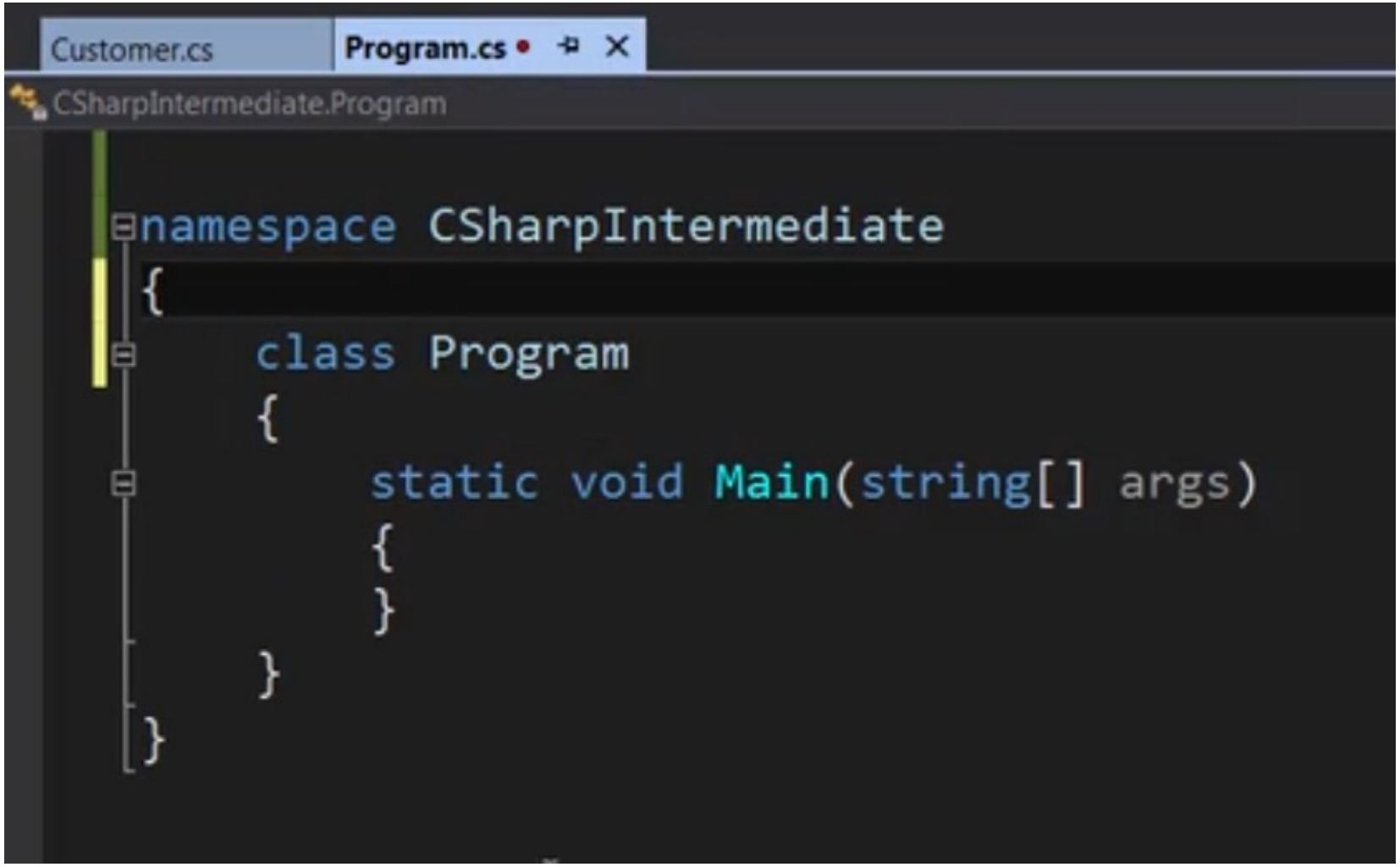
```
public class Customer
{
    public string Name;

    public Customer(string name)
    {
        this.Name = name;
    }
}
```

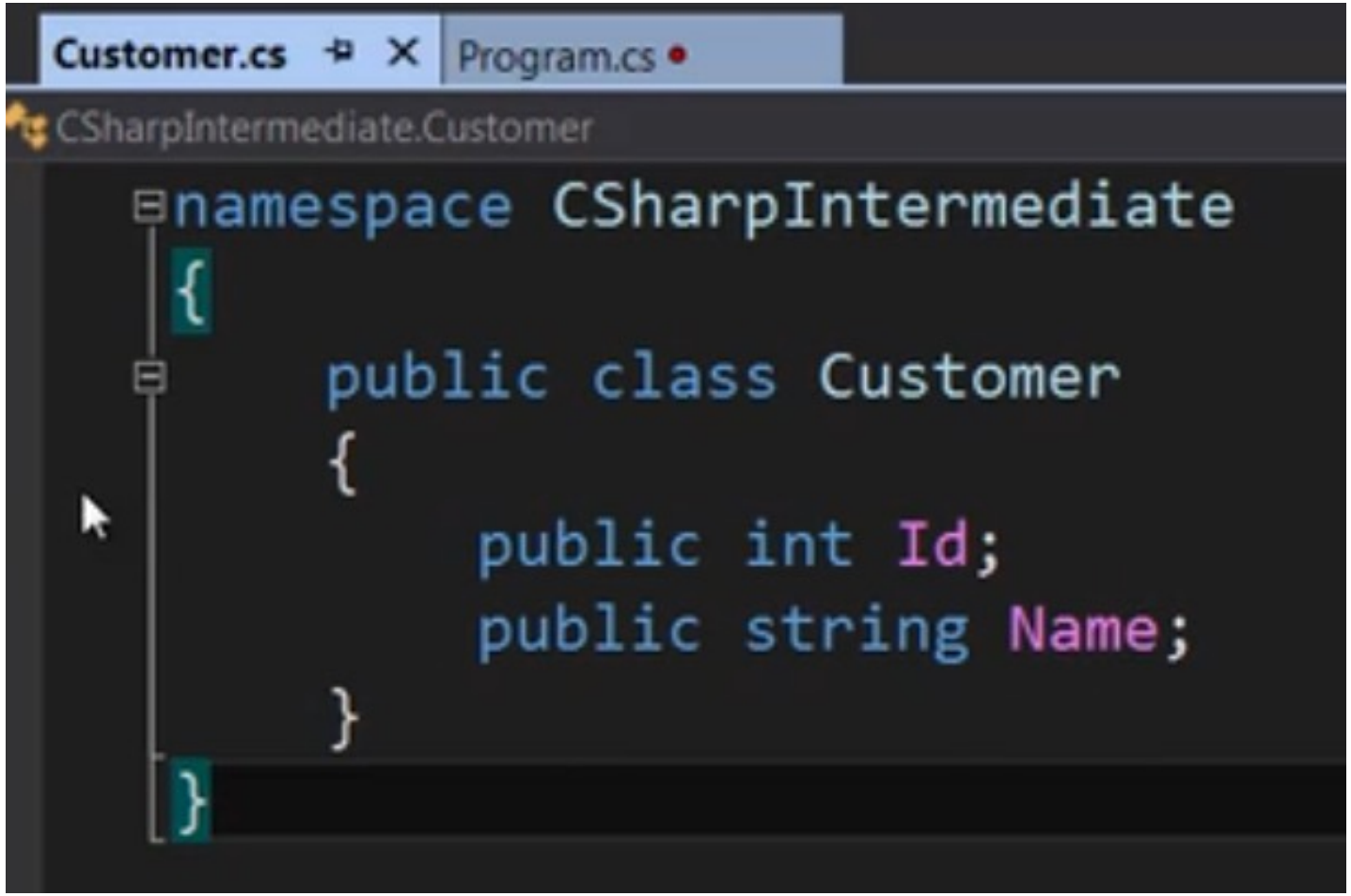
Constructors

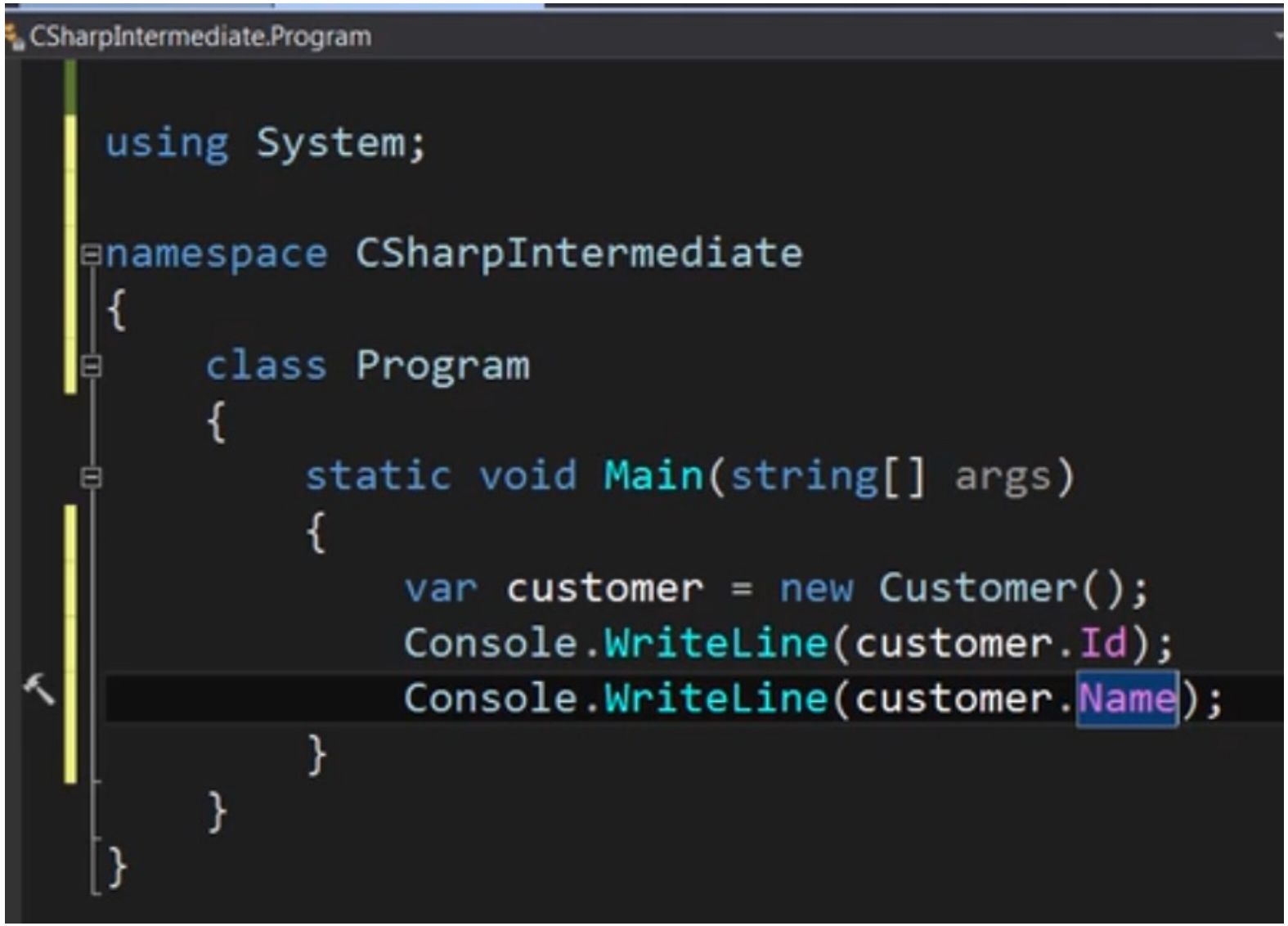
```
var customer = new Customer("John");
```

