

# Working with Text



**Zouhair Rimale, Ph.D.**

Expert technique .NET | SharePoint | ASP.NET MVC | WEB API | Angular

# Strings



**Zouhair Rimale, Ph.D.**

Expert technique .NET | SharePoint | ASP.NET MVC | WEB API | Angular

# In this lecture...

- Useful string methods
- Converting a string to a number
- Converting a number to a string



**Zouhair Rimale, Ph.D.**

Expert technique .NET | SharePoint | ASP.NET MVC | WEB API | Angular

# Formatting

`ToLower()`            `// "hello world"`

`ToUpper()`           `// "HELLO WORLD"`

`Trim()`



**Zouhair Rimale, Ph.D.**

Expert technique .NET | SharePoint | ASP.NET MVC | WEB API | Angular

# Searching

```
IndexOf( 'a' )
```

```
LastIndexOf( "Hello" )
```



**Zouhair Rimale, Ph.D.**

Expert technique .NET | SharePoint | ASP.NET MVC | WEB API | Angular

# Substrings

```
Substring(startIndex)
```

```
Substring(startIndex, length)
```



**Zouhair Rimale, Ph.D.**

Expert technique .NET | SharePoint | ASP.NET MVC | WEB API | Angular

# Replacing

```
Replace('a', '!')
```

```
Replace("mosh", "moshfegh")
```



**Zouhair Rimale, Ph.D.**

Expert technique .NET | SharePoint | ASP.NET MVC | WEB API | Angular

# Null checking

```
String.IsNullOrEmpty(str)
```

```
String.IsNullOrWhiteSpace(str)
```



**Zouhair Rimale, Ph.D.**

Expert technique .NET | SharePoint | ASP.NET MVC | WEB API | Angular



# Splitting

```
str.Split(' ')
```



**Zouhair Rimale, Ph.D.**

Expert technique .NET | SharePoint | ASP.NET MVC | WEB API | Angular

# Converting Strings to Numbers

```
string s = "1234";
```

```
int i = int.Parse(s);
```

```
int j = Convert.ToInt32(s);
```



**Zouhair Rimale, Ph.D.**

Expert technique .NET | SharePoint | ASP.NET MVC | WEB API | Angular

# Converting Numbers to Strings

```
int i = 1234;
```

```
string s = i.ToString();           // "1234"
```

```
string t = i.ToString("C");       // "$1,234.00"
```

```
string t = i.ToString("C0");      // "$1,234"
```



**Zouhair Rimale, Ph.D.**

Expert technique .NET | SharePoint | ASP.NET MVC | WEB API | Angular

# Format Strings

Format Specifier	Description	Example
<b>c</b> or <b>C</b>	Currency	123456 (C) -> \$123,456
<b>d</b> or <b>D</b>	Decimal	1234 (D6) -> 001234
<b>e</b> or <b>E</b>	Exponential	1052.0329112756 (E) -> 1.052033E+003
<b>f</b> or <b>F</b>	Fixed Point	1234.567 (F1) -> 1234.5
<b>x</b> or <b>X</b>	Hexadecimal	255 (X) -> FF



**Zouhair Rimale, Ph.D.**

Expert technique .NET | SharePoint | ASP.NET MVC | WEB API | Angular

# Demo Strings



**Zouhair Rimale, Ph.D.**

Expert technique .NET | SharePoint | ASP.NET MVC | WEB API | Angular

Program.cs

CSharpFundamentals

CSharpFundamentals.Program

Main(string[] args)

```
using System;
```

```
namespace CSharpFundamentals
```

```
{
```

```
    class Program
```

```
    {
```

```
        static void Main(string[] args)
```

```
        {
```

```
            var fullName = "Mosh Hamedani ";
```

```
            Console.WriteLine("Trim: '{0}'", fullName.Trim());
```

```
            Console.WriteLine("ToUpper: '{0}'", fullName.Trim().ToUpper());
```

```
            var index = fullName.IndexOf(' ');
```

```
            var firstName = fullName.Substring(0, index);
```

```
            var lastName = fullName.Substring(index + 1);
```

```
            Console.WriteLine("FirstName: " + firstName);
```

```
            Console.WriteLine("LastName: " + lastName);
```

```
            var names = fullName.Split(' ');
```

```
            Console.WriteLine("FirstName: " + names[0]);
```

```
            Console.WriteLine("LastName: " + names[1]);
```

```
            Console.WriteLine(fullName.Replace("Mosh", "Moshfegh"));
```

```
            if (String.IsNullOrEmpty(" "))
```

```
                Console.WriteLine("Invalid");
```



