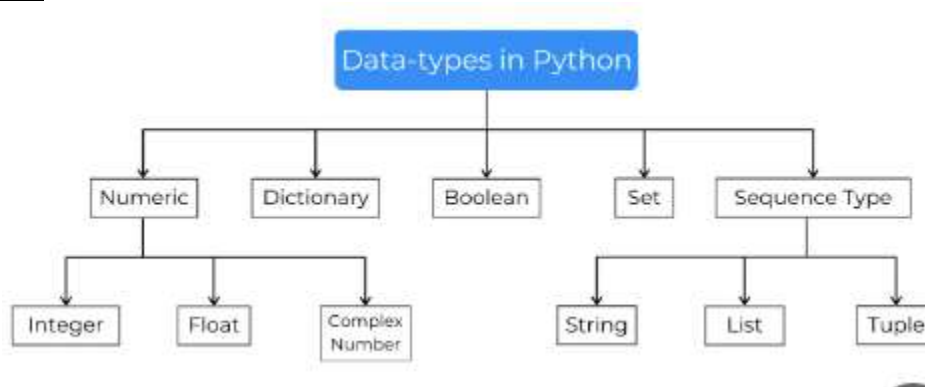


## Arithmetic operators

- + Addition
- Subtraction
- Multiplication
- / Division
- % Mod (the remainder after dividing)
- \*\* Exponentiation (note that ^ does not do this operation, as you might have seen in other languages)
- // Divides and rounds down to the nearest integer

## Data Types



## Important python methods

Strings	Lists	Sets	Dictionaries
1. startswith()	1. append()	1. add()	1. clear()
2. endswith()	2. clear()	2. clear()	2. copy()
3. count()	3. copy()	3. copy()	3. fromkeys()
4. upper(), lower(), title()	4. count()	4. difference()	4. get()
5. find(), rfind()	5. extend()	5. discard()	5. items()
6. index()	6. index()	6. intersection()	6. keys()
7. strip(),rstrip(), lstrip()	7. insert()	7. issubset()	7. pop()
8. split(), rsplit()	8. pop()	8. issuperset()	8. popitem()
9. replace()	9. remove()	9. pop()	9. setdefault()
10. count()	10. reverse()	10. remove()	10. update()
	11. sort()	11. union()	11. values()

# Data Structures

## lists

- Lists are used to store multiple items in a single variable.
- List items are ordered, changeable, and allow duplicate values.
- List comprehension offers a shorter syntax when you want to create a new list based on the values of an existing list. Ex: `newlist = [expression for item in iterable if condition == True]`

## Tuple

- Tuples are used to store multiple items in a single variable.
- Tuple items are ordered, unchangeable, and allow duplicate values.
- Tuples are unchangeable, meaning that we cannot change, add or remove items after the tuple has been created.

## Set

- Sets are used to store multiple items in a single variable.
- Set items are unordered, unchangeable, and **do not allow duplicate values.**
- Unordered means that the items in a set do not have a defined order.
- Set items are unchangeable, meaning that we cannot change the items after the set has been created.

## Dictionary

- Dictionaries are used to store data values in key:value pairs.
- Dictionaries are written with curly brackets, and have keys and values:
- Dictionary items are ordered, changeable, and **does not allow duplicates.**

## Loops

- while loops
- for loops

With the **while** loop we can execute a set of statements as long as a condition is true.

**Break:** we can stop the loop even if the while condition is true

**Continue:** we can stop the current iteration, and continue with the next.

## Conditions

Python supports the usual logical conditions from mathematics:

**Logical operators: *and* or not**

If ... Else

Short Hand If: If you have only one statement to execute, you can put

it on the same line as the if statement. Ex: `print("A") if a > b else print("B")`

- Equals: `a == b`
- Not Equals: `a != b`
- Less than: `a < b`
- Less than or equal to: `a <= b`
- Greater than: `a > b`
- Greater than or equal to: `a >= b`