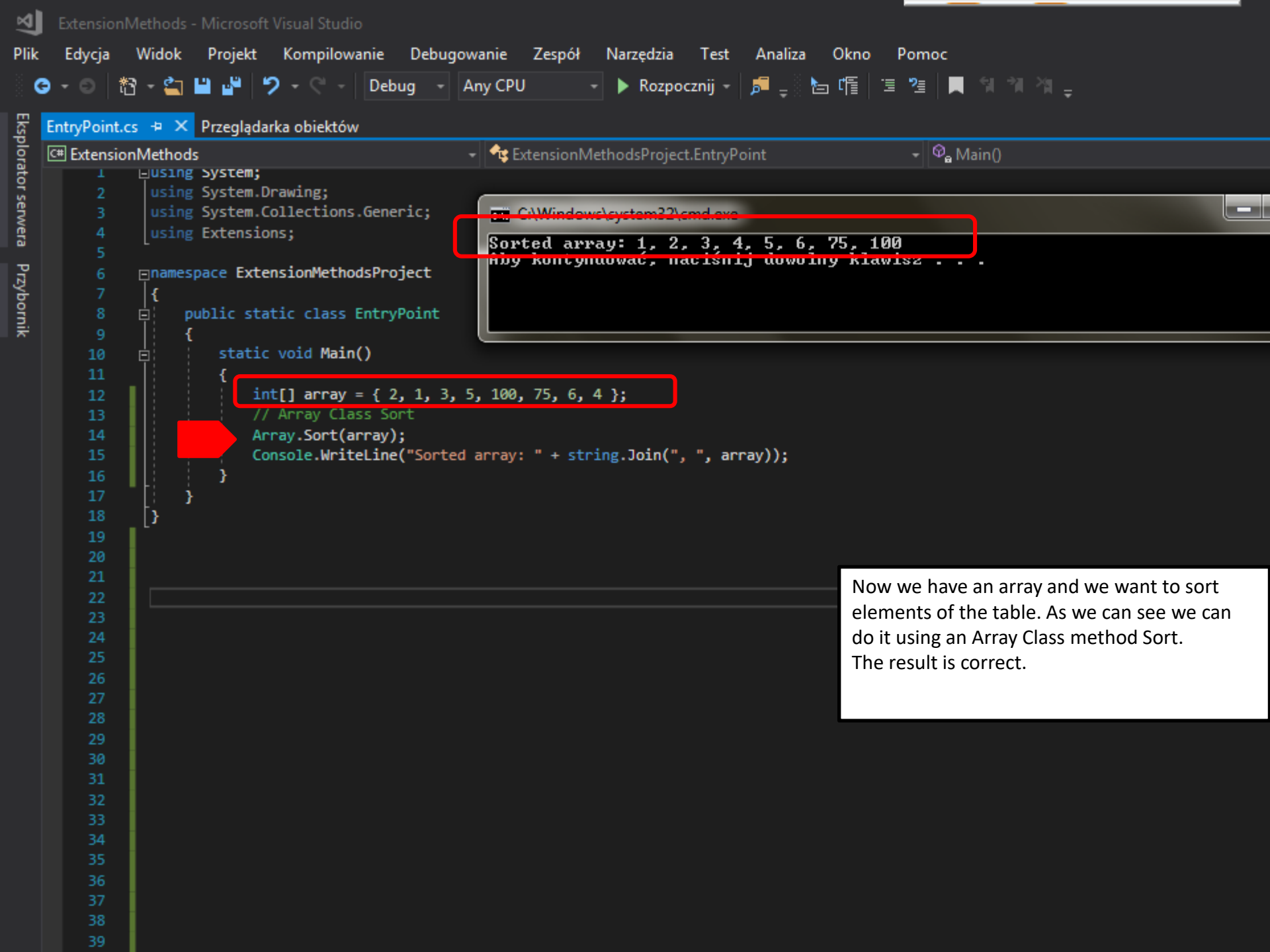
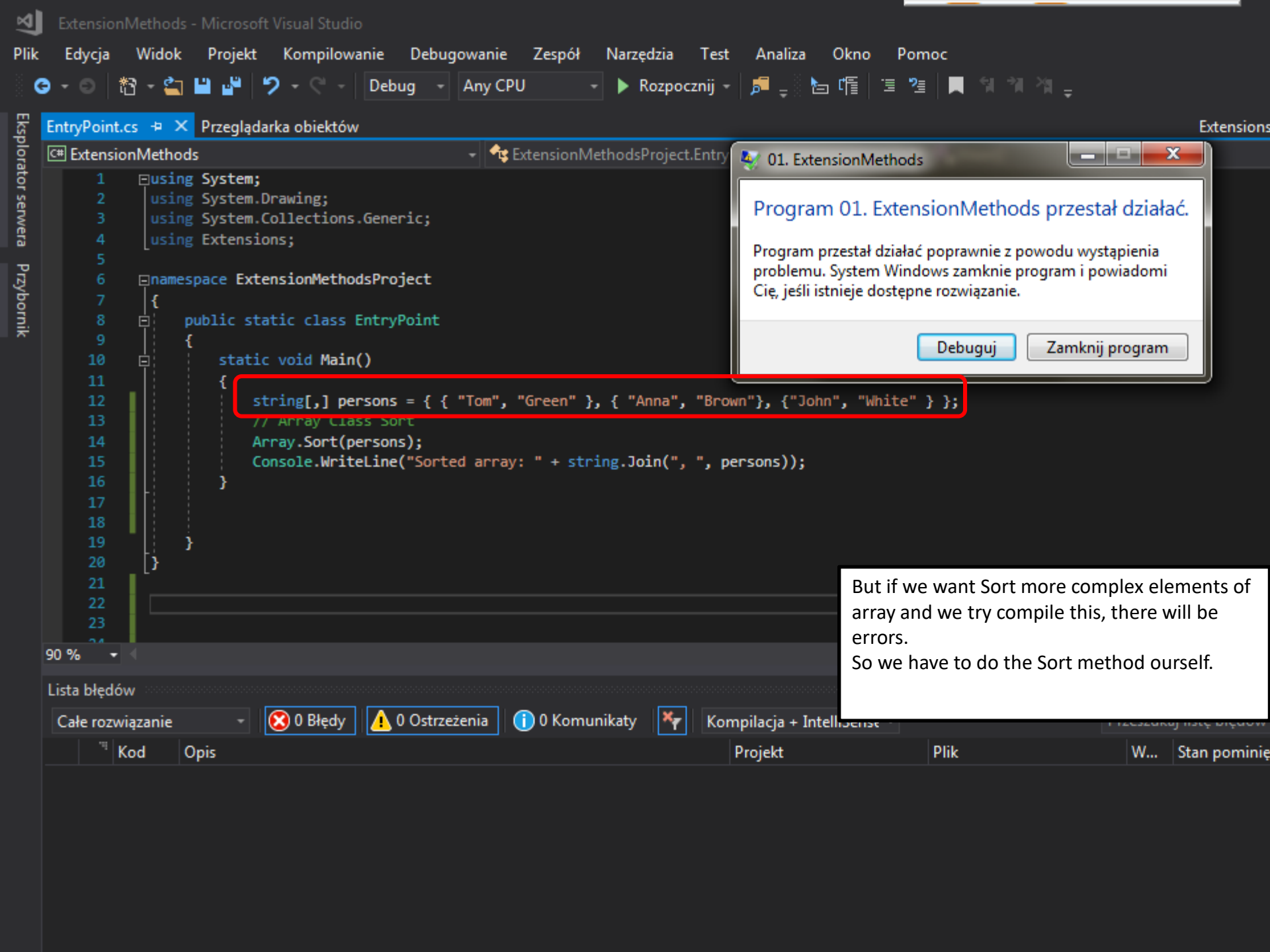
Extension methods