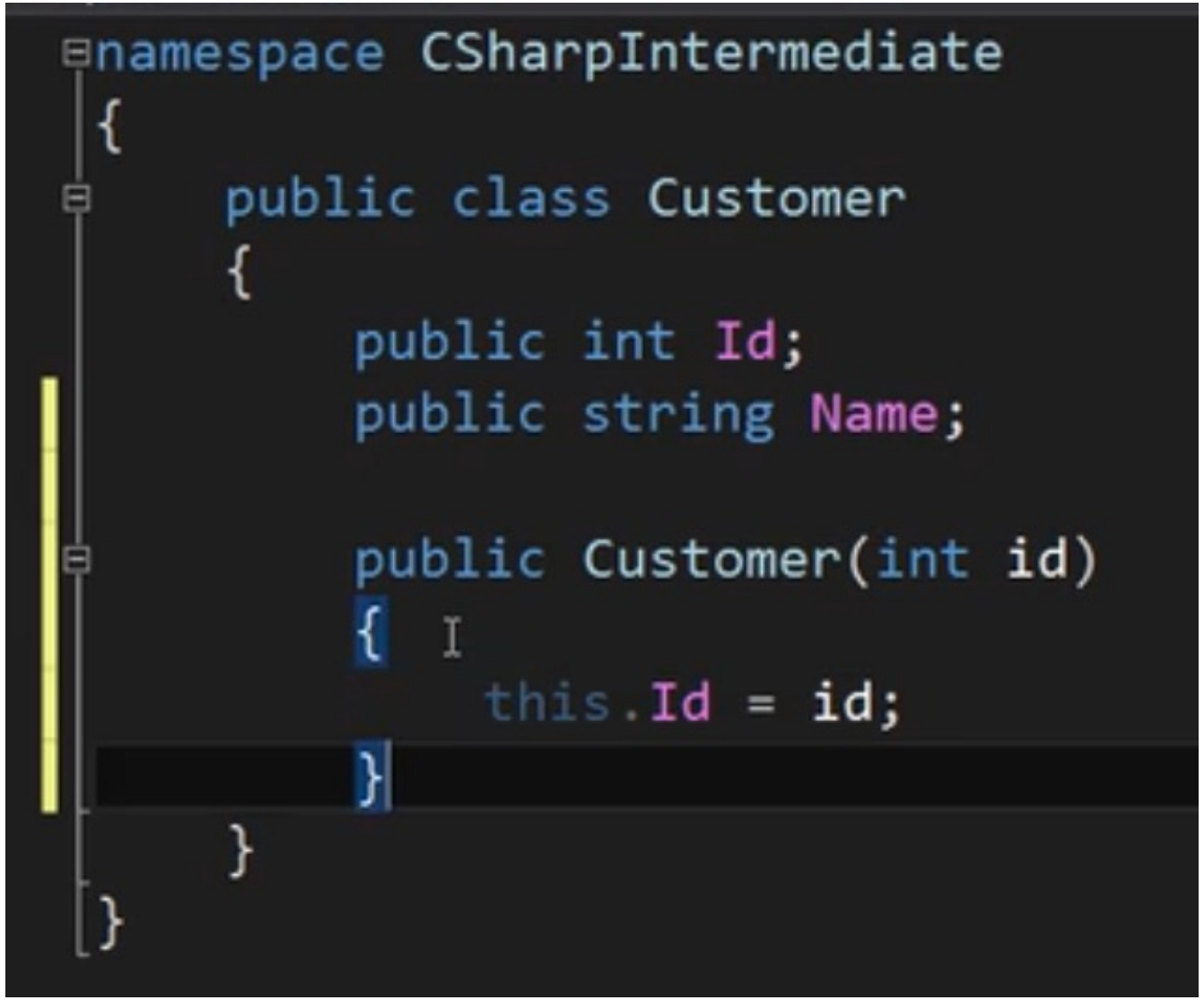
Constructor Overloading

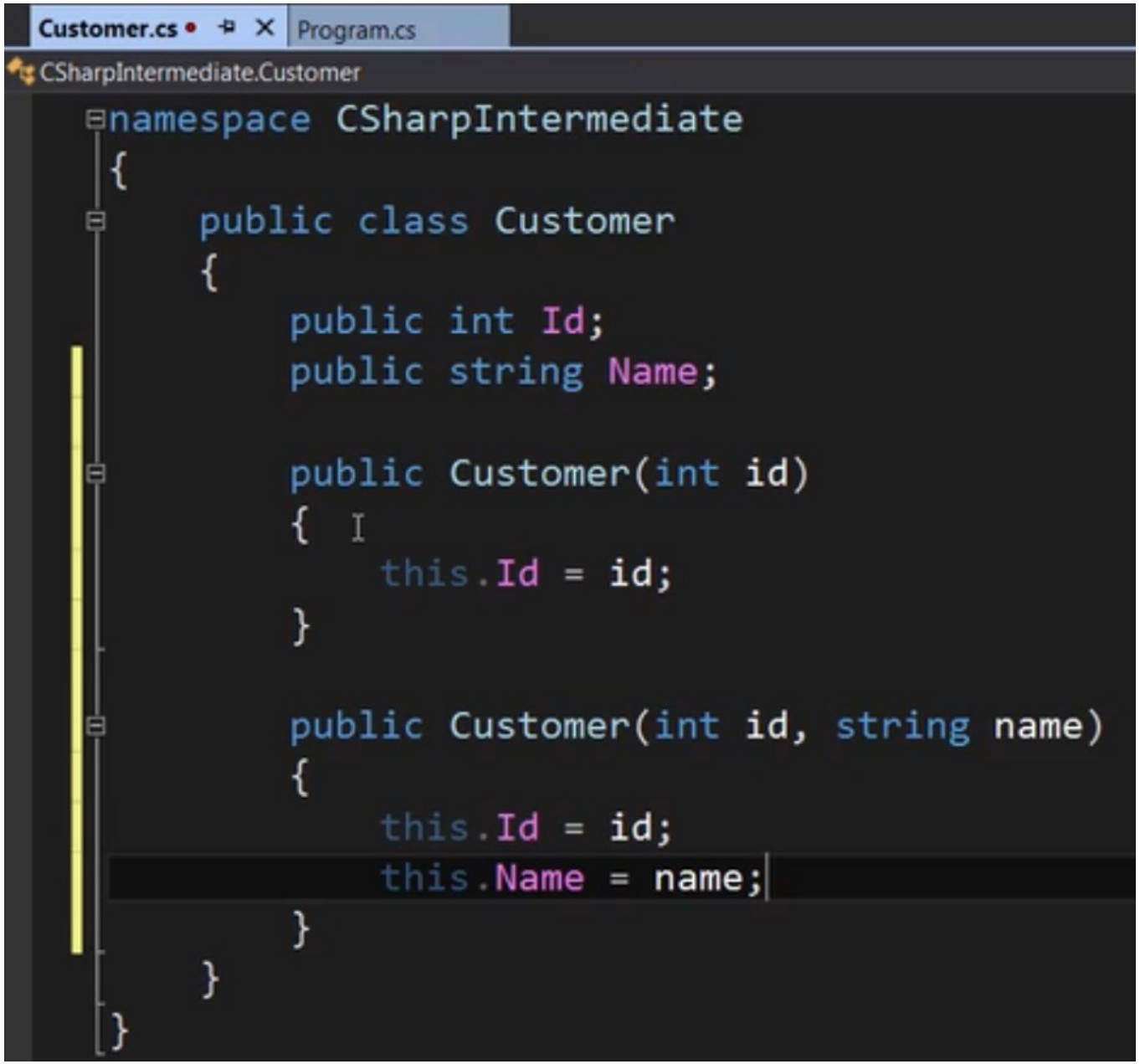
```
public class Customer
{
    public Customer() { … }

    public Customer(string name) { … }

    public Customer(int id, string name) { … }
}
```

