TutorialKart

C# Math.Min() – Syntax & Examples

C# Math.Min() – Examples

In this tutorial, we will learn about the C# Math.Min() method, and learn how to use this method to find minimum of two numbers/values, with the help of examples.

Min(Byte, Byte)

Math.Min(val1, val2) returns the smaller of two 8-bit unsigned integers: `val1` and `val2`.

Syntax

The syntax of Min() method is

```
Math.Min(Byte val1, Byte val2)
```

where

Parameter	Description
val1	The first of two 8-bit unsigned integers to compare.
val2	The second of two 8-bit unsigned integers to compare.

Return Value

The method returns val1 or val2, whichever is minimum.

Example 1 – Min(Byte val1, Byte val2)

In this example, we will find the smallest of two 8-bit unsigned integers using Min() method.

C# Program

```
using System;
```

```
using System;

class Example {
    static void Main(string[] args) {
        byte val1 = 7;
        byte val2 = 25;

        byte result = Math.Min(val1, val2);
        Console.WriteLine($"Minimum of {val1} and {val2} is {result}.");
    }
}
```

Output

```
Minimum of 7 and 25 is 7.
```

Min(Decimal, Decimal)

Math.Min(val1, val2) returns the smaller of two decimal numbers: `val1` and `val2`.

Syntax

The syntax of Min() method is

```
Math.Min(Decimal val1, Decimal val2)
```

where

Parameter	Description
Decimal	The first of two decimal numbers to compare.
Decimal	The second of two decimal numbers to compare.

Return Value

The method returns val1 or val2, whichever is minimum.

Example 2 – Min(Decimal val1, Decimal val2)

In this example, we will find the smallest of two decimal numbers using Min() method.

C# Program

```
using System;

class Example {
    static void Main(string[] args) {
        Decimal val1 = 7.5M;
        Decimal val2 = 7.1M;

        Decimal result = Math.Min(val1, val2);
        Console.WriteLine($"Minimum of {val1} and {val2} is {result}.");
    }
}
```

Output

```
Minimum of 7.5 and 7.1 is 7.1.
```

Min(Double, Double)

Math.Min(val1, val2) returns the smaller of two double-precision floating-point numbers: `val1` and `val2`.

Syntax

The syntax of Min() method is

```
Math.Min(Double val1, Double val2)
```

where

Parameter	Description
val1	The first of two double values to compare.
val2	The second of two double values to compare.

Return Value

The method returns value.

Example 3 – Min(Double val1, Double val2)

In this example, we will find the smallest of two double values using Min() method.

C# Program

```
using System;

class Example {
    static void Main(string[] args) {
        Double val1 = 7.5785;
        Double val2 = 7.18974;

        Double result = Math.Min(val1, val2);
        Console.WriteLine($"Minimum of {val1} and {val2} is {result}.");
    }
}
```

Output

```
Minimum of 7.5785 and 7.18974 is 7.18974.
```

Min(Int16, Int16)

Math.Min(val1, val2) returns the smaller of two 16-bit signed integers: `val1` and `val2`.

Syntax

The syntax of Min() method is

```
Math.Min(Int16 val1, Int16 val2)
```

where

Parameter	Description
val1	The first of two 16-bit signed integers to compare.
val2	The second of two 16-bit signed integers to compare.

Return Value

The method returns val1 or val2, whichever is minimum.

Example 4 – Min(Int16 val1, Int16 val2)

In this example, we will find the smallest of two 16-bit signed integers using Min() method.

C# Program

```
using System;

class Example {
    static void Main(string[] args) {
        Int16 val1 = 7;
        Int16 val2 = 24;

        Int16 result = Math.Min(val1, val2);
        Console.WriteLine($"Minimum of {val1} and {val2} is {result}.");
    }
}
```

Output

```
Minimum of 7 and 24 is 7.
```

Min(Int32, Int32)

Math.Min(val1, val2) returns the smaller of two 32-bit signed integers: `val1` and `val2`.

Syntax

The syntax of Min() method is

```
Math.Min(Int32 val1, Int32 val2)
```

where

Parameter	Description
val1	The first of two 32-bit signed integers to compare.
val2	The second of two 32-bit signed integers to compare.

Return Value

The method returns val1 or val2, whichever is minimum.

Example 5 – Min(Int32 val1, Int32 val2)

In this example, we will find the smallest of two 32-bit signed integers using Min() method.

C# Program

```
using System;

class Example {
    static void Main(string[] args) {
        Int32 val1 = 7;
        Int32 val2 = 24;

        Int32 result = Math.Min(val1, val2);
        Console.WriteLine($"Minimum of {val1} and {val2} is {result}.");
    }
}
```

Output

```
Minimum of 7 and 24 is 7.
```

Min(Int64, Int64)

Math.Min(val1, val2) returns the smaller of two 64-bit signed integers: `val1` and `val2`.

Syntax

The syntax of Min() method is

```
Math.Min(Int64 val1, Int64 val2)
```

where

Parameter	Description
val1	The first of two 64-bit signed integers to compare.
val2	The second of two 64-bit signed integers to compare.

Return Value

The method returns val1 or val2, whichever is minimum.

Example 6 – Min(Int64 val1, Int64 val2)

In this example, we will find the smallest of two 64-bit signed integers using Min() method.

