

C#-4

The screenshot shows the Microsoft Visual Studio IDE interface. In the center is a code editor window titled "LocalDemo.cs" under the project "ConsoleApp". The code is as follows:

```
4  using System.Text;
5  using System.Threading.Tasks;
6
7  namespace ConsoleApp
8  {
9      class LocalDemo
10     {
11         static void Main()
12         {
13             int i = 100;
14             for (int i = 0; i < 10; i++)
15             {
16                 Console.WriteLine(i);
17             }
18         }
19     }
20 }
```

A red arrow points from the text "here we got error but when we try to do this in javascript we'll not get the error" to the line "int i = 100;".

A red arrow points from the text "the error is saying that the local name is already there" to the line "for (int i = 0; i < 10; i++)".

A callout box highlights the error at line 14: "CS0136: A local or parameter named 'i' cannot be declared in this scope because that name is used in an enclosing local scope to define a local or parameter".

**this variable is local to this particular method
it can be access inside this method not
outside this method so when the execution comes
out of this method this variable will lost its value**

Statement Blocks

Use braces As
block
delimiters

A block and its parent
block cannot have a
variable with
the same name

Sibling blocks can have
variables with
the same name

```
{  
    // code  
}
```

```
{  
    int i;  
    ...  
    {  
        int i;  
        ...  
    }  
}
```

```
{  
    int i;  
    ...  
}  
...  
{  
    int i;  
    ...  
}
```

Scope of the Variable

- A field of a class is in scope as long as its containing class is in scope
- A local variable is in scope until a closing brace indicates the end of block statement in which its declared.
- A local variable that is declared in *for*, *while* or similar statement is in scope in the body of that loop.

```
public static void Main()
{
    int x; // known to all code within Main()

    x = 10;
    if (x == 10)
    { // start new scope
        int y = 20; // known only to this block

        // x and y both known here.
        Console.WriteLine("x and y: " + x + " " + y);
        x = y * 2;
    }
    // y = 100; // Error! y not known here

    // x is still known here.
    Console.WriteLine("x is " + x);
```

Returning Values from Methods

- Declare the method with non-void type
- Add a return statement with an expression
 - Sets the return value
 - Returns to caller
- Non-void methods must return a value

```
static int TwoPlusTwo() {  
    int a,b;  
    a = 2;  
    b = 2;  
    return a + b;  
}
```

```
int x;  
x = TwoPlusTwo();  
Console.WriteLine(x);
```

Declaring and Calling Parameters

- Declaring parameters
 - Place between parentheses after method name
 - Define type and name for each parameter
- Calling methods with parameters
 - Supply a value for each parameter

```
static void MethodWithParameters(int n, string y)  
{ ... }
```

```
MethodWithParameters(2, "Hello, world");
```

When you are declaring and calling a particular parameter you can use **in parameter** and **out parameter**

Mechanisms for Passing Parameters

- Three ways to pass parameters

in	Pass by value
in out	Pass by reference
out	Output parameters

That is called pass by value and pass by reference

Pass by Value

- Default mechanism for passing parameters:
 - Parameter value is copied
 - Variable can be changed inside the method
 - Has no effect on value outside the method
 - Parameter must be of the same type or compatible type

```
static void AddOne(int x)
{
    x++; // Increment x
}
static void Main( )
{
    int k = 6;
    AddOne(k);
    Console.WriteLine(k); // Display the value 6, not 7
}
```

Pass by Reference

- What are reference parameters?
 - A reference to memory location
- Using reference parameters
 - Use the **ref** keyword in method declaration and call
 - Match types and variable values
 - Changes made in the method affect the caller
 - Assign parameter value before calling the method

```
ConsoleApp.InOutDemo

0 references
static void Main()
{
    int no1 = 10, no2 = 20;
    Console.WriteLine("Before Swapping");
    Console.WriteLine("No1 = " + no1 + "\tNo2 = " + no2);
    Swap(no1, no2);
    Console.WriteLine("After Swapping");
    Console.WriteLine("No1 = " + no1 + "\tNo2 = " + no2);
}

1 reference
private static void Swap(int no1, int no2)
{
    no1 = no1 + no2;
    no2 = no1 - no2;
    no1 = no1 - no2;
    Console.WriteLine("After Swapping");
    Console.WriteLine("No1 = " + no1 + "\tNo2 = " + no2);
}
```

this are the local variables and can be accessed only in this
this method

so now after i want to pass the swapped
data to main

so in C How will you do this -----return -----no-----return
can return only one value we cannot return more than one value
In C using pointer we can do this

In C we pass the Address here

```
ConsoleApp.InOutDemo
0 references
static void Main()
{
    int no1 = 10, no2 = 20;
    Console.WriteLine("Before Swapping");
    Console.WriteLine("No1 = " + no1 + "\tNo2 = " + no2);
    Swap(&no1, &no2);
    Console.WriteLine("After Swapping");
    Console.WriteLine("No1 = " + no1 + "\tNo2 = " + no2);
}

1 reference
private static void Swap(int *no1, int *no2)
{
    no1 = no1 + no2;
    no2 = no1 - no2;
    no1 = no1 - no2;
    //Console.WriteLine("After Swapping");
    //Console.WriteLine("No1 = " + no1 + "\tNo2 = " + no2);
}
```