Program.cs X

CSharpFundamentals

CSharpFundamentals.Program

Main(string[] args)

```
var index = fullName.IndexOf(' ');  
var firstName = fullName.Substring(0, index);  
var lastName = fullName.Substring(index + 1);  
Console.WriteLine("FirstName: " + firstName);  
Console.WriteLine("LastName: " + lastName);  
  
var names = fullName.Split(' ');  
Console.WriteLine("FirstName: " + names[0]);  
Console.WriteLine("LastName: " + names[1]);  
  
Console.WriteLine(fullName.Replace("Mosh", "Moshfegh"));  
  
if (String.IsNullOrEmpty(" "))  
    Console.WriteLine("Invalid");  
  
var str = "25";  
var age = Convert.ToByte(str);  
Console.WriteLine(age);  
  
float price = 29.95f;  
Console.WriteLine(price.ToString("C0"));
```

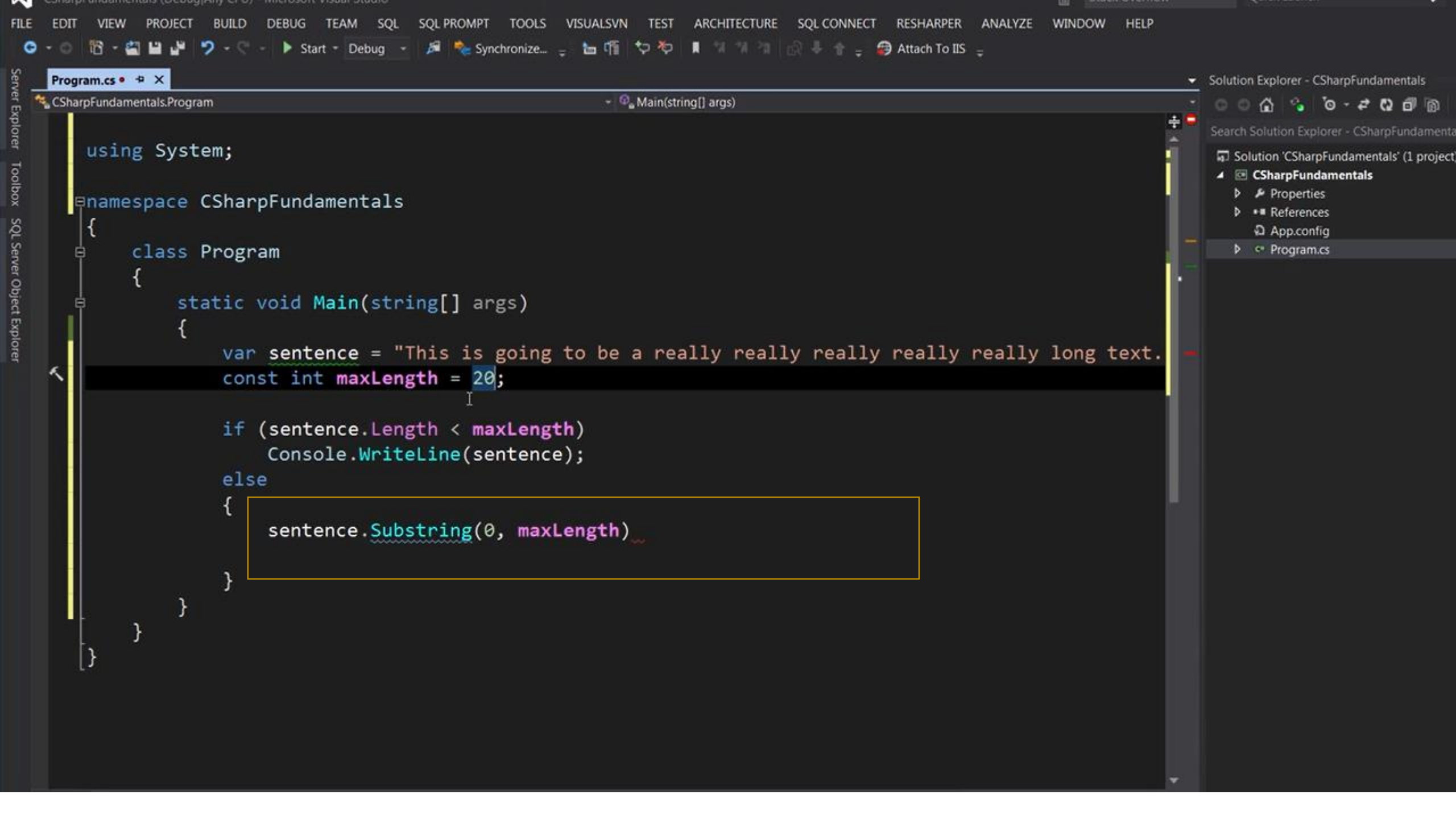
# Live Coding Summarizing Text



**Zouhair Rimale, Ph.D.**

Expert technique .NET | SharePoint | ASP.NET MVC | WEB API | Angular





Program.cs

CSharpFundamentals.Program

Main(string[] args)

```
using System;

namespace CSharpFundamentals
{
    class Program
    {
        static void Main(string[] args)
        {
            var sentence = "This is going to be a really really really really really long text."
            const int maxLength = 20;

            if (sentence.Length < maxLength)
                Console.WriteLine(sentence);
            else
            {
                sentence.Substring(0, maxLength)
            }
        }
    }
}
```

Solution Explorer - CSharpFundamentals

Search Solution Explorer - CSharpFundamentals

- Solution 'CSharpFundamentals' (1 project)
  - CSharpFundamentals
    - Properties
    - References
    - App.config
    - Program.cs

```
Program.cs
CSharpFundamentals.Program
Main(string[] args)

class Program
{
    static void Main(string[] args)
    {
        var sentence = "This is going to be a really really really really really long text."
        const int maxLength = 20;

        if (sentence.Length < maxLength)
            Console.WriteLine(sentence);
        else
        {
            var words = sentence.Split(' ');
            var totalCharacters = 0;
            var summaryWords = new List<string>();

            foreach (var word in words)
            {
                summaryWords.Add(word);

                totalCharacters += word.Length + 1;
                if (totalCharacters > maxLength)
                    break;
            }

            var summary = String.Join(" ", summaryWords) + "...";
            Console.WriteLine(summary);
        }
    }
}
```

Program.cs

CSharpFundamentals.Program

Main(string[] args)

```
using System;
using System.Collections.Generic;

namespace CSharpFundamentals
{
    class Program
    {
        static void Main(string[] args)
        {
            var sentence = "This is going to be a really really really really really long text."
            var summary = SummarizeText(sentence);
            Console.WriteLine(summary);
        }

        static string SummarizeText(string text, int maxLength = 20)
        {
            if (text.Length < maxLength)
                return text;

            var words = text.Split(' ');
            var totalCharacters = 0;
            var summaryWords = new List<string>();

            foreach (var word in words)
            {
```

Solution Explorer - CSharpFundamentals

Search Solution Explorer - CSharpFundamentals

Solution 'CSharpFundamentals' (1 project)