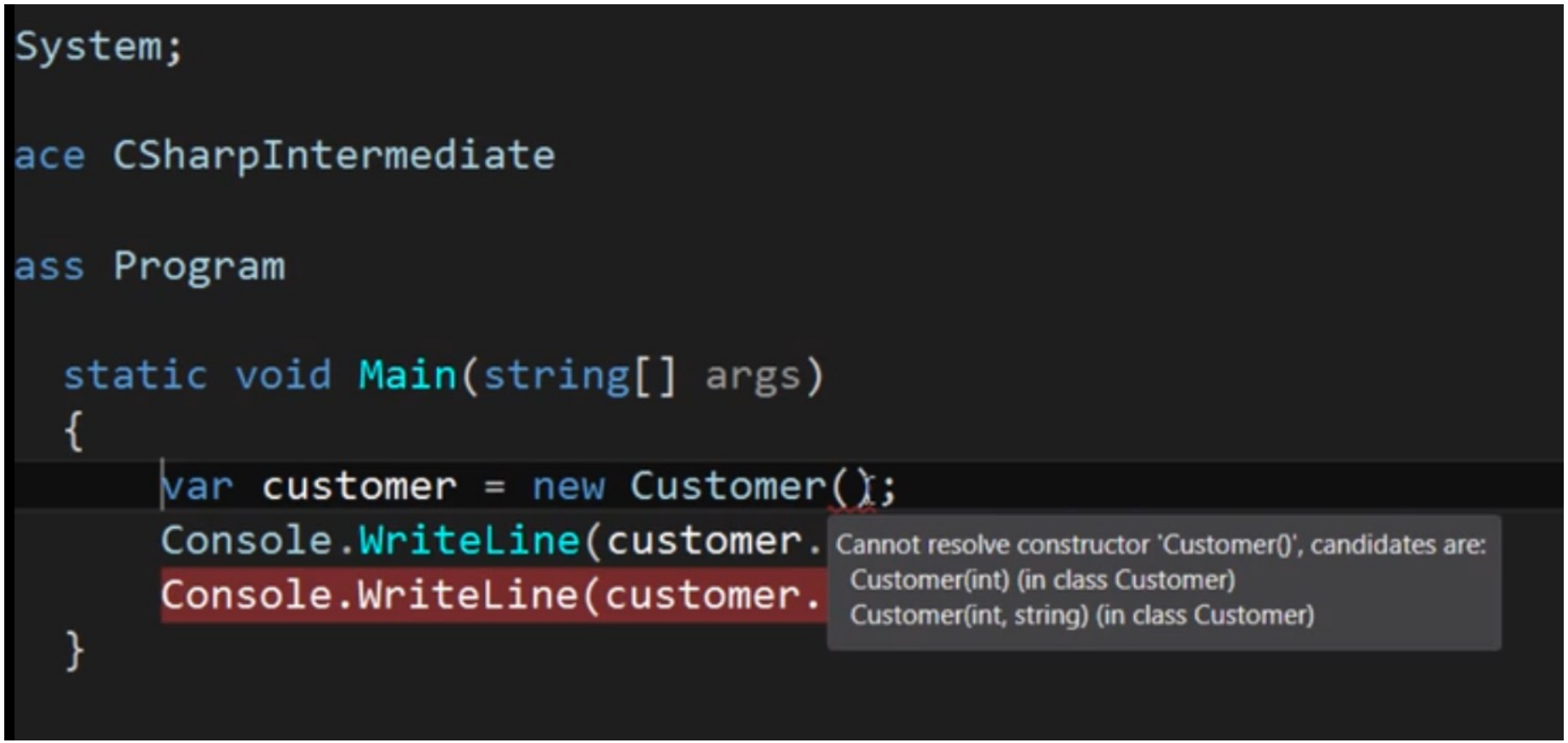

Customer.cs
Program.cs
CSharpIntermediate.Program
namespace CSharpIntermediate
{
class Program
{
static void Main(string[] args)
{
}
}
}

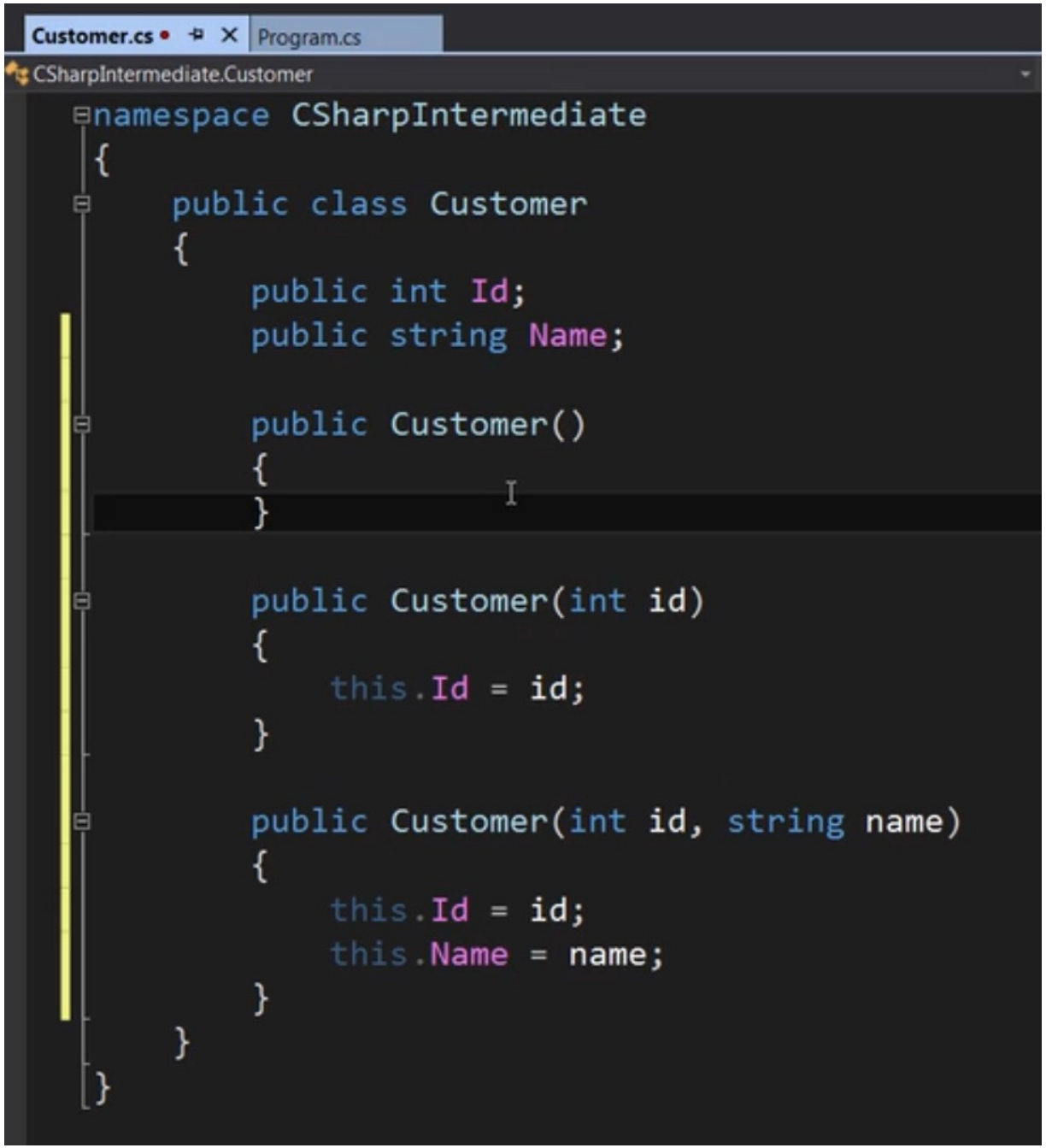

Customer.cs
Program.cs
CSharpIntermediate.Customer
namespace CSharpIntermediate
{
public class Customer
{
public int Id;
public string Name;
}
}

