C divides the operators into the following groups:

* Arithmetic operators
* Assignment operators
* Comparison operators
* Logical operators
* Bitwise operators

## Arithmetic Operators

Arithmetic operators are used to perform common mathematical operations.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Operator** | **Name** | **Description** | **Example** | **Try it** |
| + | Addition | Adds together two values | x + y | [Try it »](https://www.w3schools.com/c/tryc.php?filename=demo_oper_add) |
| - | Subtraction | Subtracts one value from another | x - y | [Try it »](https://www.w3schools.com/c/tryc.php?filename=demo_oper_sub) |
| \* | Multiplication | Multiplies two values | x \* y | [Try it »](https://www.w3schools.com/c/tryc.php?filename=demo_oper_mult) |
| / | Division | Divides one value by another | x / y | [Try it »](https://www.w3schools.com/c/tryc.php?filename=demo_oper_div) |
| % | Modulus | Returns the division remainder | x % y | [Try it »](https://www.w3schools.com/c/tryc.php?filename=demo_oper_mod) |
| ++ | Increment | Increases the value of a variable by 1 | ++x | [Try it »](https://www.w3schools.com/c/tryc.php?filename=demo_oper_inc) |
| -- | Decrement | Decreases the value of a variable by 1 | --x | [Try it »](https://www.w3schools.com/c/tryc.php?filename=demo_oper_dec) |

## Assignment Operators

Assignment operators are used to assign values to variables.

In the example below, we use the **assignment** operator (=) to assign the value **10** to a variable called **x**:

### Example

int x = 10;

[Try it Yourself »](https://www.w3schools.com/c/tryc.php?filename=demo_oper_assignment1)

The **addition assignment** operator (+=) adds a value to a variable:

### Example

int x = 10;  
x += 5;

[Try it Yourself »](https://www.w3schools.com/c/tryc.php?filename=demo_oper_assignment2)

A list of all assignment operators:

|  |  |  |  |
| --- | --- | --- | --- |
| **Operator** | **Example** | **Same As** | **Try it** |
| = | x = 5 | x = 5 | [Try it »](https://www.w3schools.com/c/tryc.php?filename=demo_oper_ass1) |
| += | x += 3 | x = x + 3 | [Try it »](https://www.w3schools.com/c/tryc.php?filename=demo_oper_ass2) |
| -= | x -= 3 | x = x - 3 | [Try it »](https://www.w3schools.com/c/tryc.php?filename=demo_oper_ass3) |
| \*= | x \*= 3 | x = x \* 3 | [Try it »](https://www.w3schools.com/c/tryc.php?filename=demo_oper_ass4) |
| /= | x /= 3 | x = x / 3 | [Try it »](https://www.w3schools.com/c/tryc.php?filename=demo_oper_ass5) |
| %= | x %= 3 | x = x % 3 | [Try it »](https://www.w3schools.com/c/tryc.php?filename=demo_oper_ass6) |
| &= | x &= 3 | x = x & 3 | [Try it »](https://www.w3schools.com/c/tryc.php?filename=demo_oper_ass7) |
| |= | x |= 3 | x = x | 3 | [Try it »](https://www.w3schools.com/c/tryc.php?filename=demo_oper_ass8) |
| ^= | x ^= 3 | x = x ^ 3 | [Try it »](https://www.w3schools.com/c/tryc.php?filename=demo_oper_ass9) |
| >>= | x >>= 3 | x = x >> 3 | [Try it »](https://www.w3schools.com/c/tryc.php?filename=demo_oper_ass10) |
| <<= | x <<= 3 | x = x << 3 | [Try it »](https://www.w3schools.com/c/tryc.php?filename=demo_oper_ass11) |

## Comparison Operators

Comparison operators are used to compare two values.

**Note:** The return value of a comparison is either true (1) or false (0).

In the following example, we use the **greater than** operator (>) to find out if 5 is greater than 3:

### Example

int x = 5;  
int y = 3;  
printf("%d", x > y); // returns 1 (true) because 5 is greater than 3

[Try it Yourself »](https://www.w3schools.com/c/tryc.php?filename=demo_oper_compare3)

A list of all comparison operators:

|  |  |  |  |
| --- | --- | --- | --- |
| **Operator** | **Name** | **Example** | **Try it** |
| == | Equal to | x == y | [Try it »](https://www.w3schools.com/c/tryc.php?filename=demo_oper_compare1) |
| != | Not equal | x != y | [Try it »](https://www.w3schools.com/c/tryc.php?filename=demo_oper_compare2) |
| > | Greater than | x > y | [Try it »](https://www.w3schools.com/c/tryc.php?filename=demo_oper_compare3) |
| < | Less than | x < y | [Try it »](https://www.w3schools.com/c/tryc.php?filename=demo_oper_compare4) |
| >= | Greater than or equal to | x >= y | [Try it »](https://www.w3schools.com/c/tryc.php?filename=demo_oper_compare5) |
| <= | Less than or equal to | x <= y | [Try it »](https://www.w3schools.com/c/tryc.php?filename=demo_oper_compare6) |

## Logical Operators

Logical operators are used to determine the logic between variables or values:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Operator** | **Name** | **Description** | **Example** | **Try it** |
| && | Logical and | Returns true if both statements are true | x < 5 &&  x < 10 | [Try it »](https://www.w3schools.com/c/tryc.php?filename=demo_oper_logical1) |
| || | Logical or | Returns true if one of the statements is true | x < 5 || x < 4 | [Try it »](https://www.w3schools.com/c/tryc.php?filename=demo_oper_logical2) |
| ! | Logical not | Reverse the result, returns false if the result is true | !(x < 5 && x < 10) | [Try it »](https://www.w3schools.com/c/tryc.php?filename=demo_oper_logical3) |

## Sizeof Operator

The memory size (in bytes) of a data type or a variable can be found with the sizeof operator:

### Example

int myInt;  
float myFloat;  
double myDouble;  
char myChar;  
  
printf("%lu\n", sizeof(myInt));  
printf("%lu\n", sizeof(myFloat));  
printf("%lu\n", sizeof(myDouble));  
printf("%lu\n", sizeof(myChar));

Note that we use the %lu format specifer to print the result, instead of %d. It is because the compiler expects the sizeof operator to return a long unsigned int (%lu), instead of int (%d). On some computers it might work with %d, but it is safer to use %lu.