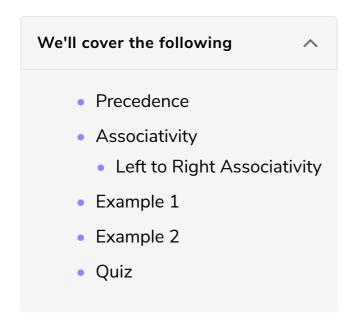
Precedence and Associativity

This lesson discusses two important characteristics of operators - precedence and associativity.



Precedence

The precedence of an operator determines which operation is performed first in an expression with more than one operators.

Operators are listed below in the order of their precedence from highest to lowest:

```
Unary

Logical/Bitwise NOT - !
Derereference - *
Borrow - &, &mut

Binary

Typecast - as
Multiplication - *, Division - /, Remainder - %
Addition - +, Subtraction - -
Left Shift - <<, Right Shift - >>
Bitwise AND - &
Bitwise XOR - ^
Bitwise OR - |
```

o Comparison - == != < > <= >=

```
    Logical AND - &&
    Logical OR - ||
    Range - start .. stop
    Assignment/Compound Assignment - = += -= *= /= %= &= |= ^= <<= >>=
```

Note: The operators that are written in the same row have the same order of precedence.

Note: The range operator will be further explored while we discuss the for loop in the Loops chapter.

Associativity

If two or more operators of the same precedence appear in a statement, then which operator will be evaluated first is defined by the associativity.

Left to Right Associativity

Left associativity occurs when an expression is evaluated from left to right. An expression such as $a \sim b \sim c$, in this case, would be interpreted as $(a \sim b) \sim c$ where \sim can be any operator.

The operators below can be chained as left associative.

- as
- *, /, %
- +, -
- << >>
- &
- ^
- |
- &&
- []

The companion essignment and the names expenses account to chained a

all.

Example 1

The example below solves an expression according to its operator precedence:

```
fn main() {
   println!("Answer : {}",( 3 + 5 ) * 9 / 7 & 8);
}
```

```
( 3+5 ) * 9 / 7 & 8
8 * 9 / 7 & 8
```

of 4

of 4

```
(3+5) * 9 / 7 & 8

8 * 9 / 7 & 8

72 / 7 & 8

10 & 8

8
```

Example 2#

The example below solves an expression according to its operator precedence:



of 5

of 5

of 5

of 5

```
2 + 3 / 5 ^ 7 & 8 | 9

2 + 0 ^ 7 & 8 | 9

2 ^ 7 & 8 | 9

2 ^ 0 | 9

2 | 9

11
```



Quiz

Test your understanding of operator precedence in Rust!

Quick Quiz on Operator Precedence!



What is the output of the following code according to its operator precedence in Rust?

```
fn main() {
    println!("{}", 3 + 4 - 9 / 6 * 6 ^ 8 & 3);
}
```