Now we have an array and we want to sort elements of the table. As we can see we can do it using an Array Class method Sort.
The result is correct.



But if we want Sort more complex elements of array and we try compile this, there will be errors.
So we have to do the Sort method ourself.

ExtensionMethods - Microsoft Visual Studio

Plik Edycja Widok Projekt Kompilowanie Debugowanie Zespół Narzędzia Test Analiza Okno Pomoc

Debug | Any CPU | Rozpocznij

EntryPoint.cs | Przeglądarka obiektów

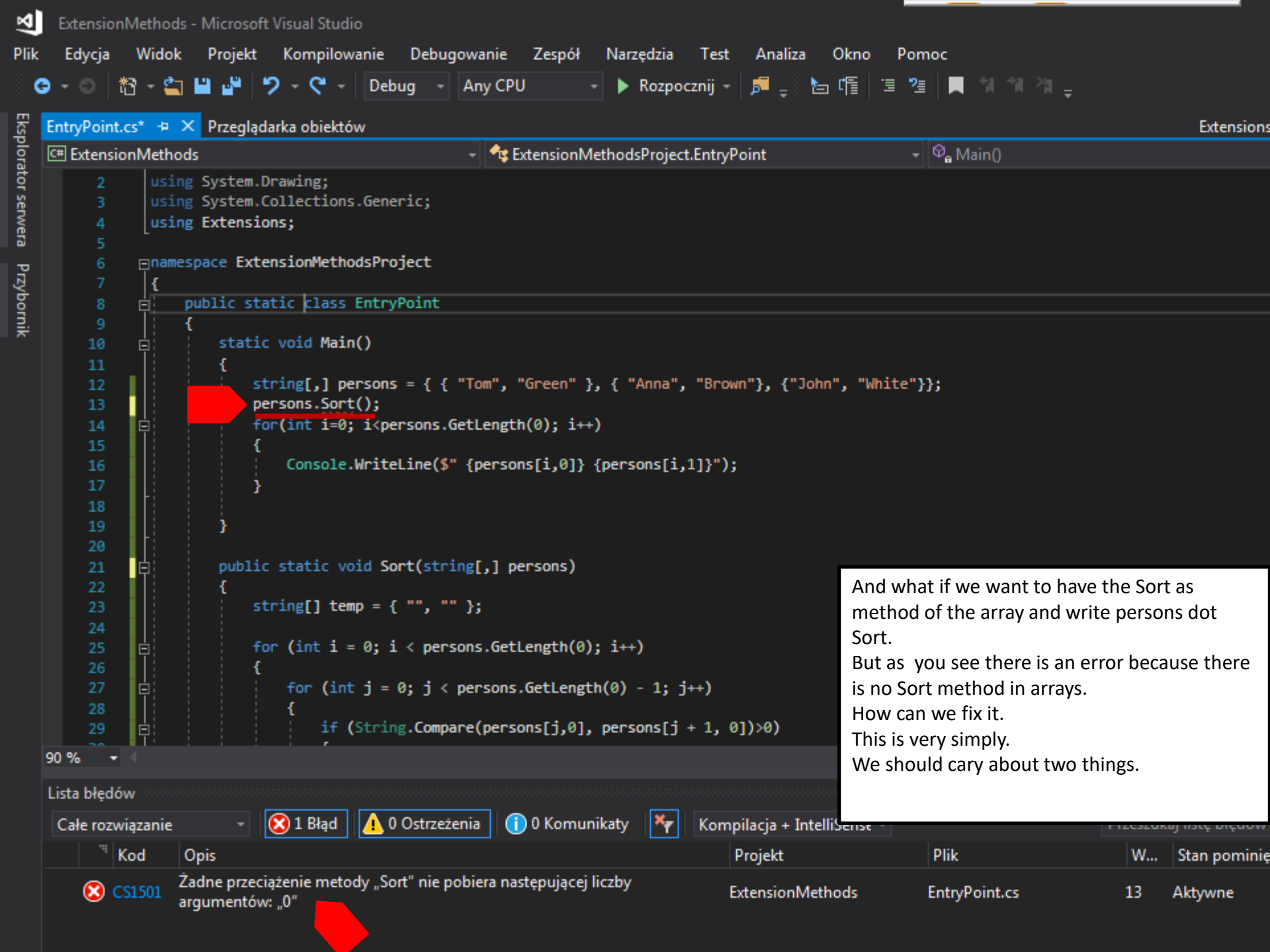
ExtensionMethods | ExtensionMethodsProject.EntryPoint | Sort(string[,] persons)

