Duhok Polytechnic University
Zakho Technical Institute
Department of Information Technology

Object Oriented Programming

2 - loops and Array

Sipan M. HAmeed

www.sipan.dev

2024-2025

Contents

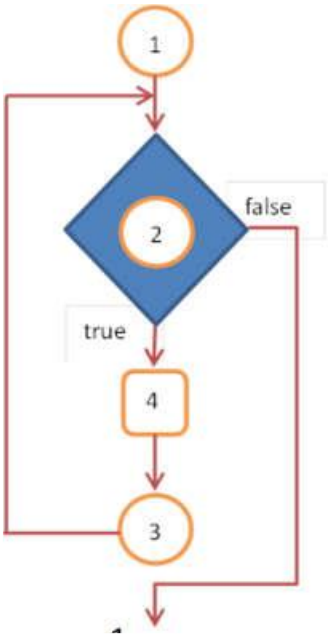
Iteration Statements:for, while, do while, and foreach

Iteration statements are used to run a set or a block of code lines (statements) repeatedly while some condition is true.

1 - For Loop:

For statement consists of 3 parts besides the block of statements that executed each iteration.

```
        ١      ٢      ٣
for (          ;          ;          )
{
        4 :  statements that executed in each
        iteration
}
```



The code after the block

١ **initialization**: This part is executed once before the loop starts and is typically used to initialize a loop control variable.

٢ **condition**: This is a Boolean expression that is evaluated before each iteration. If it's true, the loop continues; if it's false, the loop terminates.

٣ **iteration**: This part is executed after each iteration and is typically used to update the loop control variable.

Example: write a program to display "**Hello world**" **10** times on Console using for loop.

```
static void Main()
{
    for (int i=0;i<10;i++)
    {
        Console.WriteLine("Hello world");
    }
    Console.ReadKey();
}
```

```
C:\Users\96475\so
Hello world
Hello world
Hello world
Hello world
Hello world
Hello world
Hello world
Hello world
Hello world
Hello world
```

In the first part: variable (i) has been declared and initialized: int i = 0

"i" is used as a counter to know how many times to execute the: {

Console.WriteLine("Hello world");}

In the second part: the condition (i < 10) inside the (for) checks whether (i) is still less than 10 or not.

In the third part (i++): the counter "i" will increase by one every time after completing the execution of the code inside for-loop block:

Example: Write a program to display the even numbers between 1 and 101

```
for (int i = 1; i < 101; i++)
{
    if (i % 2 == 0)
        Console.WriteLine(i);
}
Console.ReadLine();
```

```
for (int i = 2; i < 101; i+=2)
{
        Console.WriteLine(i);
}
Console.ReadLine();
```

Using **for loop** inside **for loop**

```
for (int i = 0; i < 5; i++)
{
    for (int j = 0; j <= i; j++)
    {
        Console.Write("*");
    }
    Console.WriteLine();
}

Console.ReadLine();
```

```
file:///
*
**
***
****
*****
```

2 - while and do while

while (**Boolean expression**)
{
block of statements
}
	Do
	{
	block of statements
	} while (**Boolean expression**) ;

Boolean expression: represents the condition of the **while** and **do while statements,** the value of this **expression** determines whether the **block of statements** will be executed or not.

If the **Boolean expression** is **true** the block of statements will be executed
If the **Boolean expression** is **false** the block of statements will not be executed (end of **while** / **do while**)

The difference between **while loop** and **do-while loop :**
In **while loop** the condition **(Boolean expression)** is examined at the beginning of each iteration, while in **do while loop the condition (Boolean expression) is** examined at the end of each iteration.

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

int i = 0;
 while (i < 0)
{
    Console.WriteLine("work");
    i++;
}

Console.WriteLine("End");

Console.ReadLine();
```

```
while
End
```

```
Console.WriteLine("do while");

int i = 0;
do
{
    Console.WriteLine("work");
    i++;
} while (i < 0);

Console.WriteLine("End");

Console.ReadLine();
```

```
do while
work
End
```

---

1. Foreach

foreach loop is used for iterating over elements in a **collection**, such as an **array**, **list**, or any other enumerable object. The foreach loop provides a simple and convenient way to loop through each item in the collection without having to deal with manual indexing.