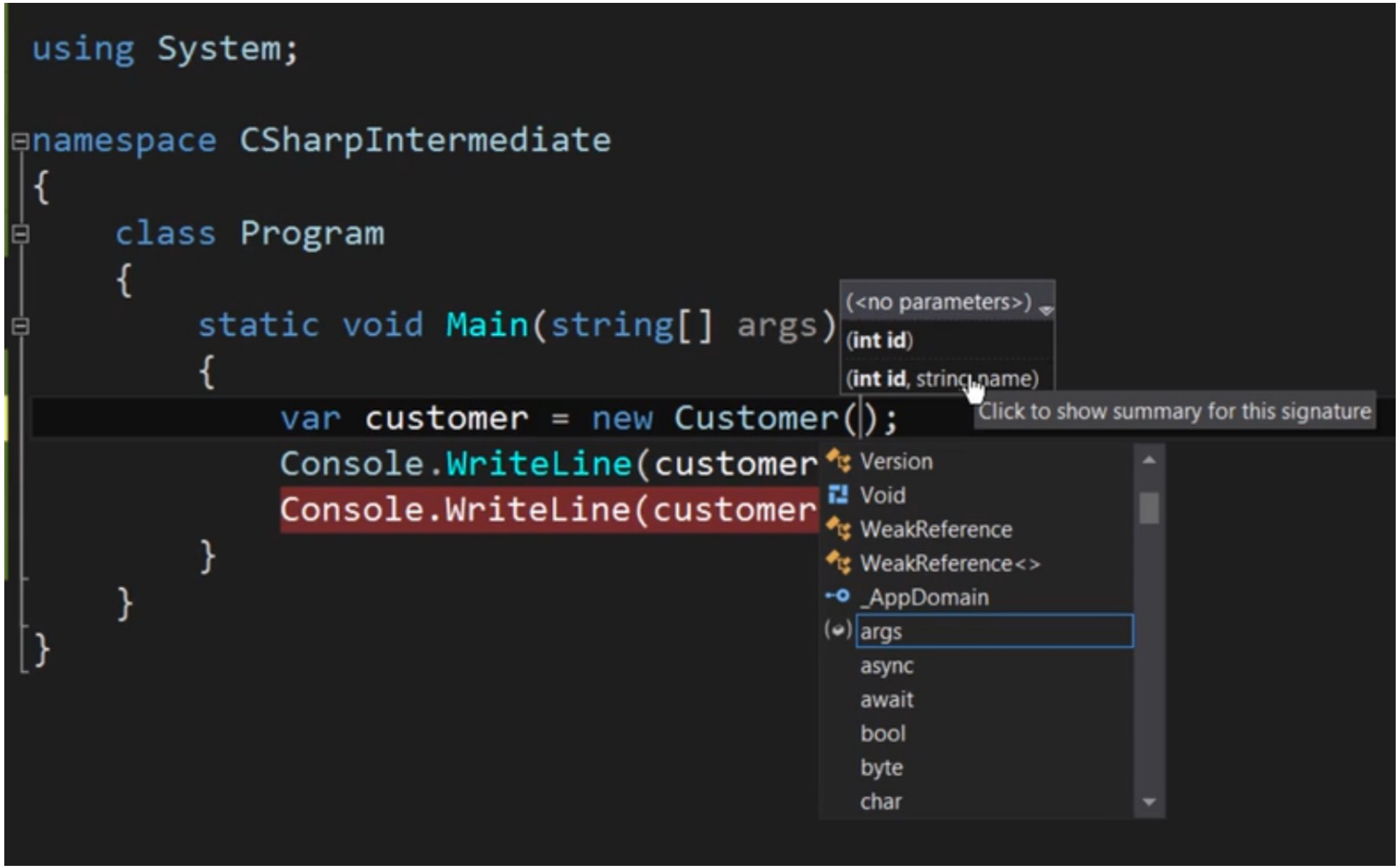

CSharpIntermediate.Program
using System;
namespace CSharpIntermediate
{
class Program
{
static void Main(string[] args)
{
var customer = new Customer();
Console.WriteLine(customer.Id);
Console.WriteLine(customer.Name);
}
}
}

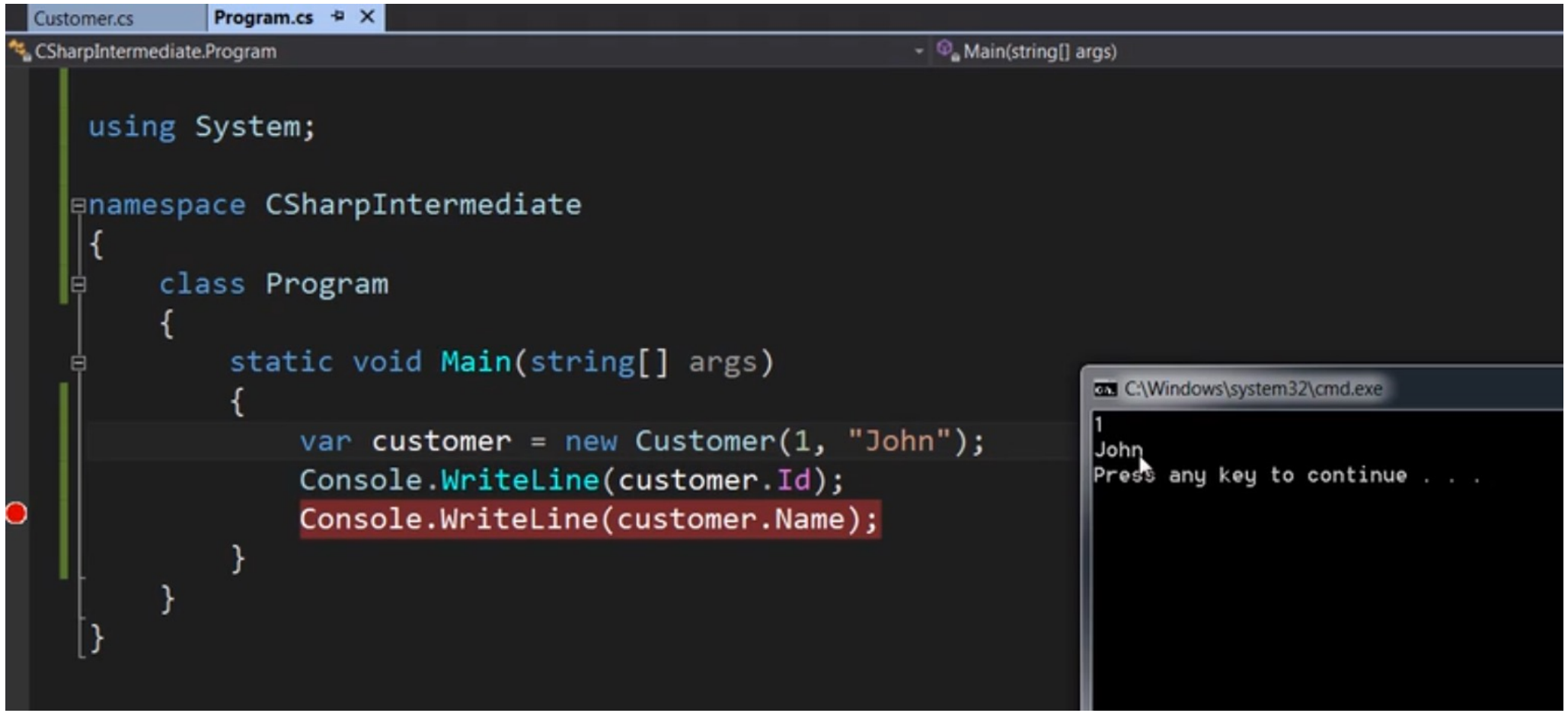

namespace CSharpIntermediate
{
public class Customer
{
public int Id;
public string Name;
public Customer(int id)
{
this.Id = id;
}
}
}

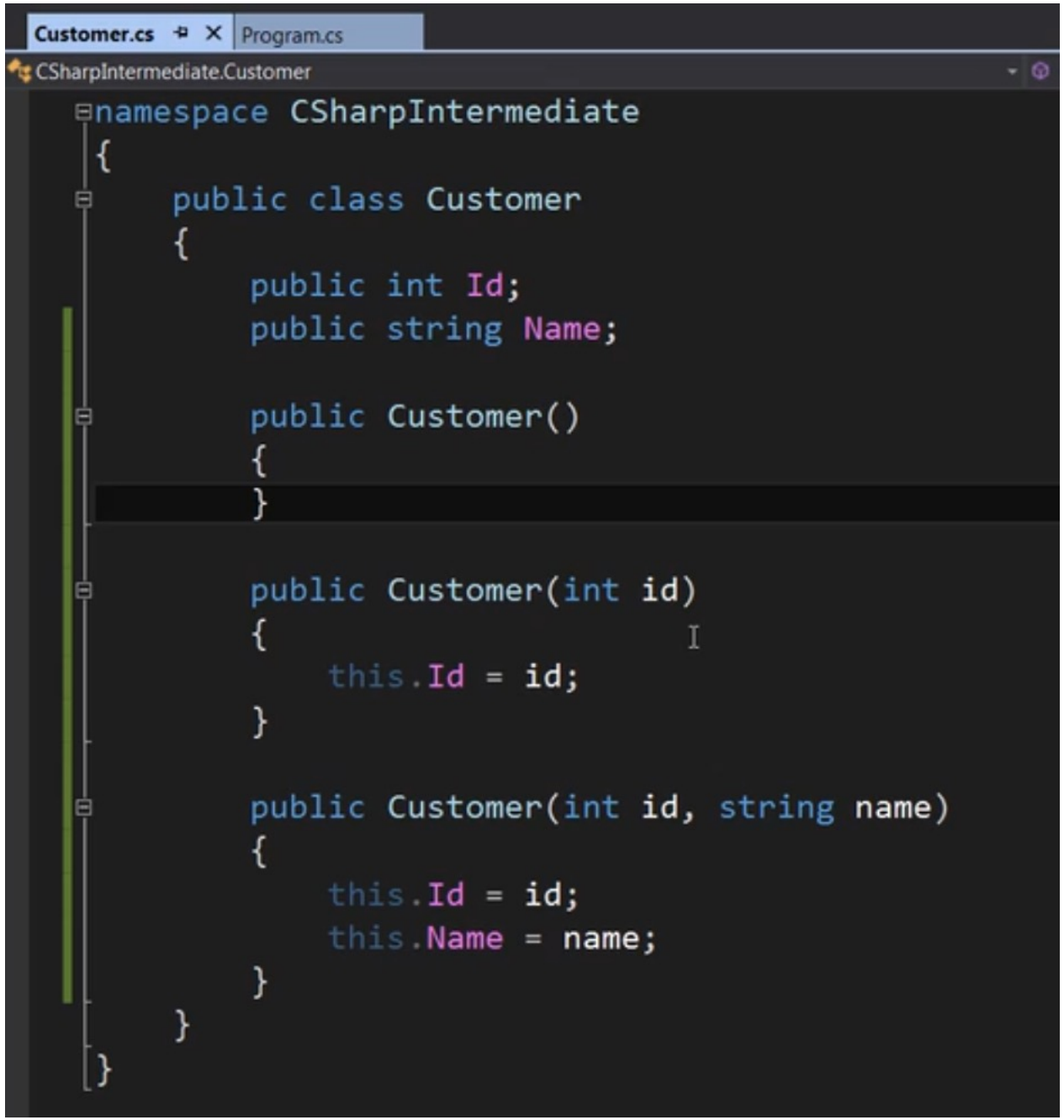

Customer.cs
Program.cs
CSharpIntermediate.Customer
namespace CSharpIntermediate
{
public class Customer
{
public int Id;
public string Name;
public Customer(int id)
{
this.Id = id;
}
public Customer(int id, string name)
{
this.Id = id;
this.Name = name;
}
}
}