But here we don't have a option of pointer So we have a other option here

we can always say **ref**

```
ConsoleApp.cs  X | ConsoleApp
ConsoleApp.cs  ConsoleApp.InOutDemo
0 references
1 static void Main()
2 {
3     int no1 = 10, no2 = 20;
4     Console.WriteLine("Before Swapping");
5     Console.WriteLine("No1 = " + no1 + "\tNo2 = " + no2);
6     Swap(ref no1, ref no2);
7     Console.WriteLine("After Swapping");
8     Console.WriteLine("No1 = " + no1 + "\tNo2 = " + no2);
9 }
0
1 reference
1 [Pencil] static void Swap(ref int no1, ref int no2)
2 {
3     no1 = no1 + no2;
4     no2 = no1 - no2;
5     no1 = no1 - no2;
6     //Console.WriteLine("After Swapping");
7     //Console.WriteLine("No1 = " + no1 + "\tNo2 = " + no2);
8 }
9
No issues found
```

I am saying that i am passing that reference here, Remember when you pass the reference here so we'll take the same value or name, and that number is manipulated

```
Before Swapping
No1 = 10      No2 = 20
After Swapping
No1 = 20      No2 = 10
Press any key to continue. . .
```

I can Achieve the same thing using out also but there is lot of difference between out and ref

```
localDemo.cs InOutDemo.cs* X ConsoleApp
ConsoleApp
ConsoleApp.InOutDemo
0 references
11 static void Main()
12 {
13     int no1 = 10, no2 = 20;
14     Console.WriteLine("Before Swapping");
15     Console.WriteLine("No1 = " + no1 + "\tNo2 = " + no2);
16     Swap(out no1, out no2);
17     Console.WriteLine("After Swapping");
18     Console.WriteLine("No1 = " + no1 + "\tNo2 = " + no2);
19 }
20
1 reference
21 private static void Swap(out int no1, out int no2)
22 {
23     no1 = no1 + no2; ←
24     no2 = no1 - no2;
25     no1 = no1 - no2;
26     //Console.WriteLine("After Swapping");
27     //Console.WriteLine("No1 = " + no1 + "\tNo2 = " + no2);
28 }
```

Look at this as soon as i say out it says boss
you need to initialize this
they are unassigned

The screenshot shows a Microsoft Visual Studio interface with the following details:

- Project Explorer:** Shows "ConsoleApp" as the active project.
- Solution Explorer:** Shows "ConsoleApp.InOutDemo" as the active solution.
- Code Editor:** Displays the following C# code:

```
11 static void Main()
12 {
13     int no1 = 10, no2 = 20;
14     Console.WriteLine("Before Swapping");
15     Console.WriteLine("No1 = " + no1 + "\tNo2 = " + no2);
16     Swap(out no1, out no2);
17     Console.WriteLine("After Swapping");
18     Console.WriteLine("No1 = " + no1 + "\tNo2 = " + no2);
19 }
20
21 private static void Swap(out int no1, out int no2)
22 {
23     no1 = ... no1 + ... no2;
24     no2 = no... [?] (parameter) out int no1
25     no1 = no... CS0269: Use of unassigned out parameter 'no1'
26     //Console.
27     //Console.WriteLine("No1 = " + no1 + "\tNo2 = " + no2);
28 }
```
- Tooltips:** A tooltip is displayed over the line "no1 = no... [?] (parameter) out int no1". It contains the text "[?] (parameter) out int no1" and "CS0269: Use of unassigned out parameter 'no1'".
- Status Bar:** Shows "133 %", "2", "0", and navigation icons.
- Bottom Navigation:** Shows tabs for "Output", "Package Manager Console", "Error List", and "Immediate Window".

Now this makes a difference

In REF i am having a particular value and by saying ref i am passing that particular reference to that particular variable

In Out also i am doing the same thing

```
namespace ConsoleApp
{
    class InOutDemo1
    {
        static void Main()
        {
            int no1 = 10, no2;
            InOutInit(ref no1, out no2);
            Console.WriteLine("No1 = {0} No2 = {1}", no1, no2);
        }

        private static void InOutInit(ref int no1, out int no2)
        {
            no2 = 100;
            no1 = no1 + no2;
        }
    }
}
```

```
C:\WINDOWS\system32\cmd.exe
No1 = 110 No2 = 100
Press any key to continue . . .
```

Now look at this ,this is the main difference between ref and out
In both of these cases we will always get the updated value
But In Ref 1st we have to initialize and then pass it, and you can manipulate

but when you talk about out, you have to initialize here in the method where it is passed and then you can play with it

Now where can we use something like this

The screenshot shows a Microsoft Visual Studio IDE window. The title bar indicates the project is named "ConsoleApp" and the file is "InOutDemo1.cs". The code editor displays the following C# code:

```
namespace ConsoleApp
{
    class InOutDemo1
    {
        static void Main()
        {
            int no1 = 10, no2;
            InOutInit(ref no1, out no2);
            Console.WriteLine("No1 = {0} No2 = {1}", no1, no2);
        }

        private static void InOutInit(ref int no1, out int no2)
        {
            no2 = 100;
            no1 = no1 + no2;
        }
    }
}
```

A green vertical bar on the left margin highlights the entire code block. A red rectangular box highlights the line of code "InOutInit(ref no1, out no2);". Another red rectangular box highlights the assignment statement "no2 = 100;" within the InOutInit method. The status bar at the bottom shows "133 %", "No issues found", and a small icon.

imagine there is a situation where i don't know the address of a employee so i'll pass a particular address object here and there i am trying to manipulate that address object

address will be initialized in this particular method, so this method will never bothered about address ,he will just store the employees information or customer information

there are lot of such places where you can use out ,may be like i'm creating a connection string

```
InOutDemo1.cs  ✘ X  ConsoleApp
ConsoleApp
    7  namespace ConsoleApp
    8  {
    9      class InOutDemo1
    10     {
    11         static void Main()
    12         {
    13             int no1 = 10, no2;
    14             InOutInit(ref no1, out no2);
    15             Console.WriteLine("No1 = {0} No2 = {1}", no1, no2);
    16         }
    17     }
    18     private static void InOutInit(ref int no1, out int no2)
    19     {
    20         no2 = 100;
    21         no1 = no1 + no2;
    22     }
    23
    24
    25 }
```