C# Program

C# Program

```
using System;

class Example {
    static void Main(string[] args) {
        Int64 val1 = 7;
        Int64 val2 = 24;

        Int64 result = Math.Min(val1, val2);
        Console.WriteLine($"Minimum of {val1} and {val2} is {result}.");
    }
}
```

Output

```
Minimum of 7 and 24 is 7.
```

Min(SByte, SByte)

Math.Min(val1, val2) returns the smaller of two 8-bit signed integers: `val1` and `val2`.

Syntax

The syntax of Min() method is

```
Math.Min(SByte val1, SByte val2)
```

where

Parameter	Description
val1	The first of two 8-bit signed integers to compare.
val2	The second of two 8-bit signed integers to compare.

Return Value

The method returns val1 or val2, whichever is minimum.

Example 7 – Min(SByte val1, SByte val2)

In this example, we will find the smallest of two 8-bit signed integers using Min() method.

C# Program

```
using System;

class Example {
    static void Main(string[] args) {
        sbyte val1 = -7;
        sbyte val2 = -4;

        sbyte result = Math.Min(val1, val2);
        Console.WriteLine($"Minimum of {val1} and {val2} is {result}.");
    }
}
```

Output

```
Minimum of -7 and -4 is -7.
```

Min(Single, Single)

Math.Min(val1, val2) returns the smaller of two single-precision floating-point numbers: `val1` and `val2`.

Syntax

The syntax of Min() method is

```
Math.Min(Single val1, Single val2)
```

where

Parameter	Description
val1	The first of two single-precision floating-point numbers to compare.
val2	The second of two single-precision floating-point numbers to compare.

Return Value

The method returns val1 or val2, whichever is minimum.

Example 8 – Min(Single val1, Single val2)

In this example, we will find the smallest of two single-precision floating-point numbers using Min() method.

C# Program

```
using System;

class Example {
    static void Main(string[] args) {
        Single val1 = 7.1F;
        Single val2 = 7.12F;

        Single result = Math.Min(val1, val2);
        Console.WriteLine($"Minimum of {val1} and {val2} is {result}.");
    }
}
```

Output

```
Minimum of 7.1 and 7.12 is 7.1.
```

Min(UInt16, UInt16)

Math.Min(val1, val2) returns the smaller of two 16-bit unsigned integers: `val1` and `val2` .

Syntax

The syntax of Min() method is

```
Math.Min(UInt16 val1, UInt16 val2)
```

where

Parameter	Description
val1	The first of two 16-bit unsigned integers to compare.
val2	The second of two 16-bit unsigned integers to compare.

Return Value

The method returns val1 or val2, whichever is minimum.

Example 9 – Min(UInt16 val1, UInt16 val2)

In this example, we will find the smallest of two 16-bit unsigned integers using Min() method.

C# Program

```
using System;

class Example {
    static void Main(string[] args) {
        UInt16 val1 = 7;
        UInt16 val2 = 8;

        UInt16 result = Math.Min(val1, val2);
        Console.WriteLine($"Minimum of {val1} and {val2} is {result}.");
    }
}
```

Output

```
Minimum of 7 and 8 is 7.
```

Min(UInt32, UInt32)

Math.Min(val1, val2) returns the smaller of two 32-bit unsigned integers: `val1` and `val2`.

Syntax

The syntax of Min() method is

```
Math.Min(UInt32 val1, UInt32 val2)
```

where

Parameter	Description
val1	The first of two 32-bit unsigned integers to compare.
val2	The second of two 32-bit unsigned integers to compare.

Return Value

The method returns val1 or val2, whichever is minimum.

Example 10 – Min(UInt32 val1, UInt32 val2)

In this example, we will find the smallest of two 32-bit unsigned integers using Min() method.

C# Program

```
using System;

class Example {
    static void Main(string[] args) {
        UInt32 val1 = 7;
        UInt32 val2 = 8;

        UInt32 result = Math.Min(val1, val2);
        Console.WriteLine($"Minimum of {val1} and {val2} is {result}.");
    }
}
```

Output

```
Minimum of 7 and 8 is 7.
```

Min(UInt64, UInt64)

Math.Min(val1, val2) returns the smaller of two 64-bit unsigned integers: `val1` and `val2`.

Syntax

The syntax of Min() method is

```
Math.Min(UInt64 val1, UInt64 val2)
```

where

Parameter	Description
val1	The first of two 64-bit unsigned integers to compare.
val2	The second of two 64-bit unsigned integers to compare.

Return Value

The method returns val1 or val2, whichever is minimum.

Example 11 – Min(UInt64 val1, UInt64 val2)

In this example, we will find the smallest of two 64-bit unsigned integers using Min() method.

C# Program

```
using System;

class Example {
    static void Main(string[] args) {
        UInt64 val1 = 7;
        UInt64 val2 = 8;

        UInt64 result = Math.Min(val1, val2);
        Console.WriteLine($"Minimum of {val1} and {val2} is {result}.");
    }
}
```

Output

```
Minimum of 7 and 8 is 7.
```

Conclusion

In this C# Tutorial, we have learnt the syntax of C# Math.Min() method, and also learnt how to use this method, with the help of C# example programs.

C# Math

- C# Math.Abs()
- C# Math.Acos()
- C# Math.Acosh()
- C# Math.Asin()
- C# Math.Asinh()
- C# Math.Atan()
- C# Math.Atan2()
- C# Math.Atanh()
- C# Math.BigMul()
- C# Math.BitDecrement()
- C# Math.BitIncrement()
- C# Math.Cbrt()
- C# Math.Ceiling()