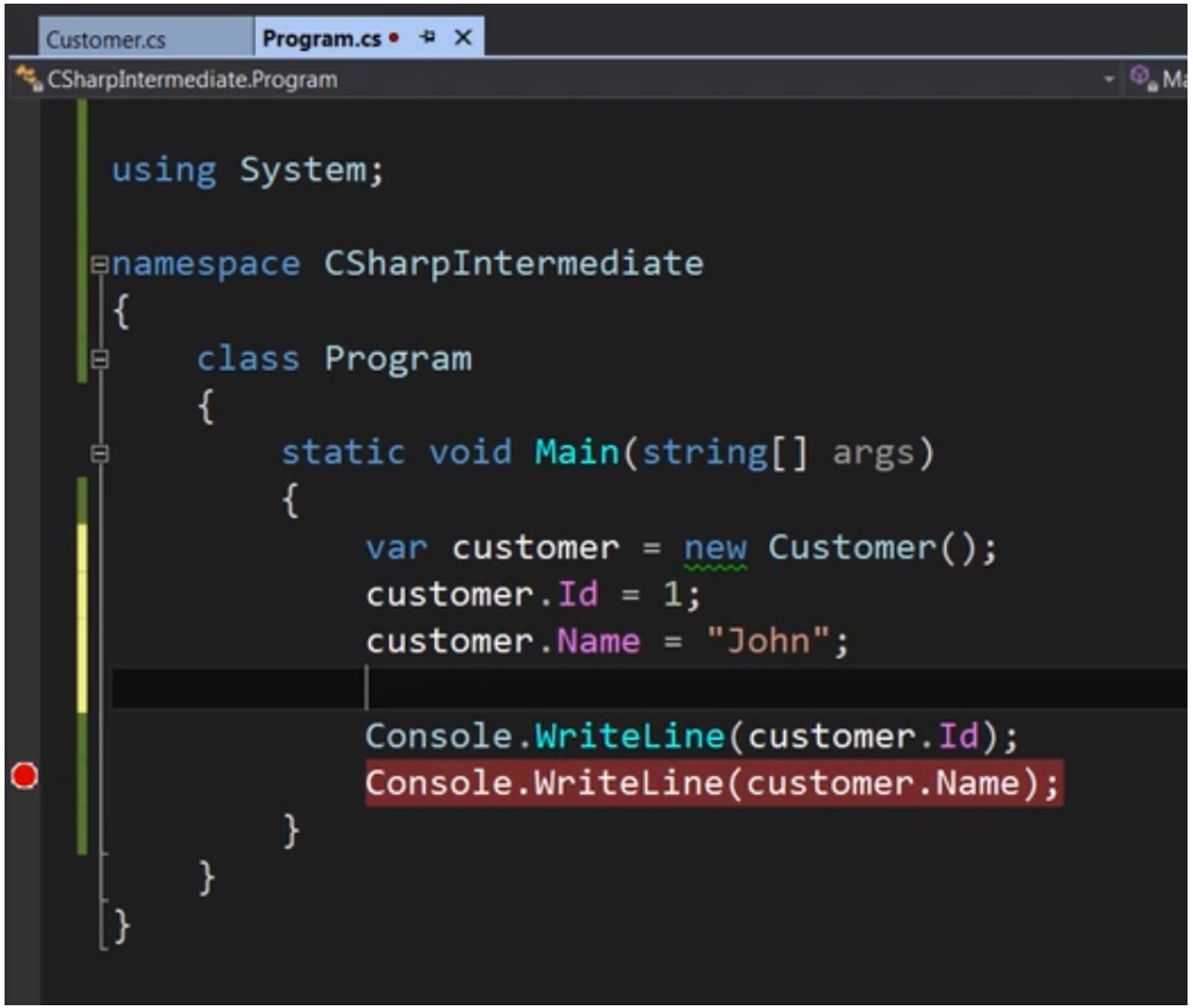

System;
ace CSharpIntermediate
ass Program
static void Main(string[] args)
{
var customer = new Customer();
Console.WriteLine(customer.
Console.WriteLine(customer.
}
Cannot resolve constructor 'Customer()', candidates are:
Customer(int) (in class Customer)
Customer(int, string) (in class Customer)

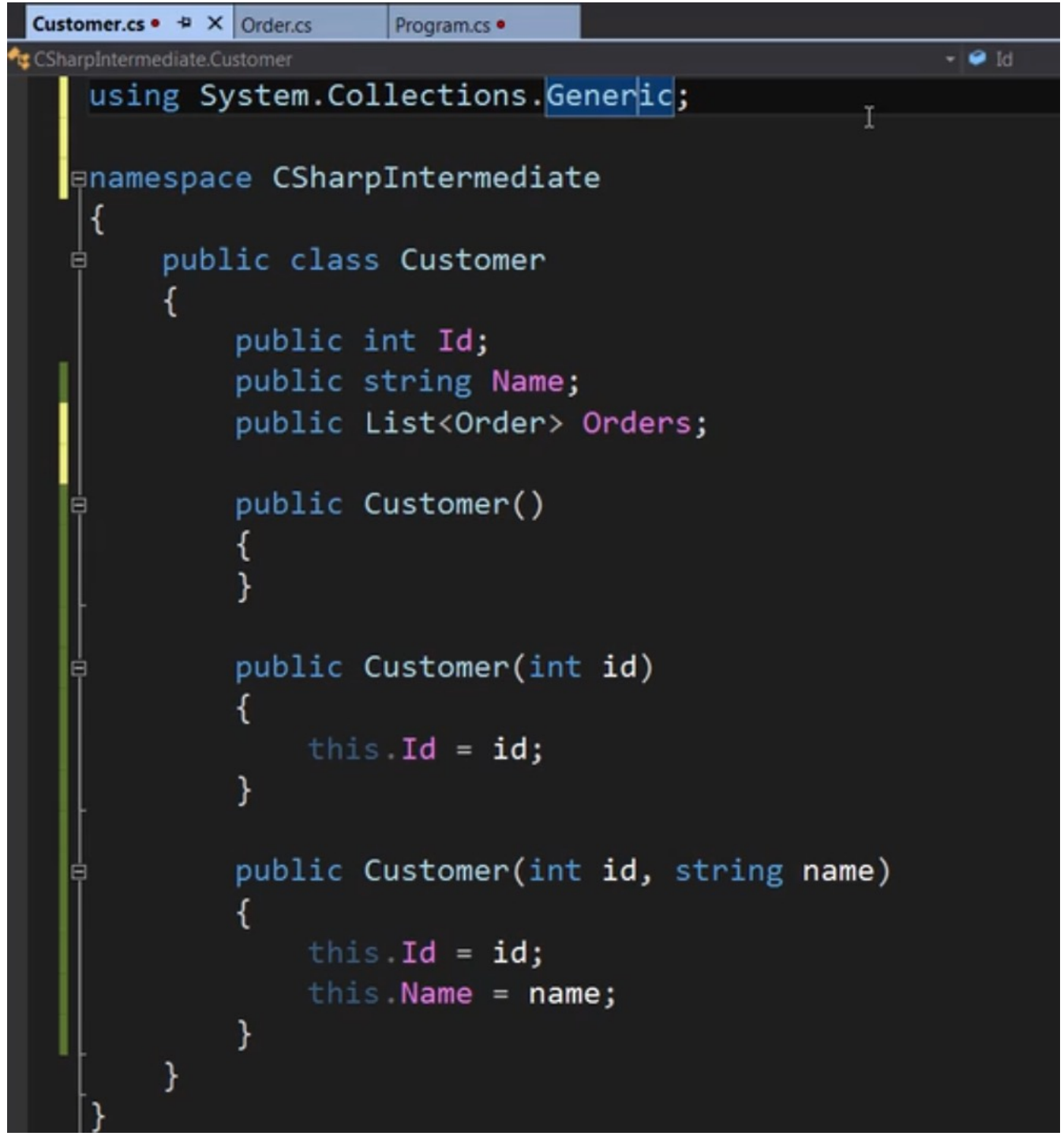

Customer.cs
Program.cs
CSharpIntermediate.Customer
namespace CSharpIntermediate
{
public class Customer
{
public int Id;
public string Name;
public Customer()
{
}
public Customer(int id)
{
this.Id = id;
}
public Customer(int id, string name)
{
this.Id = id;
this.Name = name;
}
}
}