I'm passing a connection string from this particular code

And connection string is been assigned here

there also you can use out

This can again be used in error handing

```
7  namespace ConsoleApp
8  {
9      class InOutDemo1
10     {
11         static void Main()
12         {
13             int no1 = 10, no2;
14             InOutInit(ref no1, out no2);
15             Console.WriteLine("No1 = {0} No2 = {1}", no1, no2);
16         }
17     }
18     private static void InOutInit(ref int no1, out int no2)
19     {
20         no2 = 100;
21         no1 = no1 + no2;
22     }
23 }
24 
```

you have a particular function

you will pass the particular
Error object

and here you will manipulate Error
if you have any error in this particular function
you will manipulate that error
and put it to this out parameter

SO, after line no 14 you will check if you have any error
you will put a message saying Boss

" You have a particular error "
otherwise you will continue

Output Parameters

- What are output parameters?
 - Values are passed out but not in
- Using output parameters
 - Like **ref**, but values are not passed into the method
 - Use **out** keyword in method declaration and call

```
static void OutDemo(out int p)
{
    // ...
}
int n;
OutDemo(out n);
```

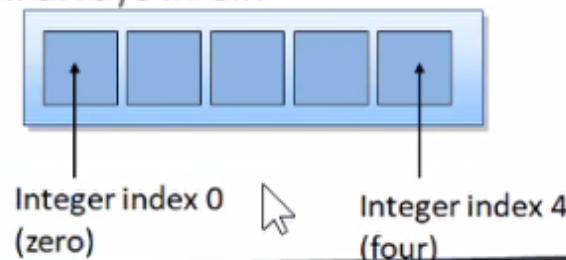
Guidelines for Passing Parameters

- Mechanisms
 - Pass by value is most common
 - Method return value is useful for single values
 - Use **ref** and/or **out** for multiple return values
 - Only use **ref** if data is transferred both ways
- Efficiency
 - Pass by value is generally the most efficient

Day9 (21-10-2021)

Array

- An array is a data structure that contains several items of the same type.
- Arrays are zero indexed: an array with n elements is indexed from 0 to n-1.
- When declaring an array, the square brackets ([]) must come after the type, not the identifier.
- Array types are reference types derived from the abstract base type `Array`. Since this type implements `IEnumerable` and `IEnumerable<T>`, you can use `foreach` iteration on all arrays in C#.



Array is the collection of Similar (Homogenous) Datatype , Which will share the same name but differs with index number , Which will starts with 0 and ends with n-1 (Where n will be the number of elements)

```
| int arr[] = {1, 'a', 2, 'b', 3};
```

All of them will be called by the same name arr arr arr arr.....

But how will you differentiate between 1 and a that's why we go with the **Index Number** which starts with **0** and ends with **n-1** (Where n will be the number of elements)

in our case we have 5 elements so index will go from 0 to 4

So,internally what happened we see example but memory will not be in the same way

```
| int arr[] = {1, 'a', 2, 'b', 3};
```

So when i write this internally its is treated as row with 5 colons in it



In Array we Have one Dimention and Multi Dimention Array

We don't have 2 dimention ,3 dimention ,4 dimention

int arr [3] [3]; -----array with 3 Rows and 3 colons in it

	0	1	2
0	1 0,0	2 0,1	3 0,2
1	4 1,0	5 1,1	6 1,2
2	7 2,0	8 2,1	9 2,2

int arr [2] [3] [3]; -----2 colons which have 3*3 boxes in it
 it is like a single dimensional array but each boxes will have 3*3 boxes in it

0			1				
0			1				
0	0,0,0 ¹	0,0,1 ²	0,0,2 ³	0	1,0,0 ¹	1,0,1 ²	1,0,2 ³
1	0,1,0 ⁴	0,1,1 ⁵	0,1,2 ⁶	1	1,1,0 ⁴	1,1,1 ⁵	1,1,2 ⁶
2	0,2,0 ⁷	0,2,1 ⁸	0,2,2 ⁹	2	1,2,0 ⁷	1,2,1 ⁸	1,2,2 ⁹

int arr [2] [2] [3] [3]; -----2 rows and 2colons each have 3*3 boxes in it

0	0	1	2	1	0	1	2
0	1	2	3	0	1	2	3
1	4	5	6	1	4	5	6
2	7	8	9	2	7	8	9
1	0	1	2	1	0	1	2
0	1	2	3	0	1	2	3
1	4	5	6	1	4	5	6
2	7	8	9	2	7	8	9

int arr [2] [2] [2] [3] [3];

		0						1			
		0	1	2	3			0	1	2	3
0	0	1	2	3		0	1	2	3		
	1	4	5	6			1	4	5	6	
	2	7	8	9			2	7	8	9	
		0	1	2	3			0	1	2	3
1	0	1	2	3		0	1	2	3		
	1	4	5	6		1	4	5	6		
	2	7	8	9		2	7	8	9		

Now to Implement each box we use **LOOP**

Why LOOP ?

now whenever you want to fetch the data ,anyway we know that our info is stored in row and colon wise

```
int arr[5] ;
arr[0] = 10;
arr[1] = 23;
arr[2] = 23;
arr[3] = 12;
arr[4] = 34;
```

now take a example so see we are getting arr[n] which is a continuous number

HOW do you get this continuous number printed ?