```
foreach (ElementType element in collection)
{
    // Code to execute for each element
}
```

- **ElementType**: This is the data type of each element in the collection. It should match the data type of the elements in the collection you are iterating over.

- **element**: This is a variable name that you create to represent each element in the collection during each iteration. You can choose any valid variable name.
- **collection**: This is the collection you want to iterate over. It can be an array, list, etc.

3 - Arrays

An array is a group of like-typed variables that are referred to by a common name. Each data item is called an element of the array. The data types of the elements may be any valid data type like char, int, float, etc. and the elements are stored in a contiguous location. The length of the array specifies the maximum number of elements that can be stored in the array. The elements in the array are ordered and each has an index beginning from 0.

Arrays are used to store multiple values in a single variable, instead of declaring separate variables for each value.

Syntax:

```
< datatype > [ ] < Array_Name > = new < datatype > [size];
```

1. **Declaring an Array:** To declare an array, you need to specify the data type of its elements and provide a name for the array. You can declare an array in various ways:

```
// Declare an integer array with 5 elements
int[] scores = new int[5];      // scores = {0,0,0,0,0}

// Declare and initialize an array
```

- `int[] scores = new int[5]{ 95, 87, 72, 88, 91 };`
- `int[] scores = { 95, 87, 72, 88, 91 };`

Array elements ➔	**95**	**87**	**72**	**88**	**91**
Indexes ➔	**0**	**1**	**2**	**3**	**4**

2. **Accessing Array Elements:** Array elements are accessed using zero-based indexing. For example, to access the first element of an array, you use index 0.

```
int firstNumber = scores[0]; // Accessing the first element=95
```

3. **Array Length:** You can get the length (the number of elements) of an array using the Length property.

```
int length = scores.Length; // Get the length of the array =5
```

4. **Modifying Array Elements:** You can change the value of an array element by assigning a new value to it.

```
scores[2] = 42; // Change the value of the third element
```

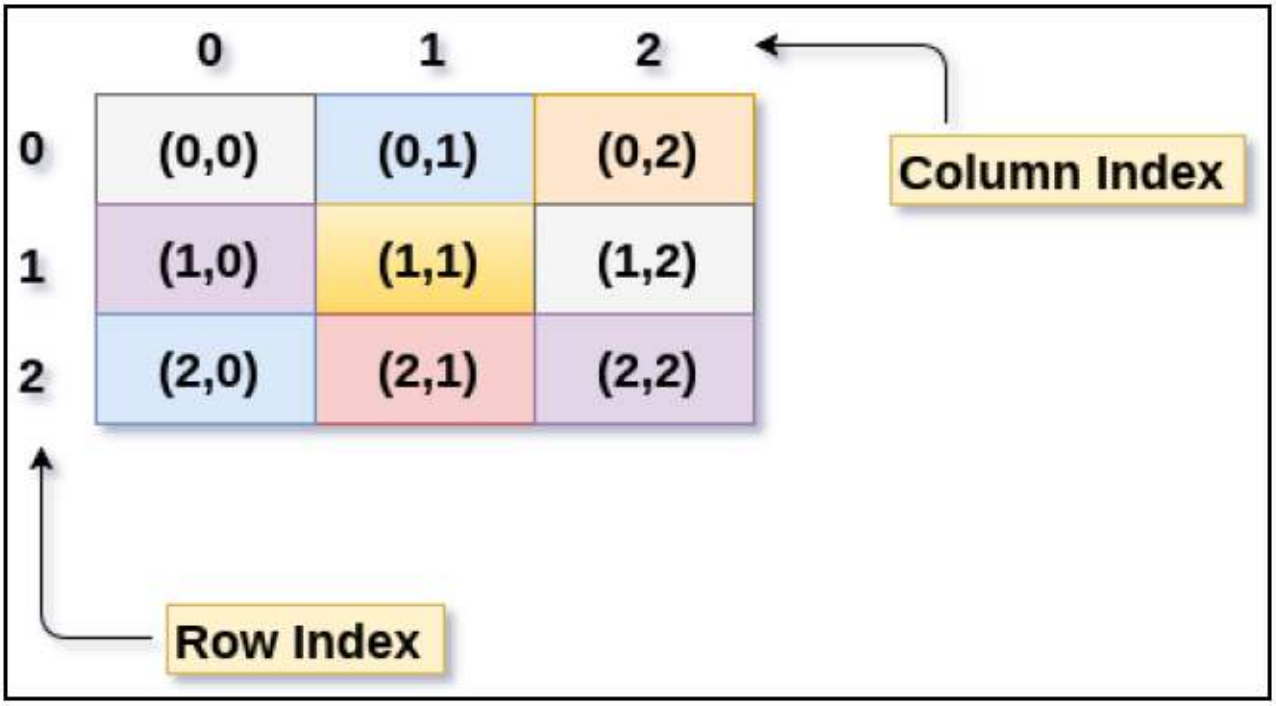
5. **Accessing array elements using loops:** You can use loops, such as **for** or **foreach**, to iterate through the elements of an array.