CSharpFundamentals

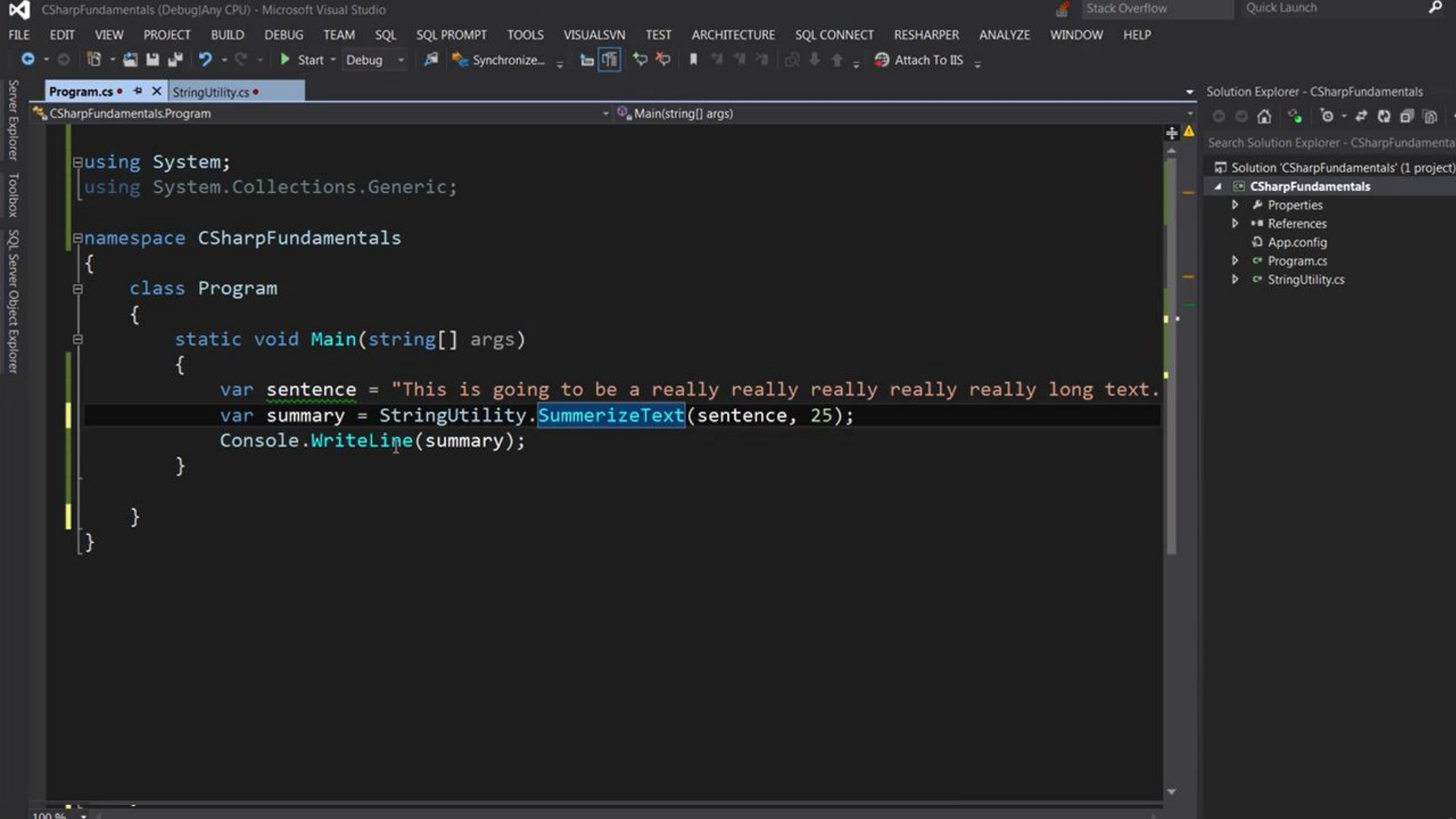
Properties

References

App.config

Program.cs





# StringBuilder



**Zouhair Rimale, Ph.D.**

Expert technique .NET | SharePoint | ASP.NET MVC | WEB API | Angular

# StringBuilder

- Defined in System.Text
- A mutable string
- Easy and fast to create and manipulate strings



**Zouhair Rimale, Ph.D.**

Expert technique .NET | SharePoint | ASP.NET MVC | WEB API | Angular

# Not for Searching

- IndexOf()
- LastIndexOf()
- Contains()
- StartsWith()
- ...



**Zouhair Rimale, Ph.D.**

Expert technique .NET | SharePoint | ASP.NET MVC | WEB API | Angular

# String Manipulation Methods

- Append()
- Insert()
- Remove()
- Replace()
- Clear()