With the help of LOOP (that's a reason we use loops)

Array Initialization

for this in java we have 2 syntax but where as in c# we have only one

```
1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Text;
5  using System.Threading.Tasks;
6
7  namespace ConsoleApp
8  {
9      class ArrayDemo1
10     {
11         static void Main()
12         {
13             int[] arr = new int[5];
14             int arr[] = new int[5]; // this will not work
15         }
16     }
17 }
18
```

In JAVA both of them will work

but in C# this will not work

Declaration with initialization

```
1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Text;
5  using System.Threading.Tasks;
6
7  namespace ConsoleApp
8  {
9      class ArrayDemo1
10     {
11         static void Main()
12         {
13             //int[] arr = new int[5];
14             int[] arr = { 1, 2, 3, 4, 5 };
15         }
16     }
17 }
18
```

Fetch the value from array

```
1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Text;
5  using System.Threading.Tasks;
6
7  namespace ConsoleApp
8  {
9      class ArrayDemo1
10     {
11         static void Main()
12         {
13             //int[] arr = new int[5];
14             int[] arr = { 1, 2, 3, 4, 5 };
15             for (int i = 0; i < 5; i++)
16             {
17                 Console.WriteLine(arr[i]);
18             }
19         }
20     }
}
```

```
C:\WINDOWS\system32\cmd.exe
1
2
3
4
5
Press any key to continue . . .
```

now what if we add few more element here ,will this loop still work **No**
because it run from 0 to 4 only i.e only first 5 element only

.Length

A screenshot of the Microsoft Visual Studio IDE interface. The top menu bar includes File, Edit, View, Git, Project, Build, Debug, Test, Analyze, Tools, Extensions, Window, Help, and a Search bar. The title bar shows "ConsoleApp". The left sidebar has "Server Explorer" and "Toolbox" tabs. The main code editor window displays "ArrayDemo1.cs" under "ConsoleApp". The code defines a class `ArrayDemo1` with a static void `Main` method. Inside `Main`, an array `arr` is initialized with values 1 through 10. A for loop iterates over the array from index 0 to `arr.Length`. A tooltip for `Length` is shown, indicating it is a property that gets the total number of elements in all dimensions of the array. Handwritten annotations in red text and arrows highlight the word "Length" in the tooltip and the property accessors in the code.

```
9  class ArrayDemo1
10 {
11     static void Main()
12     {
13         //int[] arr = new int[5];
14         int[] arr = { 1, 2, 3, 4, 5, 6, 7, 8, 9, 10 };
15
16         for (int i = 0; i < arr.Length; i++)
17         {
18             Console.WriteLine(arr[i]);
19         }
20     }
21 }
22
23
```

so instead of writing 5 or 10 here
we have a property `arrayname.Length`

Look at the symbol it represent
Property

ArrayDemo1.cs* X ConsoleApp

ConsoleApp

ConsoleApp.ArrayDemo1

```
9 class ArrayDemo1
10 {
11     static void Main()
12     {
13         //int[] arr = new int[5];
14         int[] arr = { 1, 2, 3, 4, 5, 6, 7, 8, 9, 10 };
15
16         for (int i = 0; i < arr.Length; i++)
17         {
18             Console.WriteLine(arr[i]);
19         }
20     }
21 }
22
23
```

```
1
2
3
4
5
6
7
8
9
10
Press any key to continue . . .
```

Ab hum for me initialize karre ;condition check karre; increment karre

then value fetch karre so i am doing 4jobs at this particular time .

So, instead of using for if you want to fetch this you can always use

foreach

how foreach work?

it is the same as for now,here my i will going to have the fetched value

```
9 class ArrayDemo1
10 {
11     0 references
12     static void Main()
13     {
14         //int[] arr = new int[5];
15         int[] arr = { 1, 2, 3, 4, 5, 6 };
16         /*
17         for (int i = 0; i < arr.Length; i++)
18         {
19             Console.WriteLine(arr[i]);
20         }
21         */
22         foreach (int i in arr)
23         {
24             Console.WriteLine(i);
25         }
26     }
27 }
28 }
```

now internally this
foreach will be converted
as this for loop
to C# ne or microsoft ne hamara
ye kaam bhi aasan kar diya

```
9 class ArrayDemo1
10 {
11     0 references
12     static void Main()
13     {
14         //int[] arr = new int[5];
15         int[] arr = { 1, 2, 3, 4, 5, 6 };
16         /*
17         for (int i = 0; i < arr.Length; i++)
18         {
19             Console.WriteLine(arr[i]);
20         }
21         foreach (int i in arr)
22         {
23             Console.WriteLine(i);
24         }
25     }
26 }
27
28
```

2.this will be stored in this particular variable i

1.int i will

2.it will gone to does all this things

2.define this particular array arr.Length will be fetched ,i++ will be returned

1.and what ever we have in arr[i]

1.this is my arr from this arr

3. Which i'll gone to use in my program

Assignment 1:-

1. Search for the element (Write a program to accept 10 number from the user and ask for search key ,display the element position if the element is not found print the message element not found)(here we have to find only the 1st occurrence means if the element is found in 1st place so we print that the element is found in this place and we come out) for ex :-in the above program if user enter the search key as 1 to 1 arr[0]location pe hai ,so o/p= the element 1 is found at the position 1 and you will come outif user enter the search key as 4 to 4 arr[3]location pe hai ,so o/p= the element 4 is found at the position 4 and you will come out.....if user enter the search key as 10 so 10 is not there in our array ,so o/p= element 10 is not found
2. write a program to find all the occurrence (for ex:- agar 1 arr[0] pe bhi hai and arr[5] pe bhi hai then o/p= the element 1 is found at the position 1 and 6 and you will come out).....if user enter the search key as 10 so 10 is not there in our array ,so o/p= element 10 is not found