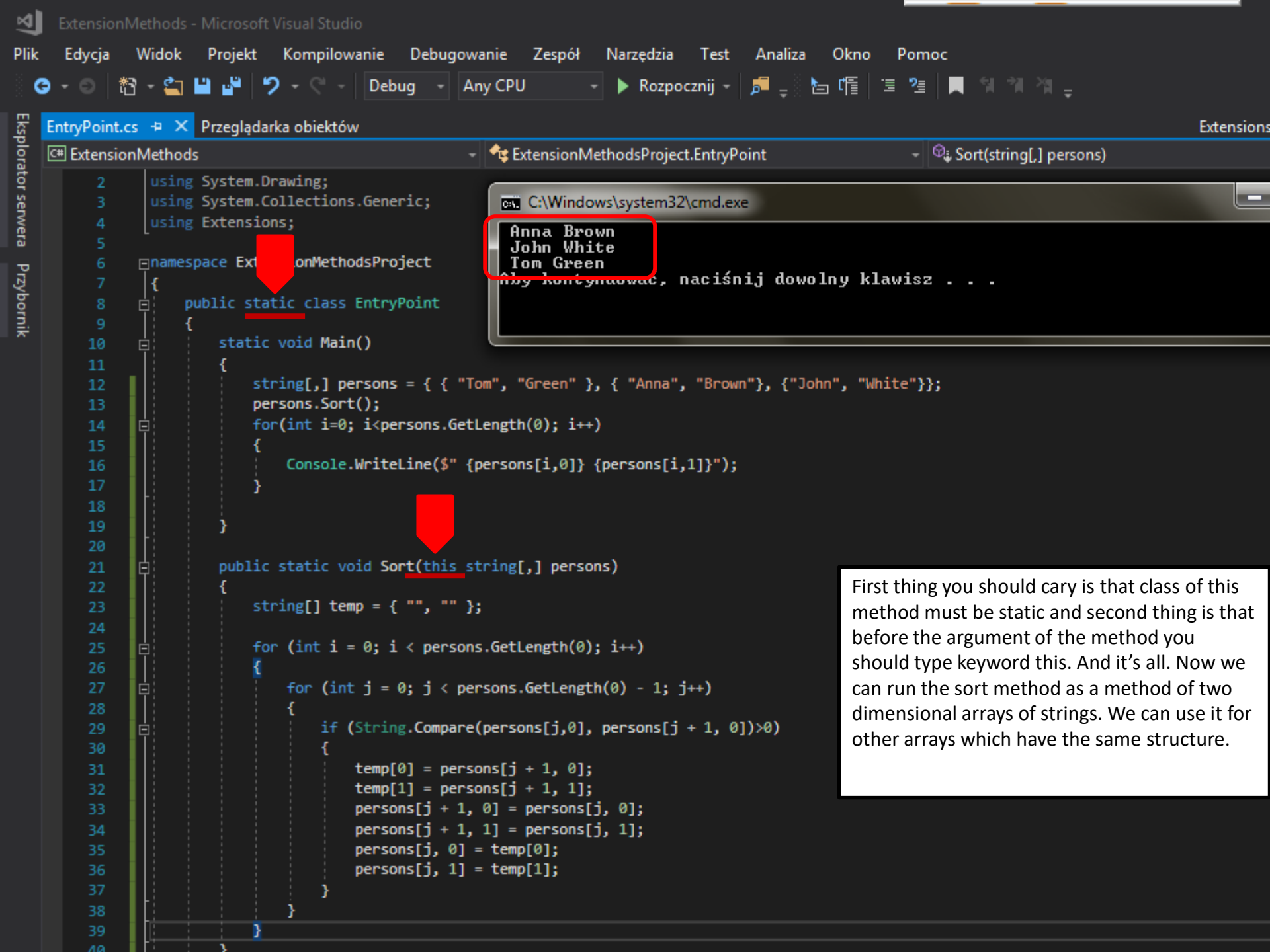
```
using System;
using System.Drawing;
using System.Collections.Generic;
using Extensions;

namespace ExtensionMethodsProject
{
    public static class EntryPoint
    {
        static void Main()
        {
            string[,] persons = { { "Tom", "Green" }, { "Anna", "Brown"}, {"John", "White"}};
            Sort(persons);
            for(int i=0; i<persons.GetLength(0); i++)
            {
                Console.WriteLine($" {persons[i,0]} {persons[i,1]}");
            }

        }

        public static void Sort(string[,] persons)
        {
            string[] temp = { "", "" };

            for (int i = 0; i < persons.GetLength(0); i++)
            {
                for (int j = 0; j < persons.GetLength(0) - 1; j++)
                {
                    if (String.Compare(persons[j,0], persons[j + 1, 0])>0)
                    {
                        temp[0] = persons[j + 1, 0];
                        temp[1] = persons[j + 1, 1];
                        persons[j + 1, 0] = persons[j, 0];
                        persons[j + 1, 1] = persons[j, 1];
                        persons[j, 0] = temp[0];
                        persons[j, 1] = temp[1];
                    }
                }
            }
```