**Zouhair Rimale, Ph.D.**

Expert technique .NET | SharePoint | ASP.NET MVC | WEB API | Angular

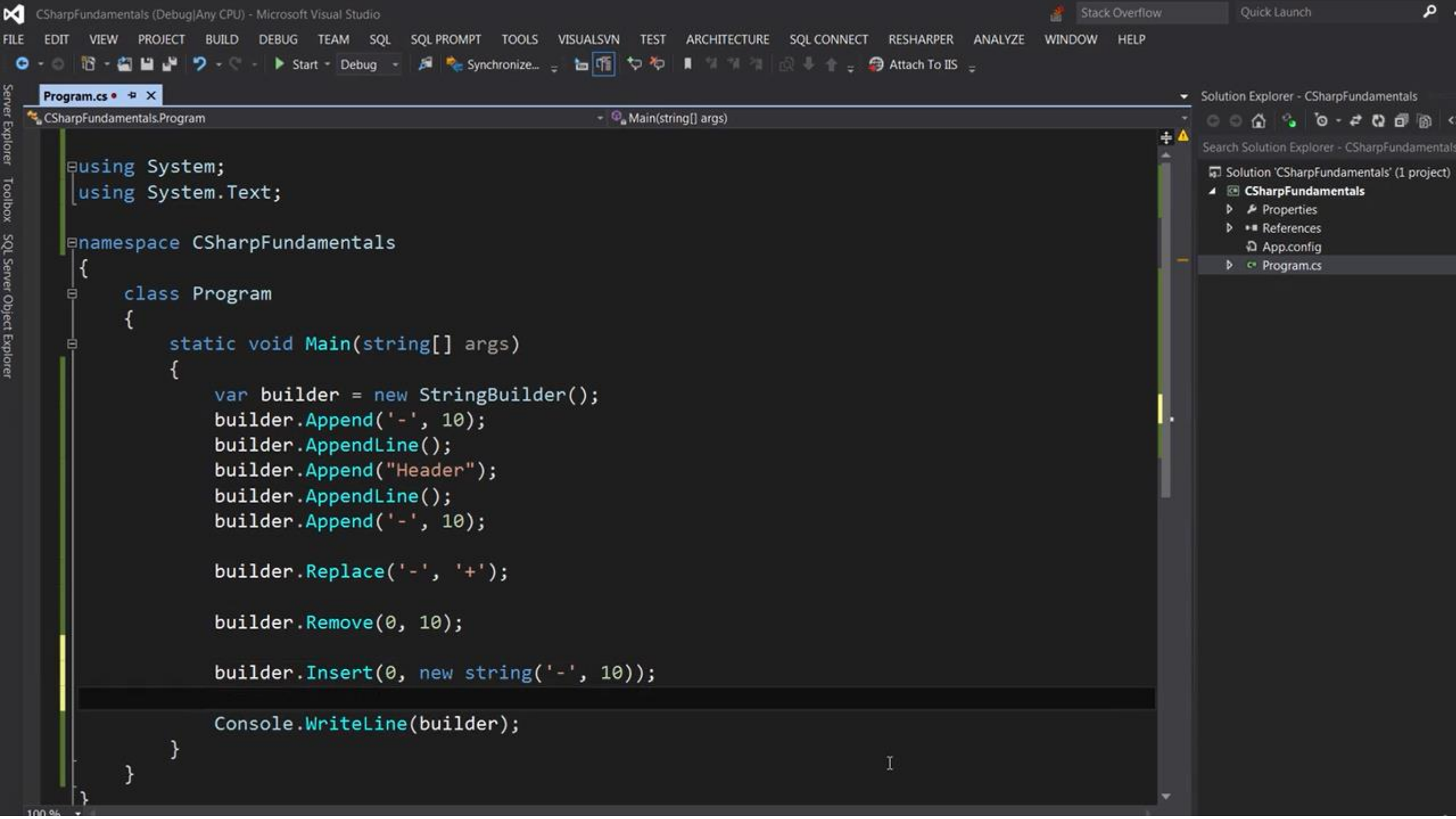
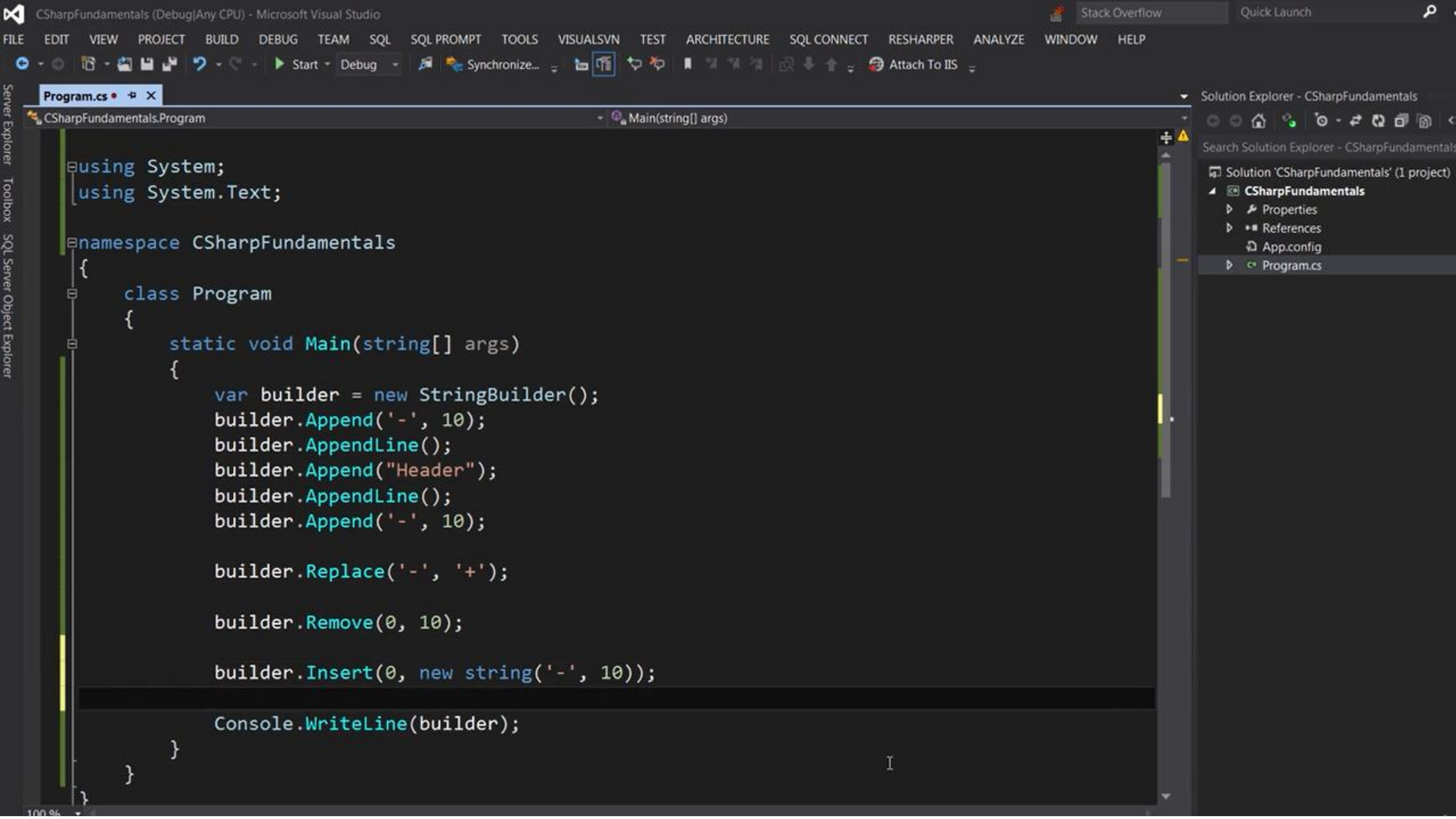