```
foreach (int score in scores)
{
    Console.WriteLine(score);
}
```

```
95
87
72
88
91
```

6. **Multidimensional Arrays:** C# supports multidimensional arrays, including two-dimensional arrays, which are often used to represent tables or grids of data.

```
int[,] matrix = new int[3, 3]; // 2D array with 3 rows and 3 columns
```



Practical Examples:

Example-1:

write a program to store 6 integers in an array and then print them.

```
static void Main()
{
        int[] numbers = new int[6];
        Console.WriteLine("please enter 6 numbers");

        for (int i = 0; i < 6; i++)
        {
           numbers[i] = Convert.ToInt32(Console.ReadLine());
        }
        Console.Write("printing array elements using for loop: ");
        for (int i = 0; i < 6; i++)
        {
           Console.Write(numbers[i] + ", ");
        }
        Console.WriteLine();
        Console.Write("printing array elements using foreach: ");
        foreach (var item in numbers)
        {
           Console.Write(item+", ");
        }
        Console.ReadKey();
}
```

Output:

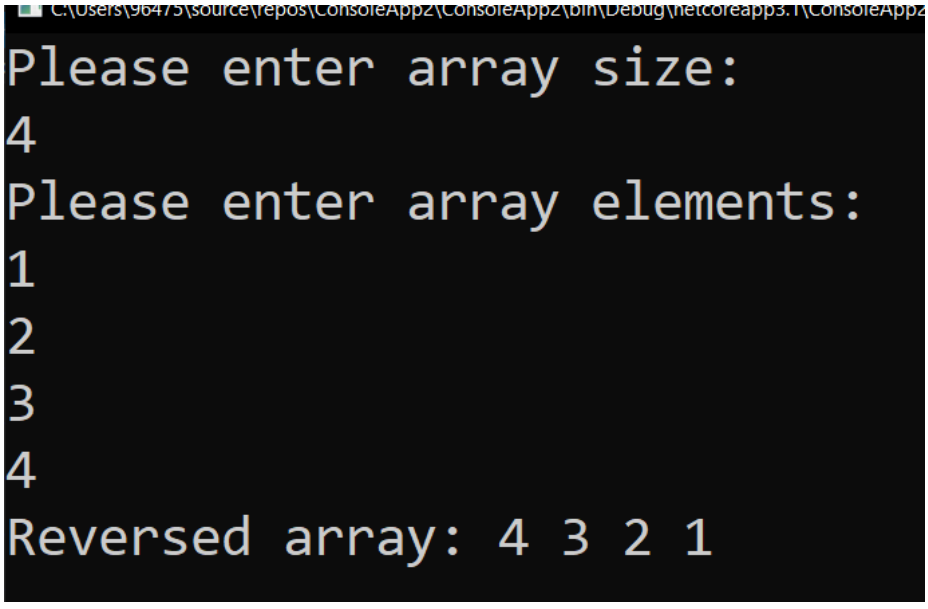
```
please enter 6 numbers
10
20
30
40
50
60
printing array elements using for loop: 10, 20, 30, 40, 50, 60,
printing array elements using foreach: 10, 20, 30, 40, 50, 60,
```

Example-2:

Write a program to store n number of integers in an array and display them in reverse order.