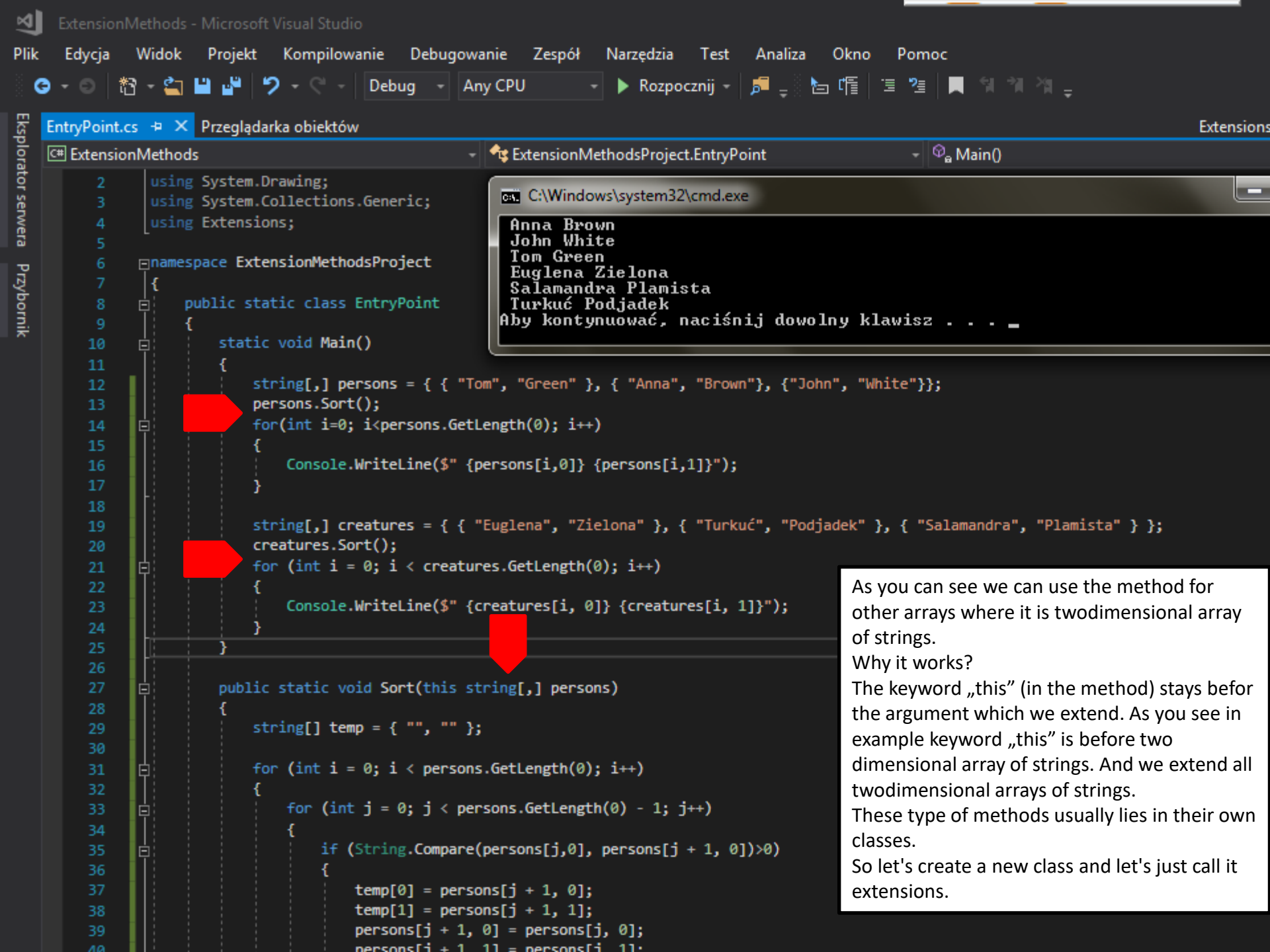
C:\Windows\system32\cmd.exe

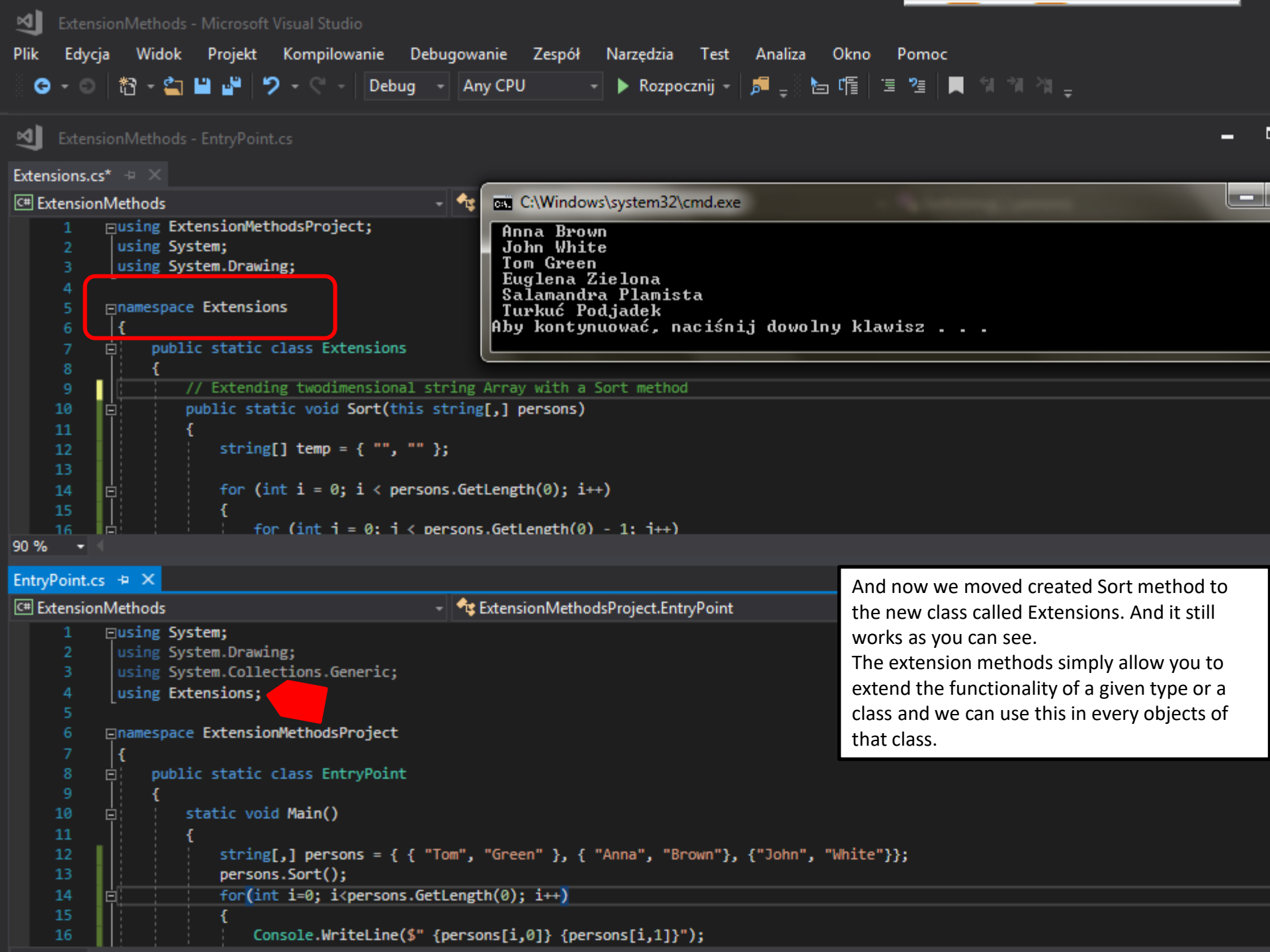
```
Anna Brown
John White
Tom Green
Aby kontynuować, naciśnij dowolny klawisz . . .
```

And here we have a bubble sort method which compares persons and sort then by the first name.
And the result: Anna Brown, John White and Tom Green. You can see that they are sorted


ExtensionMethods - Microsoft Visual Studio
Plik Edycja Widok Projekt Kompilowanie Debugowanie Zespół Narzędzia Test Analiza Okno Pomoc
Debug
Any CPU
Rozpocznij
EntryPoint.cs*
Przeglądarka obiektów
ExtensionMethods
ExtensionMethodsProject.EntryPoint
Main()
using System.Drawing;
using System.Collections.Generic;
using Extensions;
namespace ExtensionMethodsProject
{
public static class EntryPoint
{
static void Main()
{
string[,] persons = { { "Tom", "Green" }, { "Anna", "Brown"}, {"John", "White"}};
persons.Sort();
for(int i=0; i<persons.GetLength(0); i++)
{
Console.WriteLine($" {persons[i,0]} {persons[i,1]}");
}
}
public static void Sort(string[,] persons)
{
string[] temp = { "", "" };
for (int i = 0; i < persons.GetLength(0); i++)
{
for (int j = 0; j < persons.GetLength(0) - 1; j++)
{
if (String.Compare(persons[j,0], persons[j + 1, 0])>0)
90 %
Lista błędów
Całe rozwiązanie
1 Błąd
0 Ostrzeżenia
0 Komunikaty
Kompilacja + IntelliSense
Kod
Opis
Projekt
Plik
W...
Stan pominię
CS1501
Żadne przeciążenie metody „Sort" nie pobiera następującej liczby argumentów: „0"
ExtensionMethods
EntryPoint.cs
13
Aktywne
And what if we want to have the Sort as method of the array and write persons dot Sort.
But as you see there is an error because there is no Sort method in arrays.
How can we fix it.
This is very simply.
We should cary about two things.