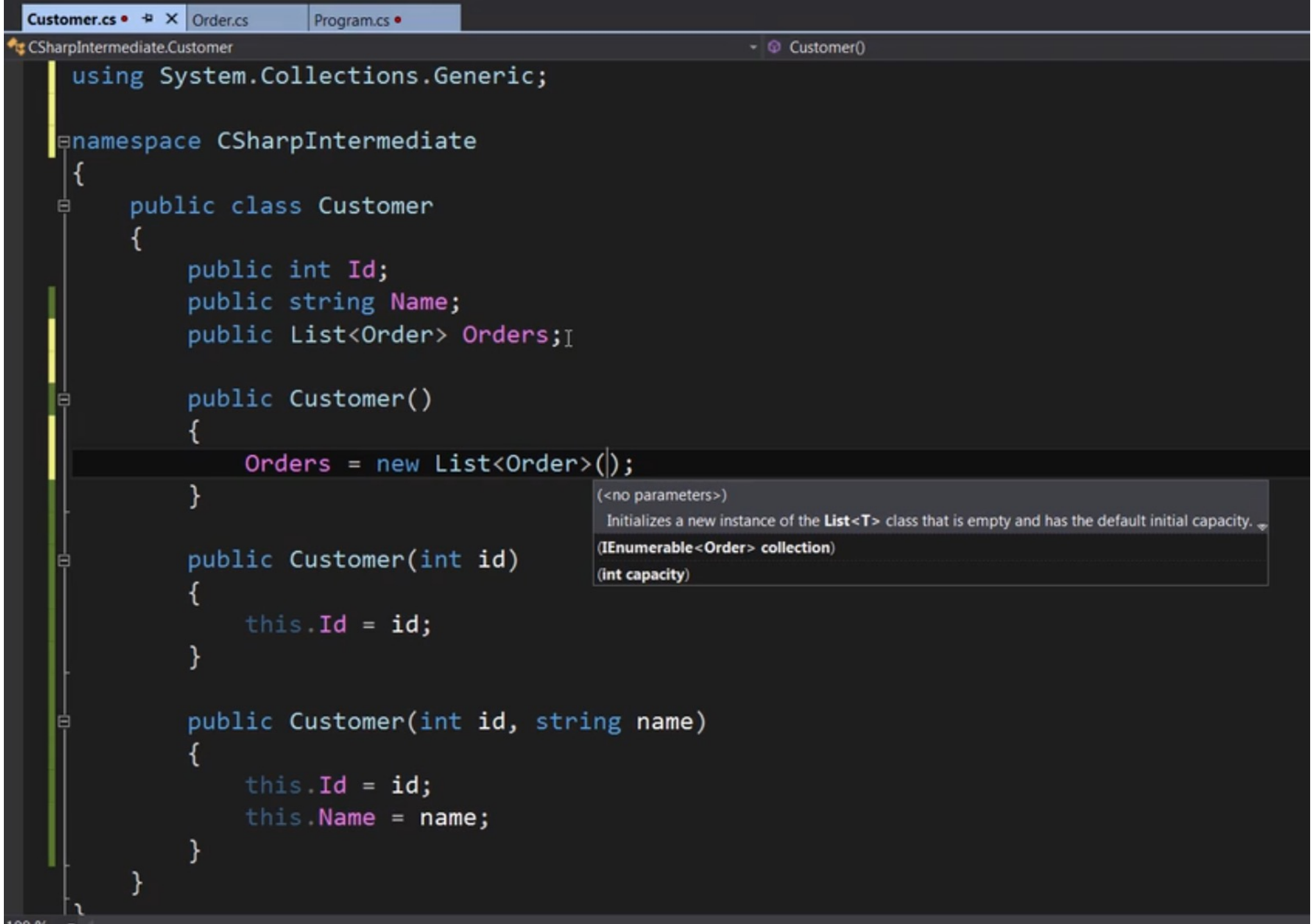

using System;
namespace CSharpIntermediate
{
class Program
{
static void Main(string[] args)
{
var customer = new Customer();
Console.WriteLine(customer
Console.WriteLine(customer
}
}
}
(<no parameters>)
(int id)
(int id, string name)
Click to show summary for this signature
Version
Void
WeakReference
WeakReference<>
_AppDomain
args
async
await
bool
byte
char

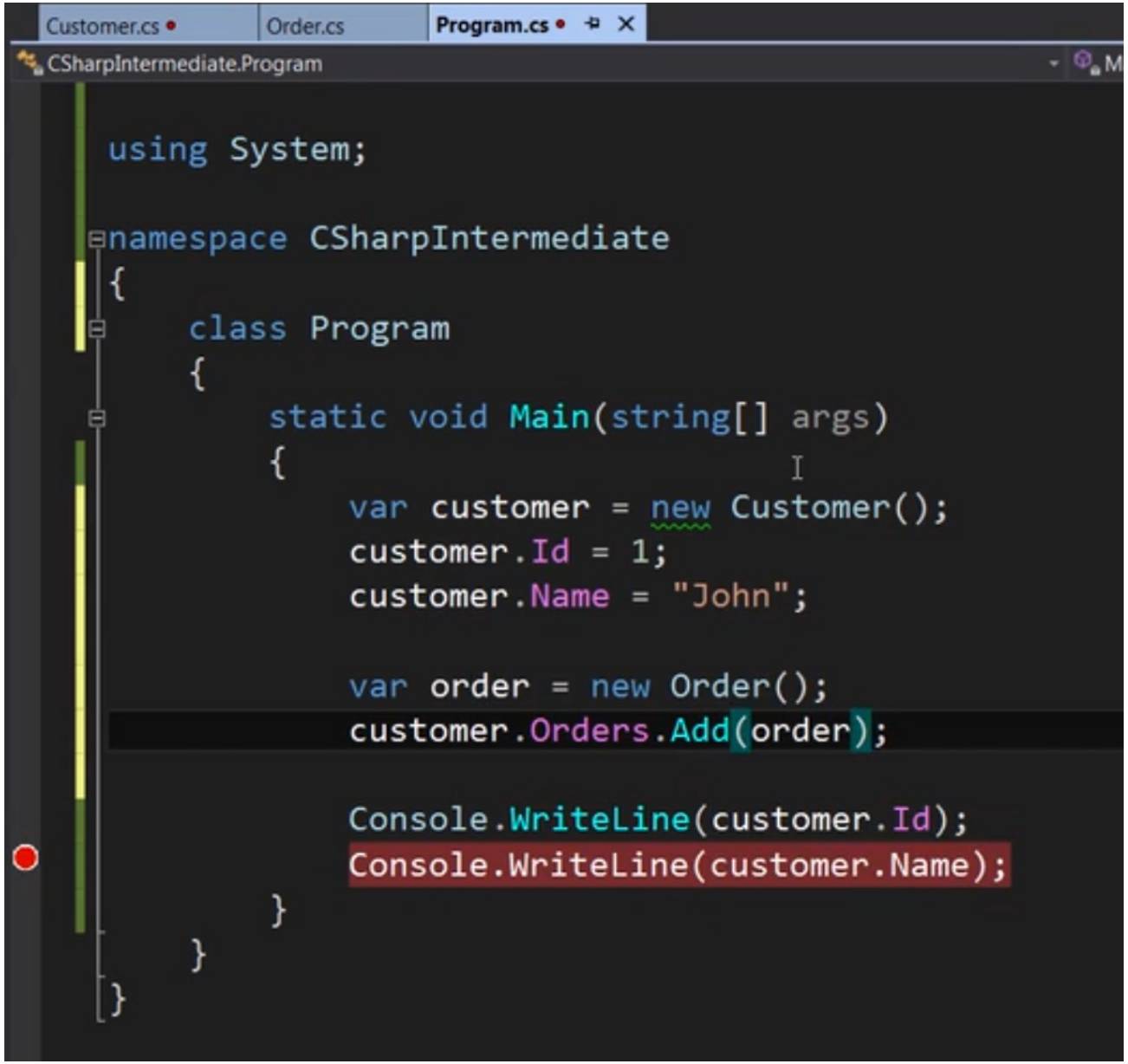

Customer.cs
Program.cs
CSharpIntermediate.Program
Main(string[] args)
using System;
namespace CSharpIntermediate
{
class Program
{
static void Main(string[] args)
{
var customer = new Customer(1, "John");
Console.WriteLine(customer.Id);
Console.WriteLine(customer.Name);
}
}
}
C:\Windows\system32\cmd.exe
1
John
Press any key to continue . . .

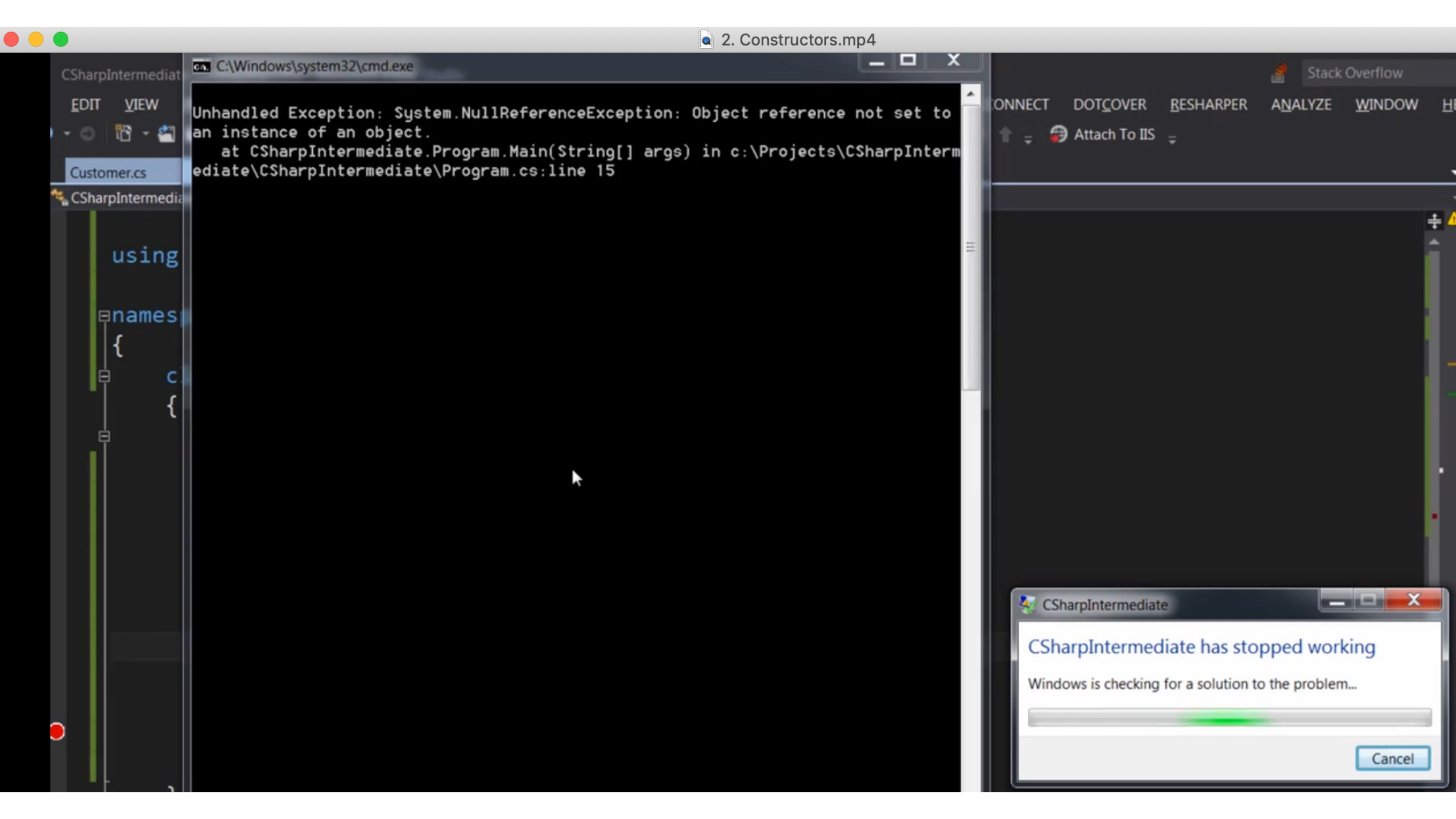

Customer.cs
Program.cs
CSharpIntermediate.Customer
namespace CSharpIntermediate
{
public class Customer
{
public int Id;
public string Name;
public Customer()
{
}
public Customer(int id)
{
this.Id = id;
}
public Customer(int id, string name)
{
this.Id = id;
this.Name = name;
}
}
}