```
static void Main()
{
        Console.WriteLine("Please enter array size: ");
        int n = Convert.ToInt32(Console.ReadLine());
        int[] numbers = new int[n];
        Console.WriteLine("Please enter array elements:");
        for (int i = 0; i < n; i++)
        {
            numbers[i] = Convert.ToInt32(Console.ReadLine());
        }
        Console.WriteLine("Reversed array: ");
        for (int i = n - 1; i >= 0; i--)
        {
            Console.Write(numbers[i] + " ");
        }
        Console.ReadKey();
}
```

Output:



Example-3:

Write a program to read 7 marks, store them in an array (named "Marks"), and calculate their summation.

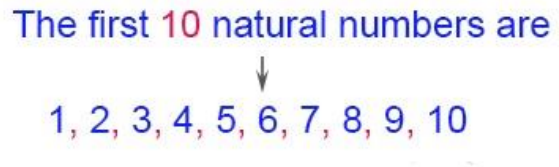
```
static void Main()
{
   Console.WriteLine("Please enter student marks: ");
   int[ ] Marks = new int[7];
   int sum=0;
   for (int i = 0; i < Marks.Length; i++)
   {
      Marks[i] = Convert.ToInt32(Console.ReadLine());
      sum += Marks[i];
   }
   Console.Write("sum= "+sum);
}
```

Output:

```
Please enter student marks:
100
50
75
75
50
50
100
sum= 500
```

Example-4

Write a program in C# Sharp to display the first 10 natural numbers.



Code:

```
using System;
public class Exercise1
{
    public static void Main()
    {
    int i;
    Console.Write("\n\n");
    Console.Write("Display the first 10 natural numbers:\n");
    Console.Write("--------------------------------------");
    Console.Write("\n\n");

     Console.WriteLine("The first 10 natural number are:");

      for (i=1;i<=10;i++)
      {
     Console.Write("{0} ",i);
      }
    Console.Write("\n\n");
  }
}
```

Output:

```
Display the first 10 natural numbers:
--------------------------------------
The first 10 natural number are:
2 3 4 5 6 7 8 9 10
```

Example 5.

Write a C# Sharp program to find the sum of first 10 natural numbers.

C# Sharp Exercises: Display the sum of first 10 natural numbers

Solution:Code:

```
using System;
public class Exercise2
{
    public static void Main()
{
    int  j, sum = 0;

     Console.Write("\n\n");
    Console.Write("Find the sum of first 10 natural numbers:\n");
    Console.Write("-----------------------------------------------");
    Console.Write("\n\n");

    Console.Write("The first 10 natural number are :\n");
     for (j = 1; j <= 10; j++)
    {
        sum = sum + j;
        Console.Write("{0} ",j);
    }
    Console.Write("\nThe Sum is : {0}\n", sum);
  }
}
```

Output:

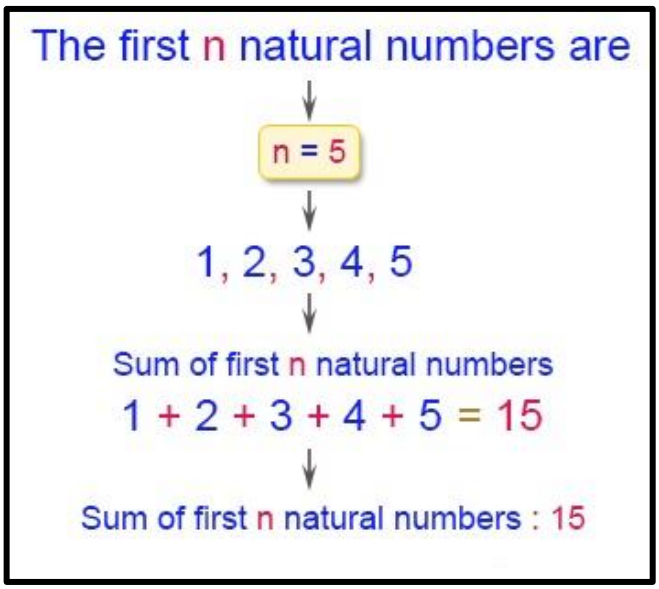
```
Find the sum of first 10 natural numbers:
-----------------------------------------------