Multi Dimensional Array

1. Whenever you want to create multidimensional array
you should always put comma

arraytype [,] array name = new arraytype[,]

```
1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Text;
5  using System.Threading.Tasks;
6
7  namespace ConsoleApp
8  {
9      class ArrayDemo2
10     {
11         static void Main()
12         {
13             //2 -dim array
14             int[,] matrix = new int[3, 3]
15             {
16                 {1,2,3 },
17                 {4,5,6 },
18                 {7,8,9 }
19             };
20         }
21     }
}
```

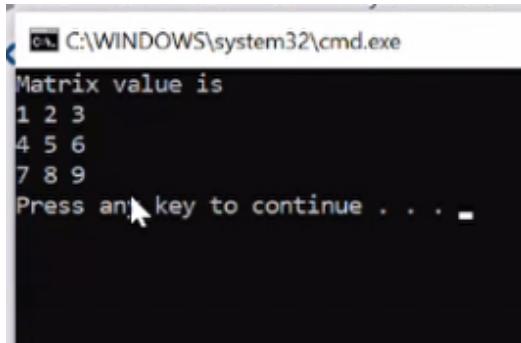
3 rows and
3 colons

initialization

How to Fetch the data from this ?

```
10  {
11      0 references
12      static void Main()
13      {
14          //2 -dim array
15          int[,] matrix = new int[3, 3]
16          {
17              { 1, 2, 3 },
18              { 4, 5, 6 },
19              { 7, 8, 9 }
20          };
21          Console.WriteLine("Matrix value is");
22          for (int i = 0; i < 3; i++) ← this will get me the rows
23          {
24              for (int j = 0; j < 3; j++) ← this will give me colons
25              {
26                  Console.Write(matrix[i,j] + " "); ← I'll give some space
27              }
28              Console.WriteLine(); ← it will print me the value
29          }                                of matrix[i,j]
30      }
```

to
print the content in the next line



C:\WINDOWS\system32\cmd.exe

Matrix value is

1	2	3
4	5	6
7	8	9

Press any key to continue . . .

Assignment 2 :

1. Write a program to print 1st row element ,last row element , 1st colon element ,last colon element ,both diagonal element .

With array what we have created has some methods Explore them like

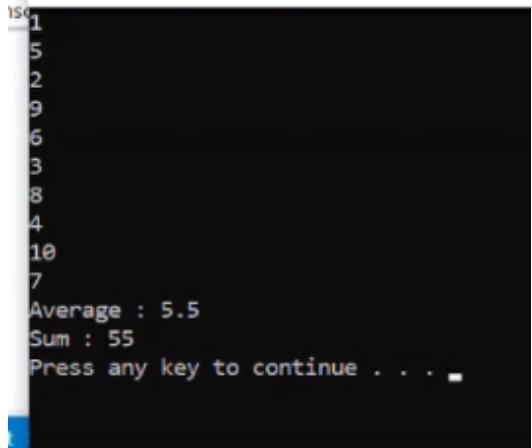
arrayname.Average()-----returns average of array elements

arrayname.Sum()-----returns sum of array elements

and may more we have pls explore

A screenshot of the Microsoft Visual Studio IDE interface. The main window shows a code editor with the file `ArrayDemo1.cs` open. The code defines a class `ArrayDemo1` with a `Main` method. The method initializes an array with values 1, 5, 2, 9, 6, 3, 8, 4, 10, 7, and then prints each element to the console. It also calculates and prints the average and sum of the array elements. The code editor features color-coded syntax highlighting and a vertical green bar indicating the current line of code. The status bar at the bottom indicates "No issues found".

```
class ArrayDemo1
{
    static void Main()
    {
        int[] arr = { 1, 5, 2, 9, 6, 3, 8, 4, 10, 7 };
        /*
        for (int i = 0; i < arr.Length; i++)
        {
            Console.WriteLine(arr[i]);
        }
        */
        foreach (int i in arr)
        {
            Console.WriteLine(i);
        }
        Console.WriteLine("Average : " + arr.Average());
        Console.WriteLine("Sum : " + arr.Sum());
    }
}
```



```
1
5
2
9
6
3
8
4
10
7
Average : 5.5
Sum : 55
Press any key to continue . . .
```

and With array we have some methods

Array.sort(arrayname); -----return the sorted array.

Array.Reverse(arrayname);

and may more we have pls explore

So you need to figure out what are the other methods what **arr.Average()** as an object will gone to support and What are the different methods which

Array.Sort() Array will gone to support

A screenshot of a C# code editor window titled "ConsoleApp". The code is named "ArrayDemo1". The code itself is as follows:

```
14     int[] arr = { 1, 2, 3, 2, 3, 0, 2, 0, 4, 10, 5, };
15     /*
16      for (int i = 0; i < arr.Length; i++)
17      {
18          Console.WriteLine(arr[i]);
19      }
20      */
21     foreach (int i in arr)
22     {
23         Console.WriteLine(i);
24     }
25     Console.WriteLine("Average : " + arr.Average());
26     Console.WriteLine("Sum : " + arr.Sum());
27     Console.WriteLine("Sorted");
28     Array.Sort(arr);
29     foreach (int i in arr)
30     {
31         Console.WriteLine(i);
32     }
33 }
34 }
35 }
```

The code demonstrates various operations on an integer array, including printing its elements, calculating average and sum, sorting it, and printing it again.

```
1
5
2
9
6
3
8
4
10
7
Average : 5.5
Sum : 55
Sorted
on1
:d2
13
ip4
lu5
6
7
8
9
10
Press any key to continue . .
```

how to dive little deep in it and learn

A screenshot of the Microsoft Visual Studio IDE. The window title is "ConsoleApp". The code editor displays the following C# code:

```
11 static void Main()
12 {
13     //int[] arr = new int[5];
14     int[] arr = { 1, 5, 2, 9, 6, 3, 8, 4, 10, 7 };
15     /*
16     for (int i = 0; i < arr.Length; i++)
17     {
18         Console.WriteLine(arr[i]);
19     }
19 */
20 foreach (int i in arr)
21 {
22     Console.WriteLine(i);
23 }
24 Console.WriteLine("Average : " + arr.Average());
25 Console.WriteLine("Sum : " + arr.Sum());
26 Console.WriteLine("Sorted");
27 Array.Sort(arr);
28 foreach (int i in arr)
29 {
30     Console.WriteLine(i);
31 }
32 }
```