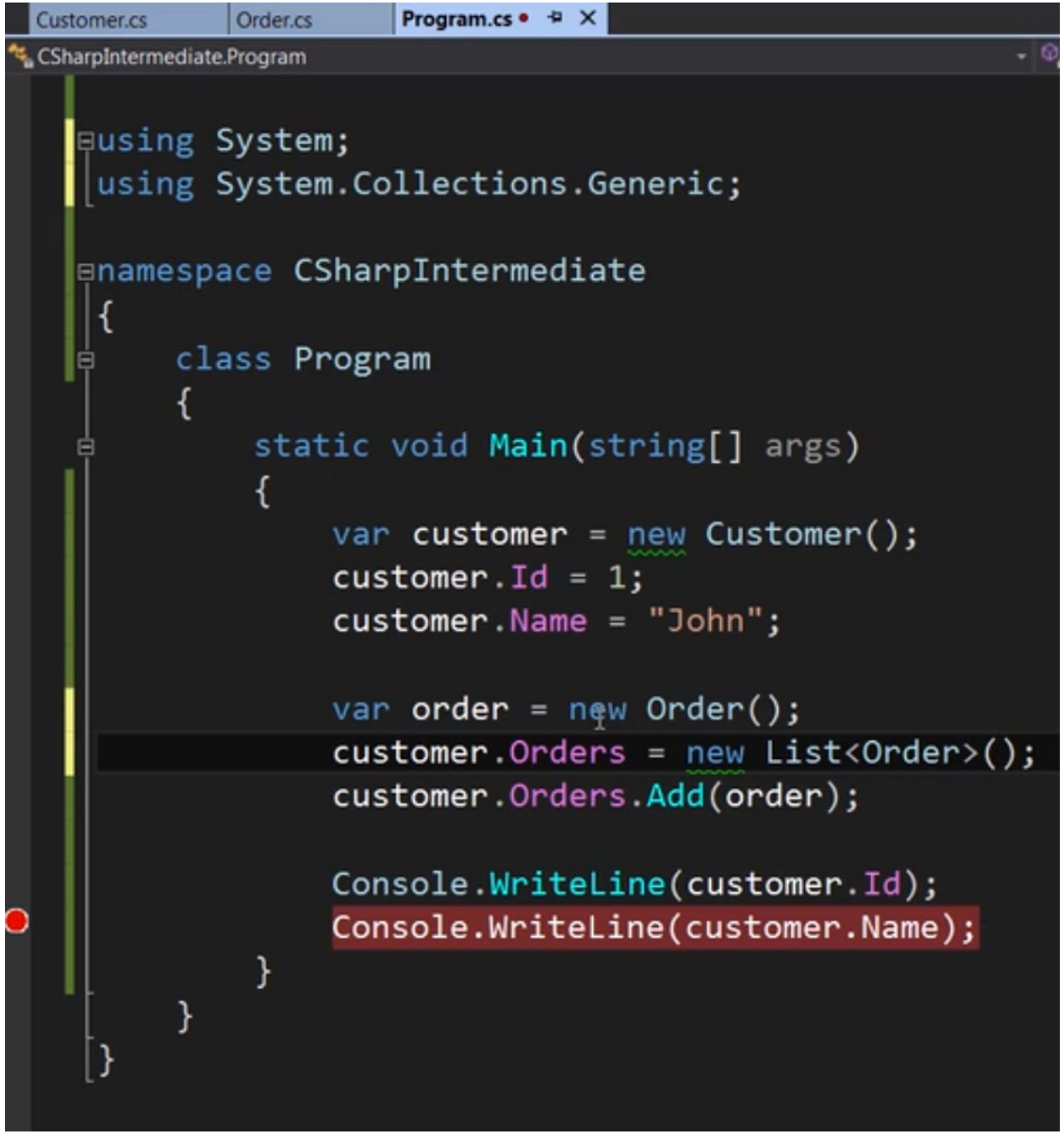

Customer.cs
Program.cs
CSharpIntermediate.Program
using System;
namespace CSharpIntermediate
{
class Program
{
static void Main(string[] args)
{
var customer = new Customer();
customer.Id = 1;
customer.Name = "John";
Console.WriteLine(customer.Id);
Console.WriteLine(customer.Name);
}
}
}

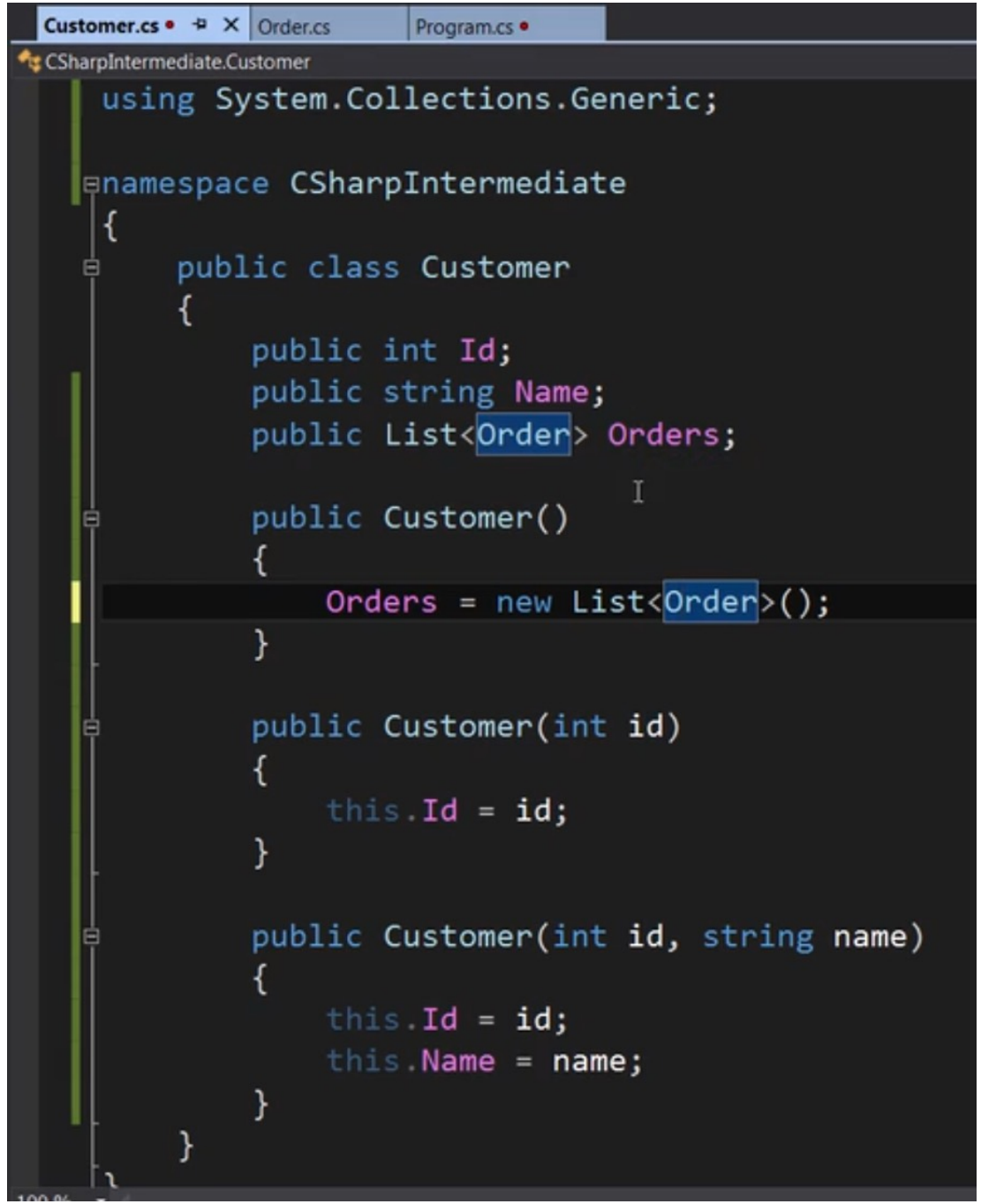

Customer.cs
Order.cs
Program.cs
CSharpIntermediate.Customer
Id
using System.Collections.Generic;
namespace CSharpIntermediate
{
public class Customer
{
public int Id;
public string Name;
public List<Order> Orders;
public Customer()
{
}
public Customer(int id)
{
this.Id = id;
}
public Customer(int id, string name)
{
this.Id = id;
this.Name = name;
}
}
}