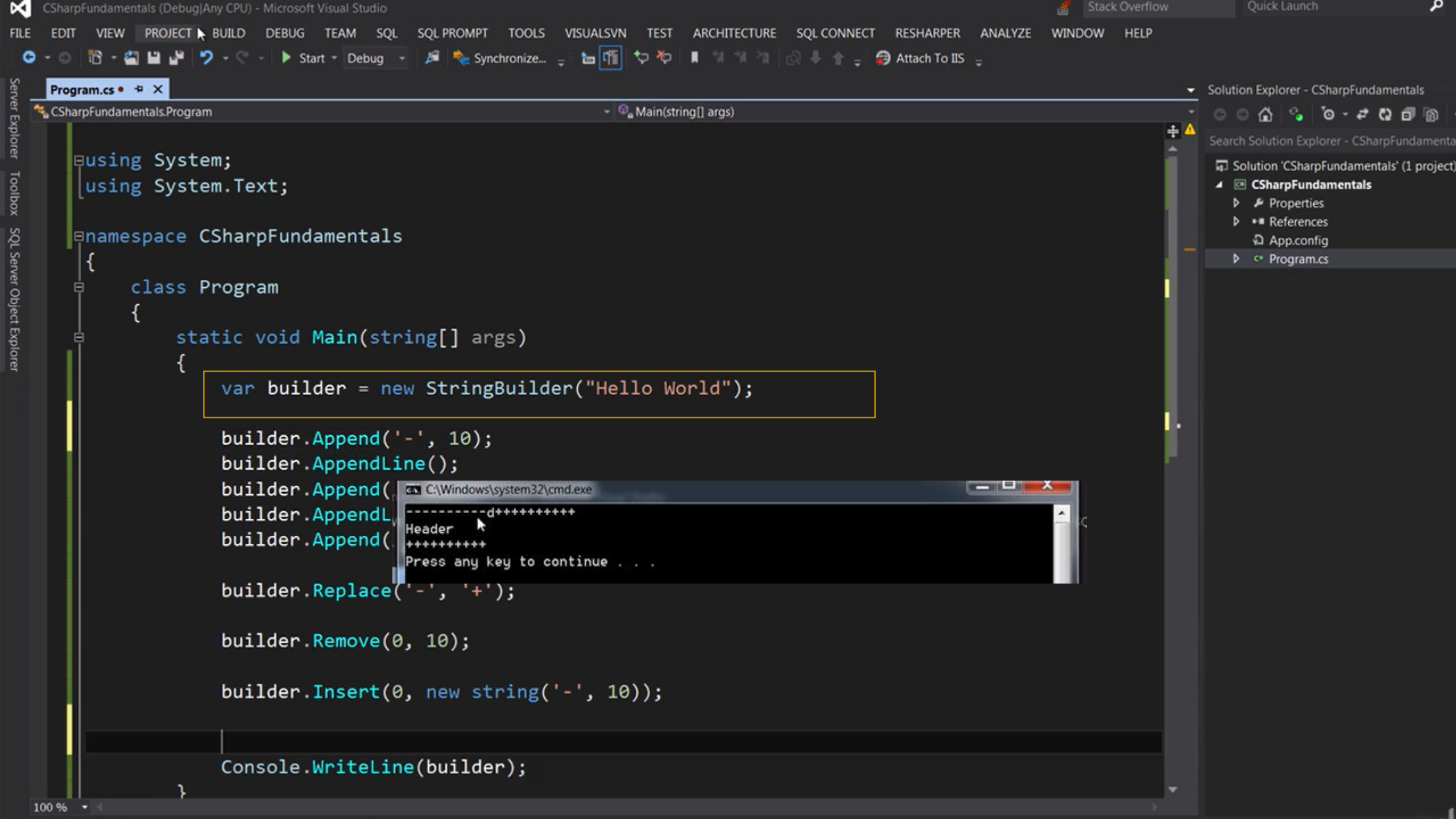
# Demo StringBuilder



**Zouhair Rimale, Ph.D.**

Expert technique .NET | SharePoint | ASP.NET MVC | WEB API | Angular





Program.cs

CSharpFundamentals.Program

Main(string[] args)

```
using System;
using System.Text;

namespace CSharpFundamentals
{
    class Program
    {
        static void Main(string[] args)
        {
            var builder = new StringBuilder("Hello World");

            builder.Append('-', 10);
            builder.AppendLine();
            builder.Append(
            builder.AppendLine(
            builder.Append(
            builder.Replace('-', '+');

            builder.Remove(0, 10);

            builder.Insert(0, new string('-', 10));

            Console.WriteLine(builder);
        }
    }
}
```

Solution Explorer - CSharpFundamentals

Search Solution Explorer - CSharpFundamentals

Solution 'CSharpFundamentals' (1 project)

- CSharpFundamentals
  - Properties
  - References
  - App.config
  - Program.cs

100 %

Program.cs

CSharpFundamentals.Program

Main(string[] args)

```
namespace CSharpFundamentals
{
    class Program
    {
        static void Main(string[] args)
        {
            var builder = new StringBuilder("Hello World");

            builder
                .Append('-', 10)
                .AppendLine()
                .Append("Header")
                .AppendLine()
                .Append('-', 10)
                .Replace('-', '+')
                .Remove(0, 10)
                .Insert(0, new string('-', 10));

            Console.WriteLine(builder);

            Console.WriteLine("First Char: " + builder[0]);
        }
    }
}
```

Solution Explorer - CSharpFundamentals

Search Solution Explorer - CSharpFundamentals

Solution 'CSharpFundamentals' (1 project)

