Concept: Immutability

Immutability refers to the state of being unchangeable: Once created, immutable objects cannot be modified any longer.

Python and immutability

In Python, immutability applies to certain built-in data types.

Immutable Types

- Numbers (int, float, complex)
- Strings
- Tuples

Mutable Types

- Lists
- Dictionaries
- Sets

Good or bad? It depends...

Pros:

- **Predictability**: Immutable objects provide a clear understanding that their state won't change, making code easier to debug and reason about
- **Thread-Safe**: Immutable objects are inherently thread-safe, as concurrent modifications are not a concern.
- **Dictionary Keys**: Immutable objects can be used as keys in dictionaries, enabling efficient data retrieval.

Cons:

- **Memory Usage**: Modifications require creating new objects, which can lead to higher memory consumption.
- **Performance**: For large datasets or frequent operations, the overhead of creating new objects instead of modifying in place can affect performance.

Understanding which types are immutable and their implications is crucial for writing efficient, safe, and clean code.