Expressions and Operators

- Expressions
- Literals
- Order of Evaluation
- Simple Arithmetic Operators
- The Remainder Operator
- Relational and Equality Comparison Operators
- Increment and Decrement Operators
- Conditional Logical Operators
- Logical Operators
- Shift Operators
- Assignment Operators
- The Conditional Operator
- Unary Arithmetic Operators
- User-Defined Type Conversions
- Operator Overloading
- The typeof Operator
- Other Operators
Expressions

This chapter defines expressions and describes the operators provided by C#. It also explains how you can define the C# operators to work with your user-defined classes.

An expression is a string of operators and operands. The following are some of the constructs that can act as operands:

- Literals
- Constants
- Variables
- Method calls
- Element accessors, such as array accessors and indexers
- Other expressions

The C# operators take one, two, or three operands. An operator does the following:

- Takes its operands as input
- Performs an action
- Returns a value, based on the action

Expressions can be combined, using operators, to create other expressions, as shown in this expression, with three operators and four operands:

\[
\frac{a + b}{expr + c} + d = a + b + c + d
\]

Evaluating an expression is the process of applying each operator to its operands, in the proper sequence, to produce a value.

- The value is returned to the position at which the expression was evaluated. There, it might in turn be an operand in an enclosing expression.
- Besides the value returned, some expressions also have side effects, such as setting a value in memory.