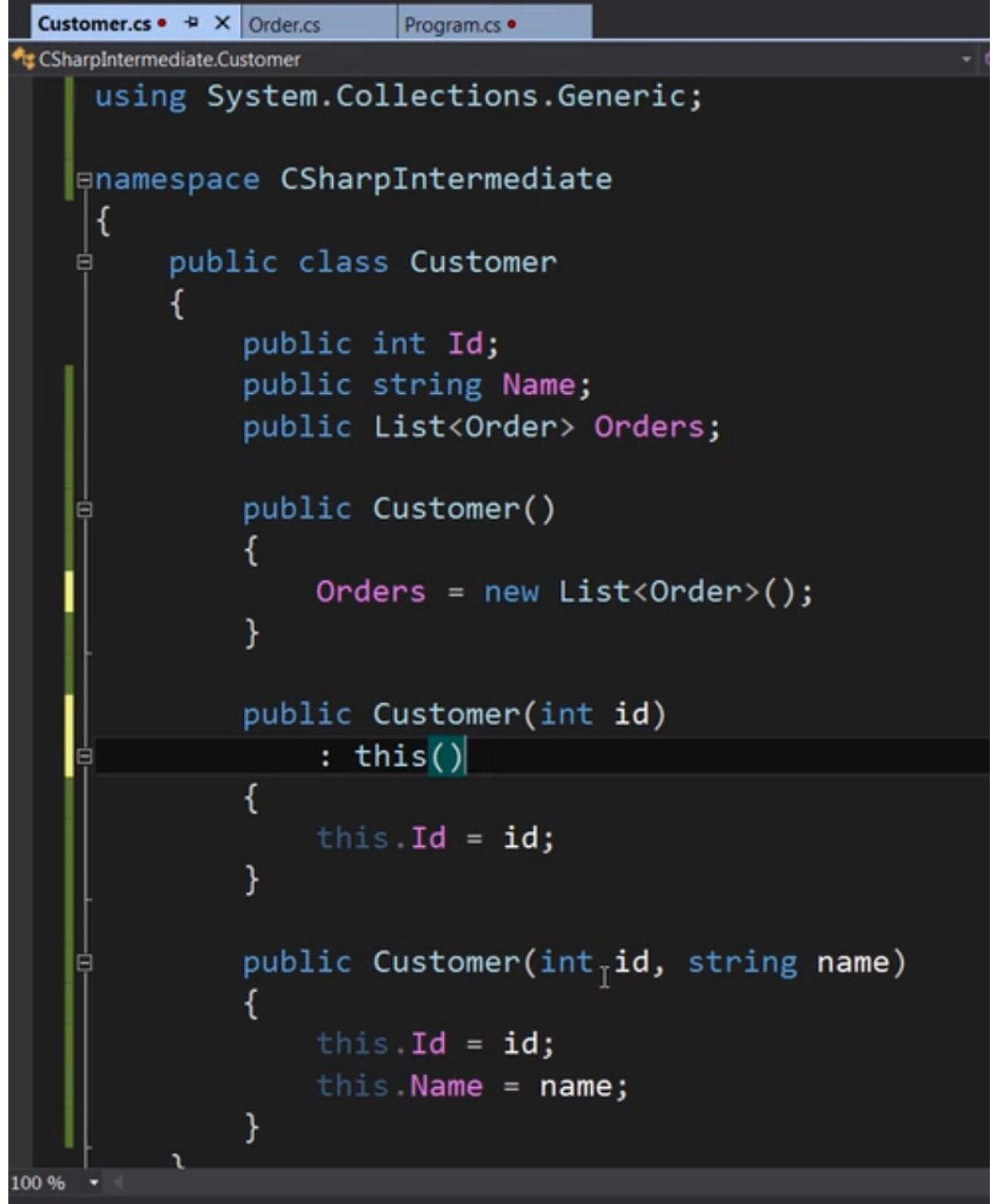

Customer.cs
Order.cs
Program.cs
CSharpIntermediate.Customer
Customer()
using System.Collections.Generic;
namespace CSharpIntermediate
{
public class Customer
{
public int Id;
public string Name;
public List<Order> Orders;
public Customer()
{
Orders = new List<Order>();
}
public Customer(int id)
{
this.Id = id;
}
public Customer(int id, string name)
{
this.Id = id;
this.Name = name;
}
}
(<no parameters>)
Initializes a new instance of the List<T> class that is empty and has the default initial capacity.
(IEnumerable<Order> collection)
(int capacity)

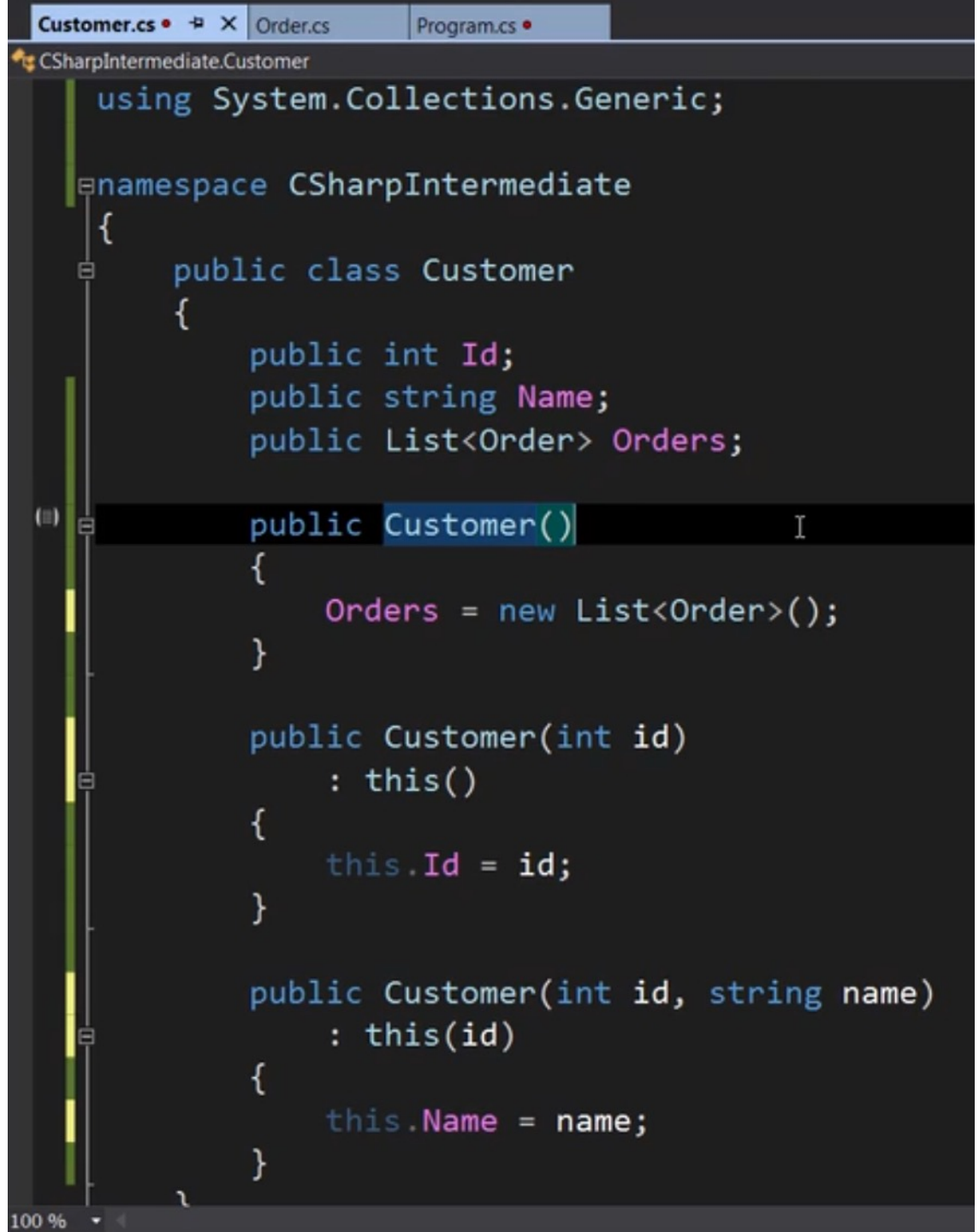

Customer.cs
Order.cs
Program.cs
CSharpIntermediate.Program
using System;
namespace CSharpIntermediate
{
class Program
{
static void Main(string[] args)
{
var customer = new Customer();
customer.Id = 1;
customer.Name = "John";
var order = new Order();
customer.Orders.Add(order);
Console.WriteLine(customer.Id);
Console.WriteLine(customer.Name);
}
}
}


2. Constructors.mp4
C:\Windows\system32\cmd.exe
Unhandled Exception: System.NullReferenceException: Object reference not set to an instance of an object.
at CSharpIntermediate.Program.Main(String[] args) in c:\Projects\CSharpIntermediate\CSharpIntermediate\Program.cs:line 15
CSharpIntermediate has stopped working
Windows is checking for a solution to the problem...
Cancel
Stack Overflow
Attach To IIS
Customer.cs


Customer.cs
Order.cs
Program.cs
CSharpIntermediate.Program
using System;
using System.Collections.Generic;
namespace CSharpIntermediate
{
class Program
{
static void Main(string[] args)
{
var customer = new Customer();
customer.Id = 1;
customer.Name = "John";
var order = new Order();
customer.Orders = new List<Order>();
customer.Orders.Add(order);
Console.WriteLine(customer.Id);
Console.WriteLine(customer.Name);
}
}
}


Customer.cs
Order.cs
Program.cs
CSharpIntermediate.Customer
using System.Collections.Generic;
namespace CSharpIntermediate
{
public class Customer
{
public int Id;
public string Name;
public List<Order> Orders;
public Customer()
{
Orders = new List<Order>();
}
public Customer(int id)
{
this.Id = id;
}
public Customer(int id, string name)
{
this.Id = id;
this.Name = name;
}
}


Customer.cs
Order.cs
Program.cs
CSharpIntermediate.Customer
using System.Collections.Generic;
namespace CSharpIntermediate
{
public class Customer
{
public int Id;
public string Name;
public List<Order> Orders;
public Customer()
{
Orders = new List<Order>();
}
public Customer(int id)
: this()
{
this.Id = id;
}
public Customer(int id, string name)
{
this.Id = id;
this.Name = name;
}
100 %


Customer.cs
Order.cs
Program.cs
CSharpIntermediate.Customer
using System.Collections.Generic;
namespace CSharpIntermediate
{
public class Customer
{
public int Id;
public string Name;
public List<Order> Orders;
public Customer()
{
Orders = new List<Order>();
}
public Customer(int id)
: this()
{
this.Id = id;
}
public Customer(int id, string name)
: this(id)
{
this.Name = name;
}
100 %

C# INTERMEDIATE

Object Initializers

What?

A syntax for quickly initialising an object without the need to call one of its constructors.

Why?

To avoid creating multiple constructors.

How?

```
public class Person
{
    public int Id;

    public string FirstName;

    public string LastName;

    public DateTime Birthdate;
}
```

How?

```
public class Person
{
    public Person(int id) {}

    public Person(int id, string firstName) {}

    public Person(int id, string firstName, string lastName) {}

    public Person(int id, DateTime birthdate) {}
}
```

How?

```
var person = new Person
                {
                    FirstName = “Mosh”,
                    LastName = “Hamedani”
                };
```