CSharpFundamentals

Properties

References

App.config

Program.cs

# Exercises



**Zouhair Rimale, Ph.D.**

Expert technique .NET | SharePoint | ASP.NET MVC | WEB API | Angular

# Exercises – 1, 2

1- Write a program and ask the user to enter a few numbers separated by a hyphen. Work out if the numbers are consecutive.

For example, if the input is "5-6-7-8-9" or "20-19-18-17-16", display a message: "Consecutive"; otherwise, display "Not Consecutive".

2- Write a program and ask the user to enter a few numbers separated by a hyphen.

If the user simply presses Enter, without supplying an input, exit immediately; otherwise, check to see if there are duplicates.

If so, display "Duplicate" on the console.



**Zouhair Rimale, Ph.D.**

Expert technique .NET | SharePoint | ASP.NET MVC | WEB API | Angular

## Exercises – 3, 4

3- Write a program and ask the user to enter a time value in the 24-hour time format (e.g. 19:00).

A valid time should be between 00:00 and 23:59.

If the time is valid, display "Ok"; otherwise, display "Invalid Time".

If the user doesn't provide any values, consider it as invalid time.

4- Write a program and ask the user to enter a few words separated by a space.

Use the words to create a variable name with PascalCase.

For example, if the user types: "number of students", display "NumberOfStudents".

Make sure that the program is not dependent on the input.

So, if the user types "NUMBER OF STUDENTS", the program should still display "NumberOfStudents".



**Zouhair Rimale, Ph.D.**

Expert technique .NET | SharePoint | ASP.NET MVC | WEB API | Angular



## Exercise - 5

5- Write a program and ask the user to enter an English word.  
Count the number of vowels (a, e, o, u, i) in the word.

