Arithmetic Operator Precedence and Associativity in Python

1. **Parentheses** `()`

   Parentheses have the highest precedence and are evaluated first.

   ```
   result = (2 + 3) * 4
   # 5 * 4
   # Output: 20
   ```

   Here, the expression inside the parentheses `(2 + 3)` is evaluated first, resulting in `5`, and then `5 * 4` is computed.

2. **Exponentiation** `*`

   The exponentiation operator has higher precedence than multiplication, division, and addition.

   ```
   result = 2 ** 3 * 4
   # Output: 32
   ```

   Here, `2 ** 3` is evaluated first to `8`, and then `8 * 4` results in `32`.

3. **Multiplication , Division `/`, Floor Division `//`, and Modulus `%`**

   These operators have the same precedence and are evaluated from left to right.

   ```
   result = 10 / 2 * 3
   # Output: 15.0
   ```

   Here, `10 / 2` is computed first to `5.0`, and then `5.0 * 3` results in `15.0`.

   ```
   result = 10 // 3 % 2
   # Output: 1
   ```

   Here, `10 // 3` is computed first to `3`, and then `3 % 2` results in `1`.

4. **Addition `+` and Subtraction**

   These operators have lower precedence and are also evaluated from left to right.

   ```
   result = 5 + 3 - 2
   # Output: 6
   ```

   Here, `5 + 3` is computed first to `8`, and then `8 - 2` results in `6`.

5. **Assignment `=`**

   The assignment operator has the lowest precedence and is evaluated last.

   ```
   a = 2 + 3
   b = a * 4
   # a = 5, b = 20
   ```

   Here, `2 + 3` is computed first to `5` and assigned to `a`, and then `a * 4` is computed to `20` and assigned to `b`.

Combining Multiple Operators

To illustrate how different operators interact, here are some combined examples:

1. **Without Parentheses**

```
result = 2 + 3 * 4 ** 2 // 5 % 3
#2 + 3 * 16 // 5 % 3
#2 + 48 // 5 % 3
#2 + 9 % 3
#2 + 0
#2
# Output: 2
```

- `4 ** 2` is evaluated first to `16`.
- `16 // 5` is evaluated next to `3`.
- `3 % 3` is evaluated next to `0`.
- `2 + 0` results in `2`.

2. **With Parentheses to Change Precedence**

```
result = (2 + 3) * (4 ** 2) // (5 % 3)
# 5 * 16 // 2
# Output: 40
```

- `(2 + 3)` evaluates first to `5`.
- `4 ** 2` evaluates to `16`.
- `5 % 3` evaluates to `2`.
- `5 * 16` evaluates to `80`.
- `80 // 2` evaluates to `40`.

3. **Associativity of Exponentiation and Assignment**

```
result = 2 ** 3 ** 2
# Output: 512
```

- Right-to-left associativity for `*` means `3 ** 2` is computed first to `9`.
- `2 ** 9` evaluates to `512`.

```
a = 2
b = a ** 2
```

```
a = 3
# a = 3, b = 4
```