ExtensionMethods - Microsoft Visual Studio
Plik Edycja Widok Projekt Kompilowanie Debugowanie Zespół Narzędzia Test Analiza Okno Pomoc
Debug
Any CPU
Rozpocznij
EntryPoint.cs
Przeglądarka obiektów
ExtensionMethods
ExtensionMethodsProject.EntryPoint
Sort(string[,] persons)
Eksplorator serwera
Przybornik
using System.Drawing;
using System.Collections.Generic;
using Extensions;
namespace ExtensionMethodsProject
{
public static class EntryPoint
{
static void Main()
{
string[,] persons = { { "Tom", "Green" }, { "Anna", "Brown"}, {"John", "White"}};
persons.Sort();
for(int i=0; i<persons.GetLength(0); i++)
{
Console.WriteLine($" {persons[i,0]} {persons[i,1]}");
}
}
public static void Sort(this string[,] persons)
{
string[] temp = { "", "" };
for (int i = 0; i < persons.GetLength(0); i++)
{
for (int j = 0; j < persons.GetLength(0) - 1; j++)
{
if (String.Compare(persons[j,0], persons[j + 1, 0])>0)
{
temp[0] = persons[j + 1, 0];
temp[1] = persons[j + 1, 1];
persons[j + 1, 0] = persons[j, 0];
persons[j + 1, 1] = persons[j, 1];
persons[j, 0] = temp[0];
persons[j, 1] = temp[1];
}
}
}
C:\Windows\system32\cmd.exe
Anna Brown
John White
Tom Green
Aby kontynuować, naciśnij dowolny klawisz . . .
First thing you should cary is that class of this method must be static and second thing is that before the argument of the method you should type keyword this. And it's all. Now we can run the sort method as a method of two dimensional arrays of strings. We can use it for other arrays which have the same structure.


ExtensionMethods - Microsoft Visual Studio
Plik Edycja Widok Projekt Kompilowanie Debugowanie Zespół Narzędzia Test Analiza Okno Pomoc
Debug
Any CPU
Rozpocznij
EntryPoint.cs
Przeglądarka obiektów
ExtensionMethods
ExtensionMethodsProject.EntryPoint
Main()
Eksplorator serwera
Przybornik
using System.Drawing;
using System.Collections.Generic;
using Extensions;
namespace ExtensionMethodsProject
{
public static class EntryPoint
{
static void Main()
{
string[,] persons = { { "Tom", "Green" }, { "Anna", "Brown"}, {"John", "White"}};
persons.Sort();
for(int i=0; i<persons.GetLength(0); i++)
{
Console.WriteLine($" {persons[i,0]} {persons[i,1]}");
}
string[,] creatures = { { "Euglena", "Zielona" }, { "Turkuć", "Podjadek" }, { "Salamandra", "Plamista" } };
creatures.Sort();
for (int i = 0; i < creatures.GetLength(0); i++)
{
Console.WriteLine($" {creatures[i, 0]} {creatures[i, 1]}");
}
}
public static void Sort(this string[,] persons)
{
string[] temp = { "", "" };
for (int i = 0; i < persons.GetLength(0); i++)
{
for (int j = 0; j < persons.GetLength(0) - 1; j++)
{
if (String.Compare(persons[j,0], persons[j + 1, 0])>0)
{
temp[0] = persons[j + 1, 0];
temp[1] = persons[j + 1, 1];
persons[j + 1, 0] = persons[j, 0];
C:\Windows\system32\cmd.exe
Anna Brown
John White
Tom Green
Euglena Zielona
Salamandra Plamista
Turkuć Podjadek
Aby kontynuować, naciśnij dowolny klawisz . . .
As you can see we can use the method for other arrays where it is twodimensional array of strings.
Why it works?
The keyword „this” (in the method) stays befor the argument which we extend. As you see in example keyword „this” is before two dimensional array of strings. And we extend all twodimensional arrays of strings.
These type of methods usually lies in their own classes.
So let's create a new class and let's just call it extensions.


ExtensionMethods - Microsoft Visual Studio
Plik Edycja Widok Projekt Kompilowanie Debugowanie Zespół Narzędzia Test Analiza Okno Pomoc
Debug
Any CPU
Rozpocznij
ExtensionMethods - EntryPoint.cs
Extensions.cs*
ExtensionMethods
using ExtensionMethodsProject;
using System;
using System.Drawing;
namespace Extensions
{
public static class Extensions
{
// Extending twodimensional string Array with a Sort method
public static void Sort(this string[,] persons)
{
string[] temp = { "", "" };
for (int i = 0; i < persons.GetLength(0); i++)
{
for (int j = 0; j < persons.GetLength(0) - 1; j++)
C:\Windows\system32\cmd.exe
Anna Brown
John White
Tom Green
Euglena Zielona
Salamandra Plamista
Turkuć Podjadek
Aby kontynuować, naciśnij dowolny klawisz . . .
90 %
EntryPoint.cs
ExtensionMethods
ExtensionMethodsProject.EntryPoint
using System;
using System.Drawing;
using System.Collections.Generic;
using Extensions;
namespace ExtensionMethodsProject
{
public static class EntryPoint
{
static void Main()
{
string[,] persons = { { "Tom", "Green" }, { "Anna", "Brown"}, {"John", "White"}};
persons.Sort();
for(int i=0; i<persons.GetLength(0); i++)
{
Console.WriteLine($" {persons[i,0]} {persons[i,1]}");
And now we moved created Sort method to the new class called Extensions. And it still works as you can see.
The extension methods simply allow you to extend the functionality of a given type or a class and we can use this in every objects of that class.

Multiple argument extension methods

Ok. And what if we want sort the array of persons but with second name. We can use second argument of the sort method to set this information.