The cursor is positioned over the variable "arr" in the line "Console.WriteLine("Sum : " + arr.Sum());". A tooltip appears with the text "(local variable) int[] arr". A red arrow points from the text "hold your mouse pointer on this it will take you here" to the "arr" variable in the tooltip.

hold your mouse pointer
on this it will take you here

A screenshot of the Microsoft Visual Studio IDE interface. The main window shows a C# file named 'ArrayDemo1.cs' with the following code:

```
4  using System.Text;
5  using System.Threading.Tasks;
6
7  namespace ConsoleApp
8  {
9      class ArrayDemo1
10     {
11         static void Main()
12         {
13             //int[] arr = new int[5];
14             int[] arr = { 1, 5, 2, 9, 6, 3, 8, 4, 10, 7 };
15             /*
16             for (int i = 0; i < arr.Length; i++)
17             {
18                 Console.WriteLine(arr[i]);
19             }
20             */
21             foreach (int i in arr)
22             {
23                 Console.WriteLine(i);
24             }
25             Console.WriteLine("Average : " + arr.Average());
26         }
27     }
28 }
```

The line `int[] arr = { 1, 5, 2, 9, 6, 3, 8, 4, 10, 7 };` is highlighted with a blue selection bar. A large red arrow points from the text "to your array" towards this line. The status bar at the bottom left indicates "121 %".

to your array

The screenshot shows the Microsoft Visual Studio IDE interface. The top menu bar includes File, Edit, View, Git, Project, Build, Debug, Test, Analyze, Tools, Extensions, Window, and Help. Below the menu is a toolbar with various icons. The title bar displays "ArrayDemo1.cs" and "ConsoleApp". The main window shows the code for "ConsoleApp.ArrayDemo1". A red arrow points from the text "in the same way see you have a Array here when you click here it will take you to array liabrary" to the word "Array" in the code. A red box highlights the word "Array". The code is as follows:

```
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
```

```
/*
for (int i = 0; i < arr.Length; i++)
{
    Console.WriteLine(arr[i]);
}
*/
foreach (int i in arr)
{
    Console.WriteLine(i);
}
Console.WriteLine("Average : " + arr.Average());
Console.WriteLine("Sum : " + arr.Sum());
Console.WriteLine("Sorted");
Array.Sort(arr);
foreach (int i in arr)
{
    Console.WriteLine(i);
}
```

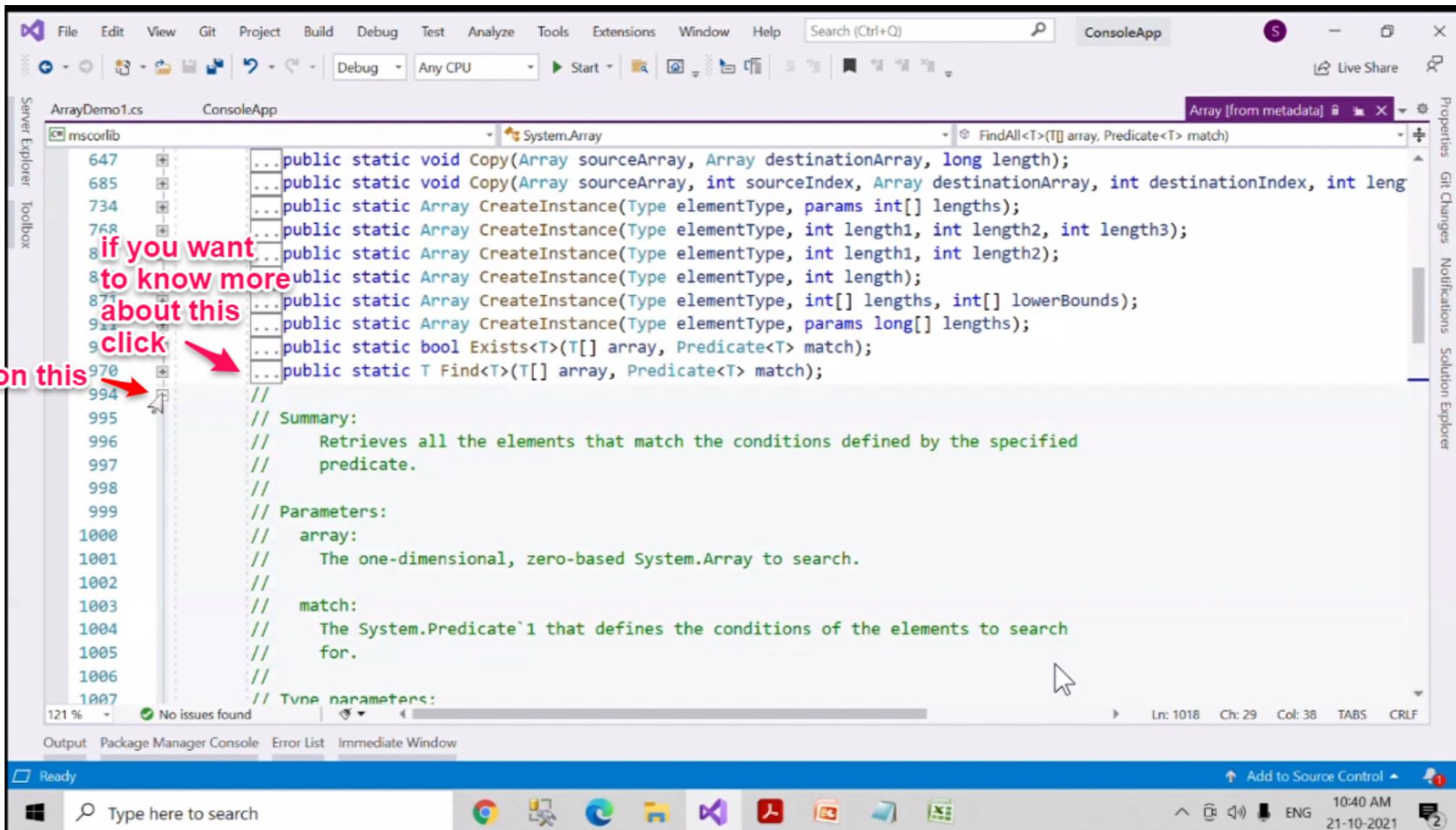
in the same way
see you have a
Array here
when you click
here it will take you
to array liabrary

```
Assembly mscorlib, Version=4.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089
using ...
namespace System
{
    public abstract class Array : ICloneable, IList, ICollection, IEnumerable, IStructuralComparable, IStructuralEquatable
    {
        public bool IsSynchronized { get; }
        public bool IsReadOnly { get; }
        public object SyncRoot { get; }
        public int Rank { get; }
        public long LongLength { get; }
        public int Length { get; }
        public bool IsFixedSize { get; }

        public static ReadOnlyCollection<T> AsReadOnly<T>(T[] array);
        public static int BinarySearch(Array array, object value);
        public static int BinarySearch(Array array, object value, IComparer comparer);
        public static int BinarySearch(Array array, int index, int length, object value, IComparer comparer);
        public static int BinarySearch<T>(T[] array, T value);
        public static int BinarySearch<T>(T[] array, T value, IComparer<T> comparer);
        public static int BinarySearch<T>(T[] array, int index, int length, T value);
        public static int BinarySearch<T>(T[] array, int index, int length, T value, IComparer<T> comparer);
    }
}
```

