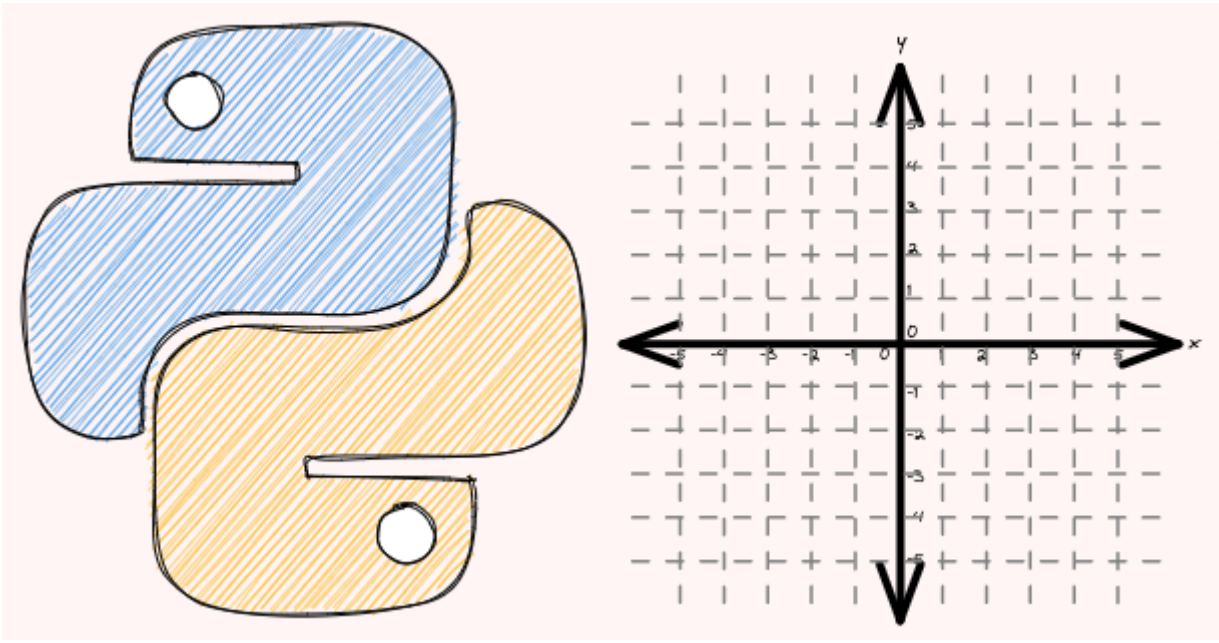
# Efficient Python Tricks and Tools for Data Scientists - By Khuyen Tran

*Number*



# *fractions: Get Numerical Results in Fractions Instead of Decimals*

Normally, when you divide a number by another number, you will get a decimal:

```
2 / 3 + 1
```

```
1.6666666666666665
```

Sometimes, you might prefer to get the results in fractions instead of decimals. There is Python built-in function called `fractions` that allows you to do exactly that.

```
from fractions import Fraction

res = Fraction(2 / 3 + 1)
print(res)
```

```
3752999689475413/2251799813685248
```

Cool! We got a fraction instead of a decimal. To limit the number of decimals displayed, use `limit_denominator()`.

```
res = res.limit_denominator()
print(res)
```

```
5/3
```

If we divide the result we got from `Fraction` by another number, we got back a fraction without using the `Fraction` object again.

```
print(res / 3)
```

```
5/9
```

# *How to Use Underscores to Format Large Numbers in Python*

When working with a large number in Python, it can be difficult to figure out how many digits that number has. Python 3.6 and above allows you to use underscores as visual separators to group digits.

In the example below, I use underscores to group decimal numbers by thousands.

```
large_num = 1_000_000
large_num
```

```
1000000
```

# *Confirm Whether a Variable Is a Number*

If you want to confirm whether a variable is a number without caring whether it is a float or an integer, numbers, use `numbers.Number`.

```
from numbers import Number

a = 2
b = 0.4

# Check if a is a number
isinstance(a, Number)
```

```
True
```

```
# Check if b is a number
isinstance(b, Number)
```

```
True
```

# *Get Multiples of a Number Using Modulus*

If you want to get multiples of a number, use the modulus operator `%`. The modulus operator is used to get the remainder of a division. For example, `4 % 3 = 1`, `5 % 3 = 2`.

Thus, to get multiples of `n`, we select only numbers whose remainders are 0 when dividing them by `n`.

```
def get_multiples_of_n(nums: list, n: int):
    """Select only numbers whose remainders
    are 0 when dividing them by n"""
    return [num for num in nums if num % n ==
0]


nums = [1, 4, 9, 12, 15, 16]
```

```
get_multiples_of_n(nums, 2)  # multiples of 2
```

```
[4, 12, 16]
```

```
get_multiples_of_n(nums, 3)  # multiples of 3
```

```
[9, 12, 15]
```

```
get_multiples_of_n(nums, 4)  # multiples of 4
```

```
[4, 12, 16]
```