ExtensionMethods - EntryPoint.cs

Extensions.cs

ExtensionMethods | Extensions.Extensions | Sort(s

```
1   using ExtensionMethodsProject;
2   using System;
3   using System.Drawing;
4
5   namespace Extensions
6   {
7       public static class Extensions
8       {
9           // Extending twodimensional string Array with a Sort method
10          public static void Sort(this string[,] persons, int index)
11          {
12              string[] temp = { "", "" };
13
14              for (int i = 0; i < persons.GetLength(0); i++)
15              {
16                  for (int j = 0; j < persons.GetLength(0) - 1; j++)
17                  {
18                      if (String.Compare(persons[j, index], persons[j + 1, index]) > 0)
19                      {
```

```
C:\Windows\system32\cmd.exe
Anna Brown
John White
Tom Green
Salamandra Plamista
Turkuć Podjadek
Euglena Zielona
Aby kontynuować, naciśnij dowolny klawisz . . .
```

90 %

EntryPoint.cs

ExtensionMethods | ExtensionMethodsProject.EntryPoint | Main

```
11          {
12              string[,] persons = { { "Tom", "Green" }, { "Anna", "Brown"}, {"John", "White"}};
13              persons.Sort(0);
14              for(int i=0; i<persons.GetLength(0); i++)
15              {
16                  Console.WriteLine($" {persons[i,0]} {persons[i,1]}");
17              }
18
19              string[,] creatures = { { "Euglena", "Zielona" }, { "Turkuć", "Podjadek" }, { "Salamandra", "Plamista" } };
20              creatures.Sort(1);
21              for (int i = 0; i < creatures.GetLength(0); i++)
22              {
23                  Console.WriteLine($" {creatures[i, 0]} {creatures[i, 1]}");
24              }
25          }
26
27
28
```

It's not a big problem. We should modify the bubble sort method in three places and we can use the extension sort method with an argument.

We use int index as second parameter of method and we changed Compare condition in two points. As you can see in the example the first array is sorted by first string in every items and second array is sorted by second string (second name).

ExtensionMethods - Extensions.cs

Extensions.cs

ExtensionMethods | Extensions.Extensions | Sort(

```
1   using ExtensionMethodsProject;
2   using System;
3   using System.Drawing;
4
5   namespace Extensions
6   {
7       public static class Extensions
8       {
9           // Extending twodimensional string Array with a Sort method
10          public static void Sort(this string[,] persons, int index = 0)
11          {
12              string[] temp = { "", "" };
13
14              for (int i = 0; i < persons.GetLength(0); i++)
15              {
16                  for (int j = 0; j < persons.GetLength(0) - 1; j++)
17                  {
18                      if (String.Compare(persons[j, index], persons[j + 1, index]) > 0)
19                      {
```

C:\Windows\system32\cmd.exe

```
Anna Brown
John White
Tom Green
Euglena Zielona
Salamandra Plamista
Turkuć Podjadek
Aby kontynuować, naciśnij dowolny klawisz . . .
```

90 %

EntryPoint.cs

ExtensionMethods | ExtensionMethodsProject.EntryPoint | Main

```
11          {
12              string[,] persons = { { "Tom", "Green" }, { "Anna", "Brown"}, {"John", "White"}};
13              persons.Sort();
14              for(int i=0; i<persons.GetLength(0); i++)
15              {
16                  Console.WriteLine($" {persons[i,0]} {persons[i,1]}");
17              }
18
19              string[,] creatures = { { "Euglena", "Zielona" }, { "Turkuć", "Podjadek" }, { "Salamandra", "Plamista" } };
20              creatures.Sort(0);
21              for (int i = 0; i < creatures.GetLength(0); i++)
22              {
23                  Console.WriteLine($" {creatures[i, 0]} {creatures[i, 1]}");
24              }
25          }
26
27
28
```

If we want make the second argument optional we should write a default value for it. And there is value of 0. And now we can try to run sort with argument 0 and without arguments. As you can see the result is the array sorted by first name in both cases.

ExtensionMethods - EntryPoint.cs

Extensions.cs

ExtensionMethods | Extensions.Extensions | Sort(i

```
37          }
38      }
39
40      // Extending twodimensional string Array with Reverse method
41      public static void Reverse(this string[,] array)
42      {
43          string[] temp = { "", "" };
44          for (int i = 0, j = array.GetLength(0)-1; i < j; i++, j--)
45          {
46              temp[0] = array[i, 0];
47              temp[1] = array[i, 1];
48              array[i, 0] = array[j, 0];
49              array[i, 1] = array[j, 1];
50              array[j, 0] = temp[0];
51              array[j, 1] = temp[1];
52          }
53      }
54
55
```

90 %

We can make another method that will reverse the order of persons. It changes first item with the last, second with one before last and so other items.
We execute the reverse method and see that it works – all elements are in reverse order.
Ok. But what we should do if we want sort an array and optionally reverse the order of it.

