Data Types

A Data Type Specifies the size of and type of variable values.

INTEGERAL DATA TYPE

- SIGNED INTEGERS (which takes negative and positive values)
- UNSIGNED INTEGERS (which only takes positive values)
  - SBYTE
  - BYTE
  - SHORT
  - USHORT
  - INT
  - UINT
  - LONG
  - ULONG

Type	Range	Size	Description
sbyte	-128 to 127	Signed 8-bit integer	Stores small signed integers
byte	0 to 255	Unsigned 8-bit integer	Stores small positive integers
char	U+0000 to U+ffff	Unicode 16-bit character	Stores a single character
short	-32,768 to 32,767	Signed 16-bit integer	Stores small signed integers
ushort	0 to 65,535	Unsigned 16-bit integer	Stores small positive integers
int	-2,147,483,648 to 2,147,483,647	Signed 32-bit integer	Common integer type
uint	0 to 4,294,967,295	Unsigned 32-bit integer	Positive-only integers
long	-9,223,372,036,854,775,808 to 9,223,372,036,854,775,807	Signed 64-bit integer	Large signed integers
ulong	0 to 18,446,744,073,709,551,615	Unsigned 64-bit integer	Large positive integers

☑ The highlighted part in red (ulong) represents the unsigned 64-bit integer type with a range from 0 to 18,446,744,073,709,551,615.

How to find max value and min value in C# ?

There is a method

- MINVALUE ()
- MAXVALUE ()

BOOLEAN DATA TYPE

Bool keyword is used for Boolean data type which store only True OR False.

FLOAT DOUBLE AND DECIMAL DATA TYPES

C# Alias	.NET Type	Size	Precision
float	System.Single	4 bytes	≈ 7 digits
double	System.Double	8 bytes	≈ 15–16 digits
decimal	System.Decimal	16 bytes	≈ 28–29 decimal places

STRING AND CHARACTER DATA TYPE

- **String**
  - Stores multiple characters in a single variable.
  - Declared using double quotes (" ").

    Example:

    ```
    string name = "Subhash";
    ```
- **char**
  - Stores a single character at a time.
  - Declared using single quotes (' ').

    Example:

    ```
    char grade = 'A';
    ```

- **Escape Sequence**

Used to represent special characters inside strings, like:

Escape	Meaning
\n	New line
\t	Tab
\\	Backslash
\"	Double quote

Escape	Meaning
\'	Single quote

Example:

```
string path = "D:\\Csharp\\Tutorials";
```

- 📌 Verbatim Literal
  - A string preceded by @ symbol.
  - It treats escape sequences as normal text and improves readability.

  Example:

  ```
  string path = @"D:\Csharp\Tutorials";
  ```

☑ Practical Example Comparison:

Without Verbatim Literal	With Verbatim Literal
"D:\\Adil\\Csharp\\Tutorials" – less readable	@"D:\Adil\Csharp\Tutorials" – more readable

🔄 DATA TYPES CONVERSION IN C# PROGRAMMING

There are two types of conversions in C#:

1. Implicit Conversion
2. Explicit Conversion

☸ Implicit Conversion (Type Casting done by the compiler)

Implicit conversion is automatically handled by the compiler when:

1. There is no loss of information during the conversion.
2. There is no risk of an exception being thrown.

☑ Example:

```
int num = 10;
float result = num;   // Implicit conversion (int → float)
```

Here, the conversion is safe — no data loss occurs.

⚠ Explicit Conversion (Type Casting done manually by the programmer)

Explicit conversion is required when there's a risk of data loss or overflow.

Example:

```
float value = 12.75f;

int number = (int)value;   // Explicit conversion (float → int)
```

- The fractional part (0.75) will be lost.
- Hence, the conversion must be done explicitly using (int).

Summary:

Type	Who Performs It	Data Loss	Example	Syntax
Implicit	Compiler	✗ No	int → float	float f = i;
Explicit	Programmer	☑ Possible	float → int	int i = (int)f;

CONSTANT in C#

Definition:

A constant has a fixed value that remains unchanged throughout the execution of the program.

Key Points:

- In C#, constants can be declared for all data types.
- You must initialize a constant at the time of declaration — it cannot be left unassigned.
- Constants are generally declared for value types (like int, float, char) rather than reference types.
- Use the const keyword to declare a constant.

Syntax:

```
const dataType constantName = value;
```

Example:

```
const double PI = 3.14159;

const int MaxScore = 100;
```

Important Notes:

- Constants are compile-time constants (their value is known at compile time).
- Once assigned, their value cannot be changed.

- **If you need a runtime constant, you can use the read-only keyword instead of const.**

Date and time format specifiers

A date and time format specifier is a special character that enables you to display the date and time values in different formats

FORMAT SPECIFIERS	NAME
d	short date
D	long date
f	full date/time (short time)
F	full date/time (long time)
G	General date / time (short time)