So, if the user enters "inadequate", the program should display 6 on the console.



**Zouhair Rimale, Ph.D.**

Expert technique .NET | SharePoint | ASP.NET MVC | WEB API | Angular



# Procedural Programming



**Zouhair Rimale, Ph.D.**

Expert technique .NET | SharePoint | ASP.NET MVC | WEB API | Angular

## Procedural Programming

A programming paradigm based on procedure calls.



**Zouhair Rimale, Ph.D.**

Expert technique .NET | SharePoint | ASP.NET MVC | WEB API | Angular

# Object-oriented Programming

A programming paradigm based on objects.



**Zouhair Rimale, Ph.D.**

Expert technique .NET | SharePoint | ASP.NET MVC | WEB API | Angular

## Procedural Programming

A programming paradigm based on procedure calls.



**Zouhair Rimale, Ph.D.**

Expert technique .NET | SharePoint | ASP.NET MVC | WEB API | Angular

Program.cs

CSharpFundamentals

CSharpFundamentals.Program

Main(string[] args)

```
using System;
```

```
namespace CSharpFundamentals
```

```
{
```

```
    internal class Program
```

```
    {
```

```
        public static void Main(string[] args)
```

```
        {
```

```
            Console.Write("What's your name? ");  
            var name = Console.ReadLine();
```

```
            var array = new char[name.Length];  
            for (var i = name.Length; i > 0; i--)  
                array[name.Length - i] = name[i - 1];
```

```
            var reversed = new string(array);
```

```
            Console.WriteLine("Reversed name: " + reversed);
```

```
        }
```

```
    }
```

```
}
```

gram.cs • X

CSharpFundamentals CSharpFundamentals.Program ReverseName(string name)

```
using System;

namespace CSharpFundamentals
{
    internal class Program
    {
        public static void Main(string[] args)
        {
            Console.WriteLine("What's your name? ");
            var name = Console.ReadLine();
            var reversed = ReverseName(name);
            Console.WriteLine("Reversed name: " + reversed);
        }

        public static string ReverseName(string name)
        {
            var array = new char[name.Length];
            for (var i = name.Length; i > 0; i--)
                array[name.Length - i] = name[i - 1];

            return new string(array);
        }
    }
}
```

**Zouhair Rimale, Ph.D.**

Expert technique .NET | SharePoint | ASP.NET MVC | WEB API | Angular

# Example 2



**Zouhair Rimale, Ph.D.**

Expert technique .NET | SharePoint | ASP.NET MVC | WEB API | Angular

Program.cs X Lists.cs

CSharpFundamentals

CSharpFundamentals.Program

Main(string[] args)

```
using System;
using System.Collections.Generic;

namespace CSharpFundamentals
{
    internal class Program
    {
        public static void Main(string[] args)
        {
            var numbers = new List<int>();

            while (true)
            {
                Console.Write("Enter a number (or 'Quit' to exit): ");
                var input = Console.ReadLine();

                if (input.ToLower() == "quit")
                    break;

                numbers.Add(Convert.ToInt32(input));
            }

            var uniques = new List<int>();
            foreach (var number in numbers)
            {
                if (!uniques.Contains(number))
                    uniques.Add(number);
            }
        }
    }
}
```



```
while (true)
{
    Console.Write("Enter a number (or 'Quit' to exit): ");
    var input = Console.ReadLine();

    if (input.ToLower() == "quit")
        break;

    numbers.Add(Convert.ToInt32(input));
}
```

```
var uniques = new List<int>();
foreach (var number in numbers)
{
    if (!uniques.Contains(number))
        uniques.Add(number);
}
```

```
Console.WriteLine("Unique numbers:");
foreach (var number in uniques)
    Console.WriteLine(number);
}
```

```
Console.Write("Enter a number (or 'quit' to exit): ");  
var input = Console.ReadLine();  
  
if (input.ToLower() == "quit")  
    break;  
  
numbers.Add(Convert.ToInt32(input));  
}
```

```
Console.WriteLine("Unique numbers:");  
foreach (var number in GetUniqueNumbers(numbers))  
    Console.WriteLine(number);  
}
```

```
public static List<int> GetUniqueNumbers(List<int> numbers)  
{  
    var uniques = new List<int>();  
    foreach (var number in numbers)  
    {  
        if (!uniques.Contains(number))  
            uniques.Add(number);  
    }  
  
    return uniques;  
}
```