This is my meta data you can see that array has sooo many methods in it
,,,no need to write any search algorithm by you self you already have a
binary search So you can always use it ,

you have a **find method**, **find all method**.....many more just check exactly what it does



The screenshot shows the Microsoft Visual Studio IDE interface. The main window displays the code for the `Find<T>` method in the `System.Array` class. The code is as follows:

```
public static T Find<T>(T[] array, Predicate<T> match);
```

Below the code, there is a summary and parameter information:

// Summary:
// Retrieves all the elements that match the conditions defined by the specified
// predicate.

// Parameters:
// array:
// The one-dimensional, zero-based System.Array to search.
//
// match:
// The System.Predicate`1 that defines the conditions of the elements to search
// for.

A red callout box with the text "if you want to know more about this click on this" points to the line number 970, which contains the opening brace of the `Find<T>` method definition. A red arrow also points from the text to the line number 970.

it will give you the complete help like what it does now while reading line no 997 you dont know about **predicate**

The screenshot shows the Microsoft Visual Studio IDE interface. The main window displays the code for `ArrayDemo1.cs` under the project `ConsoleApp`. The code is part of the `mscorlib` namespace and shows the implementation of the `System.Array` class. A tooltip is displayed over the `Predicate<T>` parameter in the `Find<T>` method, which is highlighted with a red arrow. The tooltip provides the following information:

- Delegate bool System.Predicate<in T>(T obj)**
- Represents the method that defines a set of criteria and determines whether the specified object meets those criteria.**
- Returns:** true if obj meets the criteria defined within the method represented by this delegate; otherwise, false.

The code snippet for the `Find<T>` method is as follows:

```
public static T Find<T>(T[] array, Predicate<T> match);
```

The cursor is positioned at the end of the line number 994, and the status bar at the bottom right shows the coordinates `Ln: 993 Ch: 43 Col: 52 TABS CRLF`.

File Edit View Git Project Build Debug Test Analyze Tools Extensions Window Help Search (Ctrl+Q) Co

ArrayDemo1.cs ConsoleApp

mscorlib

System.Predicate<in T>

```
1  Assembly mscorlib, Version=4.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089
4
5  namespace System
6  {
7  public delegate bool Predicate<in T>(T obj);
25 }
```

The screenshot shows the Microsoft Visual Studio IDE interface. The title bar displays "ConsoleApp". The menu bar includes File, Edit, View, Git, Project, Build, Debug, Test, Analyze, Tools, Extensions, Window, Help, and a search bar. The toolbar contains various icons for file operations like Open, Save, and Print.

The main code editor window shows the file "ConsoleApp.cs" with the following code:

```
Assembly mscorlib, Version=4.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089
namespace System
{
    // Summary:
    //     Represents the method that defines a set of criteria and determines whether the
    //     specified object meets those criteria.
    // Parameters:
    //     obj:
    //         The object to compare against the criteria defined within the method represented
    //         by this delegate.
    // Type parameters:
    //     T:
    //         The type of the object to compare.
    // Returns:
    //     true if obj meets the criteria defined within the method represented by this
    //     delegate; otherwise, false.
    public delegate bool Predicate<in T>(T obj);
}
```

The code editor highlights the word "Predicate" in blue. The status bar at the bottom shows "121 %", "No issues found", "Ln: 24", and tabs for Output, Package Manager Console, Error List, and Immediate Window. The taskbar at the bottom includes icons for Start, Search, Google Chrome, File Explorer, Task View, Task Manager, File History, and Task Scheduler.