The first 10 natural number are :
1 2 3 4 5 6 7 8 9 10
The Sum is : 55
```

Exercise-6 with Solution

Write a program in C# Sharp to display n terms of natural number and their sum.

C# Sharp Exercises: Display n natural numbers and their sum

Solution:



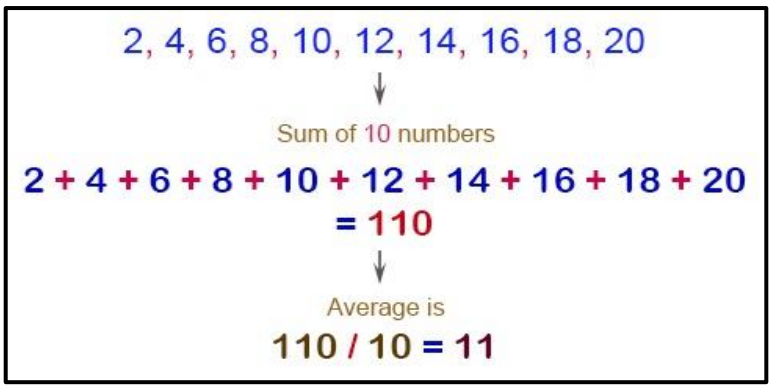

```
using System;
public class Exercise3
{
    public static void Main()
{
   int i,n,sum=0;

      Console.Write("\n\n");
    Console.Write("Display n terms of natural number and their sum:\n");
    Console.Write("-----------------------------------------");
    Console.Write("\n\n");

   Console.Write("Input Value of terms : ");
   n= Convert.ToInt32(Console.ReadLine());
   Console.Write("\nThe first {0} natural number are :\n",n);
   for(i=1;i<=n;i++)
   {
     Console.Write("{0} ",i);
     sum+=i;
   }
   Console.Write("\nThe Sum of Natural Number upto {0} terms : {1} \n",n,sum);
  }
}
```

Output:

```
Display n terms of natural number and their sum:
-----------------------------------------------

Input Value of terms : 5

The first 5 natural number are :
1 2 3 4 5
The Sum of Natural Number upto 5 terms : 15
```

Exercise-7 with Solution(array)

Write a program in C# Sharp to read 10 numbers from keyboard ,store it in array and find their sum and average and print the element of array. Then print the array.



Solution:

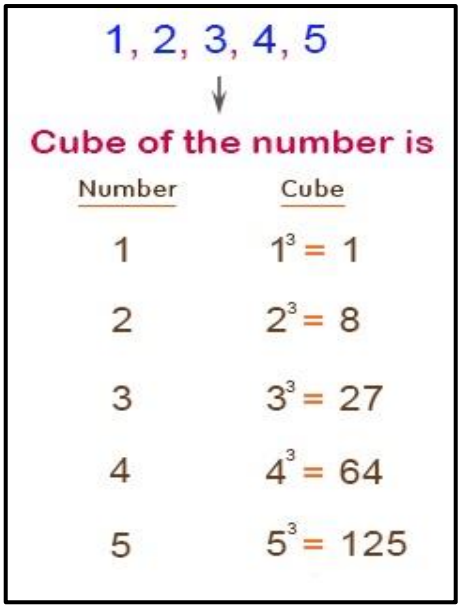
```
using System;
public class Exercise4
{
    public static void Main()
    {
        int i, n, sum = 0;
        double avg;
        int[] ar = new int[10];

        Console.Write("\n\n");
        Console.Write("Read 10 numbers and calculate sum and average:\n");
        Console.Write("-----------------------------------------------");
        Console.Write("\n\n");

        Console.Write("Input the 10 numbers : \n");
        for (i = 0; i < 10; i++)
        {
            Console.Write("Number-{0} :", i);

            n = Convert.ToInt32(Console.ReadLine());
            sum += n;

            ar[i]=n;
        }
        avg = sum / 10.0;
        Console.Write("The sum of 10 no is : {0}\nThe Average is : {1}\n",
sum, avg);

        for (i = 0; i < 10; i++)
        {
            Console.WriteLine(ar[i]);
        }
    }
}
```

Output:

```
Read 10 numbers and calculate sum and average:
-----------------------------------------------

Input the 10 numbers :
Number-0 :66
Number-1 :32
Number-2 :4
Number-3 :8
Number-4 :6
Number-5 :7
Number-6 :9
Number-7 :2
Number-8 :11
Number-9 :32
The sum of 10 no is : 177
The Average is : 17.7
66
32
4
8
6
7
9
2
11
32
Press any key to continue . . .
```

Exercise-8 with Solution(array)

Write a program in C# Sharp to display the cube of the number upto given an integer. And store the numbers and their cube in two array



Solution:

```
using System;
public class Exercise5
{
    public static void Main()
    {
        int i, ctr;
        Console.Write("\n\n");
        Console.Write("Display the cube of the
number:\n");
        Console.Write("------------------------
---------");
        Console.Write("\n\n");