C:\Windows\System32\cmd.exe

```
John White
Anna Brown
Tom Green
Salamandra Plamista
Turkuć Podjadek
Euglena Zielona
Aby kontynuować, naciśnij dowolny klawisz . . .
```

EntryPoint.cs

ExtensionMethods

```
9       {
10          static void Main()
11          {
12              string[,] persons = { { "Tom", "Green" }, { "Anna", "Brown"}, {"John", "White"}};
13              persons.Reverse();
14              for(int i=0; i<persons.GetLength(0); i++)
15              {
16                  Console.WriteLine($" {persons[i, 0]} {persons[i, 1]}");
17              }
18
19              string[,] creatures = { { "Euglena", "Zielona" }, { "Turkuć", "Podjadek" }, { "Salamandra", "Plamista" } };
20              creatures.Reverse();
21              for (int i = 0; i < creatures.GetLength(0); i++)
22              {
23                  Console.WriteLine($" {creatures[i, 0]} {creatures[i, 1]}");
24              }
25          }
26
```

ExtensionMethods - EntryPoint.cs

Extensions.cs

ExtensionMethods | Extensions.Extensions | Sort(

```
7   public static class Extensions
8   {
9       // Extending twodimensional string Array with a Sort method
10      public static void Sort(this string[,] persons, int index = 0, bool reverse = false)
11      {
12          persons.Sort(index);
13          if (reverse)
14          {
15              persons.Reverse();
16          }
17      }
18
19      public static void Sort(this string[,] persons, int index = 0)
20      {
21          string[] temp = { "", "" };
22
23          for (int i = 0; i < persons.GetLength(0); i++)
24          {
25              for (int j = 0; j < persons.GetLength(0) - 1; j++)
```

We can overload the sort method. As you can see there are two Sort methods. They are different because they have different arguments.
When you execute the method. It will be executed method that has exactly all the same type of arguments.

```
C:\Windows\system32\cmd.exe
Tom Green
John White
Anna Brown
Salamandra Plamista
Turkuć Podjadek
Euglena Zielona
Aby kontynuować, naciśnij dowolny klawisz . . .
```

90 %

EntryPoint.cs

ExtensionMethods | ExtensionMethodsProject.EntryPoint | Main

```
4   using Extensions;
5
6   namespace ExtensionMethodsProject
7   {
8       public static class EntryPoint
9       {
10          static void Main()
11          {
12              string[,] persons = { { "Tom", "Green" }, { "Anna", "Brown"}, {"John", "White"}};
13              persons.Sort(0, true);
14              for(int i=0; i<persons.GetLength(0); i++)
15              {
16                  Console.WriteLine($" {persons[i, 0]} {persons[i, 1]}");
17              }
18
19              string[,] creatures = { { "Euglena", "Zielona" }, { "Turkuć", "Podjadek" }, { "Salamandra", "Plamista" } };
20              creatures.Sort(1);
21              for (int i = 0; i < creatures.GetLength(0); i++)
```

Now we call the Sort method firstly for first name and descending order (first implementation of method).
And secondly for second name and without argument about order (second implementation of the method).
You can pass multiple arguments and you can overload these methods.

Extending classes that we can’t modify

Another example - we're going to extend a class.
We can use the same approach to extend any Classes and sometimes it makes sense sometimes it doesn't.
Sometimes it's just better and easier to simply create the method inside that class but not always you have access to the class to the extent.
Especially if you're using frameworks or libraries from other people.

ExtensionMethods - EntryPoint.cs

Extensions.cs

```
ExtensionMethods                                   Extensions.
85
86
87          // Extending the built in Point class
88          public static Distance DistanceTo(this Point p1, Point p2)
89          {
90              Console.WriteLine($"The distance between P1 and P2 is " +
91                  $"\n{p2.X - p1.X} units in the X direction" +
92                  $"\n{p2.Y - p1.Y} units in the Y direction");
93
94              return new Distance() { XDistance = p2.X - p1.X, YDistance = p2.Y - p1.Y };
95          }
96      }
97  }
```

90 %

Let's take the Point class for example.
It's a built in class in the C-sharp that we can use but we don't have access to it because we need a special reference for it. System dot Drawing.
Point is simply a class that allow us to create points in space.
Point is simply a class that lets us save x and y coordinates.
We created two points – first of coordinates 20, 20 and second of coordinates 50 and 60.

And now let's extend the point class and give it a method that tells us what's the distance between two points.
So for this purpose it has been created one new class called distance.
It simply stores two values x distance and y distance.
So it was created a new method named DistanceTo. This method will return a distance - an object of class Distance.

EntryPoint.cs

```
ExtensionMethods
1   using System;
2   using System.Drawing;
3   using System.Collections.Generic;
4   using Extensions;
5
6   namespace ExtensionMethodsProject
7   {
8       public static class EntryPoint
9       {
10          static void Main()
11          {
12              Point p1 = new Point(20, 20);
13              Point p2 = new Point(50, 60);
14
15              // Distance between p1 and p2, saved in a Distance object and printed on the console
16              Distance dist = p1.DistanceTo(p2);
17          }
18      }
19  }
20
```

Distance.cs

```
ExtensionMethods    ExtensionMethodsP    XDistance
1   namespace ExtensionMethodsProject
2   {
3       public class Distance
4       {
5           public int XDistance { get; set; }
6
7           public int YDistance { get; set; }
8       }
9   }
```

90 %

And in this object we have two properties Xdistance and Ydistance
So we created a DistanceTo method. It has two points as arguments. Before first argument we write keyword this – to extend class Point.
First we're going to print it on the console.
So then we compute the distance between point 1 and point 2 as “Xdistance = point2.x - point1.x”; and the same way we compute Ydistance.

This is simply a message that you're going to get on the Console.

```
C:\Windows\system32\cmd.exe
The distance between P1 and P2 is
30 units in the X direction
40 units in the Y direction
Aby kontynuować, naciśnij dowolny klawisz . . .
```

And then we return a new distance and we give a value to the x distance and y distance.
All right. We execute the method writing a DistanceTo as a method of point class. The method is working as you can see that.

End of extension methods

And this is end of extension methods. Sometimes it's very useful to be able to create an extension method. And it’s all.