So no need to go to google for searching all this things
but if you know more about this always goto Google Aunty pick the
keyword and search
like **C# predicate**

RE: 202 .NET Batch | Mphasis - Webex - | RE: 202 .NET Batch | WhatsApp | c# predicate - Google | Loading... | +

← → C google.com/search?q=c%23+predicate&rlz=1C1CHBF_enIN947IN947&oq=C%23+Predicate&aqs=chrome.0.0i512l7j69i58.2407j0j7&sourceid=chrome&i

Google

c# predicate

All Videos Images News Shopping More Tools

About 4,86,000 results (0.55 seconds)

<https://docs.microsoft.com> › ... › System

Predicate<T> Delegate (System) | Microsoft Docs

Note that it is customary to use a lambda expression rather than to explicitly define a delegate of type `Predicate<T>`, as the second example illustrates. C#

[Definition](#) · [Examples](#) · [Remarks](#)

<https://www.tutorialsteacher.com> › csharp-predicate

Predicate delegate in C# - TutorialsTeacher

The `Predicate` is the delegate type like `Func` and `Action` in C#. It represents a method that contains a set of criteria and checks whether the passed ...

People also ask

What is a predicate in C#?

What is predicate int?

<https://docs.microsoft.com/en-us/dotnet/api/system.predicate-1>

Type here to search

Chrome File Explorer Edge VS Code PDF Word Excel Powerpoint Powerpoint

please refer to microsoft document before going to any other documents
and whenever you opening this press ctrl key then enter so that it opens in a new tab



Microsoft Ignite
November 2–4, 2021 | Free digital event

Please Read it ,IT is a beautiful way of learning

Join us November 2–4, 2021 for our digital experience, including the latest product demos, Q&A with Microsoft experts, technical deep-dives, and more.

Register now >

Microsoft | Docs Documentation Learn Q&A Examples

Search Sign in

.NET Languages Workloads APIs Resources

Download .NET

Docs / .NET / .NET API browser / System / Predicate<T>

C# Save Edit Share

Version .NET 5

Search

Predicate<T>

- > Progress<T>
- > Random
- > Range
- > RankException

Is this page helpful? Yes No

In this article

- Definition
- Examples
- Remarks
- Extension Methods
- Applies to

Connecting... Copy

Type here to search

10:41 AM 21-10-2021

Predicate<T> Delegate

Definition

Namespace: [System](#)
Assembly: [System.Runtime.dll](#)

Represents the method that defines a set of criteria and determines whether the specified object meets those criteria.

arrayname.Rank-----Rank will always say what is the dimension of array you are using weather you are going with

multidimensional whether you are going with single dimension

```
'  
int[,] arr = new int[3,3];  
Console.WriteLine("Rank of array :" + arr.Rank);
```

```
Rank of array :2  
Press any key to continue . . .
```

Array are defined or Derived from the abstract class called type Array , this particular arrays are implementing called IEnumearable

When you come here we can see this are IEnumearable they are inheriting from IEnumearable

Assembly mscorlib, Version=4.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089

```
1  using ...
4
5  namespace System
11
12  {
13      public abstract class Array : ICloneable, IList, ICollection<T>, IEnumerable, IStructuralComparable, IStructuralEquatable
14  {
15      public bool IsSynchronized { get; }
16      public bool IsReadOnly { get; }
17      public object SyncRoot { get; }
18      public int Rank { get; }
19      public long LongLength { get; }
20      public int Length { get; }
21      public bool IsFixedSize { get; }
22
23      public static ReadOnlyCollection<T> AsReadOnly<T>(T[] array);
24      public static int BinarySearch(Array array, object value);
25      public static int BinarySearch(Array array, object value, IComparer comparer);
26      public static int BinarySearch(Array array, int index, int length, object value, IComparer comparer);
27      public static int BinarySearch<T>(T[] array, T value);
28      public static int BinarySearch<T>(T[] array, T value, IComparer<T> comparer);
29      public static int BinarySearch<T>(T[] array, int index, int length, T value);
30      public static int BinarySearch<T>(T[] array, int index, int length, T value, IComparer<T> comparer);
31
32  }
```

No issues found

Output Package Manager Console Error List Immediate Window

Ready Add to Source Control 1

Type here to search

121 % Ln: 19 Ch: 29 Col: 35 TABS CRLF

File Edit View Git Project Build Debug Test Analyze Tools Extensions Window Help Search (Ctrl+Q) ConsoleApp S - X Live Share Properties Git Changes Notifications Solution Explorer

File Edit View Git Project Build Debug Test Analyze Tools Extensions Window Help Search (Ctrl+Q)

Debug Any CPU Start

ArrayDemo1.cs ConsoleApp

mscorlib System.Collections.IEnumerable

```
1  Assembly mscorlib, Version=4.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089
4
5      using System.Runtime.InteropServices;
6
7  namespace System.Collections
8  {
9      public interface IEnumerable
10     {
11         IEnum...
12         ...
13         IEn...
14         ...
15         IEn...
16         ...
17         IEn...
18         ...
19         IEn...
20         ...
21         IEn...
22         ...
23         IEn...
24         ...
25     }
26 }
```

so IEnumerable has
a GetEnumerator

click here

File Edit View Git Project Build Debug Test Analyze Tools Extensions Window Help Search (Ctrl+Q) ConsoleApp

ArrayDemo1.cs ConsoleApp

mscorlib System.Collections.IEnumerator Current

```
1 Assembly mscorlib, Version=4.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089
4
5 using System.Runtime.InteropServices;
6
7 namespace System.Collections
8 {
9     public interface IEnumerator
10    {
11         object Current { get; }
12
13         bool MoveNext();
14
15         void Reset();
16     }
17 }
```

so this getEnumerator has a current move reset

Methods so this methods will help you to traverse through the collection