        Console.Write("Input number of terms : ");
        ctr = Convert.ToInt32(Console.ReadLine());
        int[] ar1 = new int[ctr];
        int[] ar2 = new int[ctr];

        for (i = 0; i < ctr; i++)
        {
            Console.Write("Number is : {0} and cube of the {1} is :{2} \n",
i, i, (i * i * i));
            ar1[i] = i;
            ar2[i] = i*i*i;
        }

        for (i = 0; i < ctr;i++)
        {
            Console.WriteLine( "  {0} the cube is  {1}", ar1[i], ar2[i] );
        }
    }
}
```

Output:

```
Display the cube of the number:
--------------------------------

Input number of terms : 5
Number is : 0 and cube of the 0 is :0
Number is : 1 and cube of the 1 is :1
Number is : 2 and cube of the 2 is :8
Number is : 3 and cube of the 3 is :27
Number is : 4 and cube of the 4 is :64
  0 the cube is  0
  1 the cube is  1
  2 the cube is  8
  3 the cube is  27
  4 the cube is  64
Press any key to continue . . .
```

Exercise-9 with Solution

Write a program in C# Sharp to display the pattern like right angle triangle using an asterisk.

Sample Output:

```
*
**
***
****
```

Solution:-

```
using System;
public class Exercise9
{
    public static void Main()
{
    int i,j,rows;
      Console.Write("\n\n");
    Console.Write("Display the pattern like right angle using asterisk:\n");
    Console.Write("----------------------------------------");
    Console.Write("\n\n");

   Console.Write("Input number of rows : ");
   rows= Convert.ToInt32(Console.ReadLine());
   for(i=1;i<=rows;i++)
   {
      for(j=1;j<=i;j++)
         Console.Write("*");
      Console.Write("\n");
   }
  }
}
```

Output:

```
Display the pattern like right angle using asterisk:
-----------------------------------------------------


Input number of rows : 10
*
**
***
****
*****
******
*******
********
*********
**********
```

Exercise-10 with Solution

Write a program in C# Sharp to make such a pattern like a pyramid with numbers increased by 1.
The pattern is as follows:

```
   1
  2 3
 4 5 6
7 8 9 10
```

Solution: -

```
using System;
public class Exercise13
{
    public static void Main()
{
   int i,j,spc,rows,k,t=1;

      Console.Write("\n\n");
    Console.Write("Display the pattern like pyramid with numbers increased by
1:\n");
    Console.Write("-----------------------------------------");
    Console.Write("\n\n");

   Console.Write("input number of rows : ");
   rows= Convert.ToInt32(Console.ReadLine());
   spc=rows+4-1;
   for(i=1;i<=rows;i++)
   {
         for(k=spc;k>=1;k--)
            {
              Console.Write(" ");
            }
         for(j=1;j<=i;j++)
         Console.Write("{0} ",t++);
      Console.Write("\n");
    spc--;
   }
}
}
```

Output:

```
Display the pattern like pyramid with numbers increased by 1
------------------------------------------------------------
input number of rows : 5
        1
       2 3
      4 5 6
     7 8 9 10
    11 12 13 14 15
```

Exercise-11with Solution (2-D array)

Write a program in C# to make such a pattern like a shown in below and store it in 2-D array,

Note: yellow is the first diagonal, and green is the second diagonal.

Find sum of first diagonal and second diagonal.

The pattern is as follows:

1	2	3	4	5
6	7	8	9	10
11	12	13	14	15
16	17	18	19	20
21	22	23	24	25

Solution: -

```
using System;
using System.Threading;
public class Exercise5
{
    public static void Main()
    {
        int[,] ar1 = new int[5,5];
        int count = 1;

        int sum1 = 0, sum2 = 0;

        for (int i = 0; i < 5; i++)
        {
            for (int j = 0; j < 5; j++)
            {
                ar1[i, j] = count;
                count++;
            }
        }

        for (int i = 0;i < 5; i++)
        {
            sum1 = sum1 + ar1[i,i];
            sum2 = sum2 + ar1[i,4 - i];
        }
        Console.WriteLine( "sum 1 = {0}   sum2  = {1}",sum1,sum2);


    }
}
```

Output:-

```
sum 1 = 65   sum2  = 65
Press any key to continue . . .
```