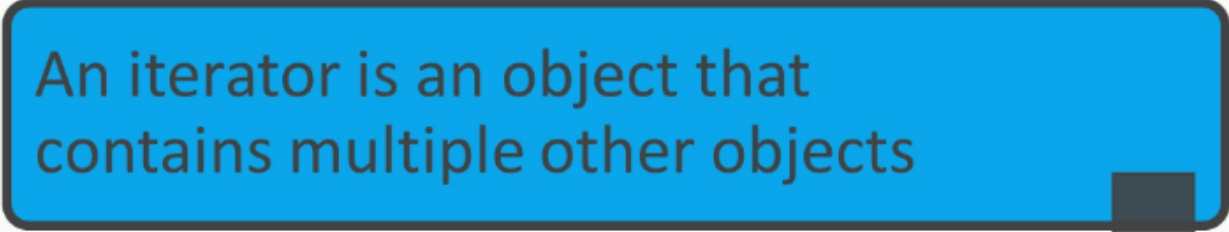


Iterators

Iterator-Collection of elements

An iterator is an object that contains multiple other objects



There are various kind of iterators



Some are built into Python

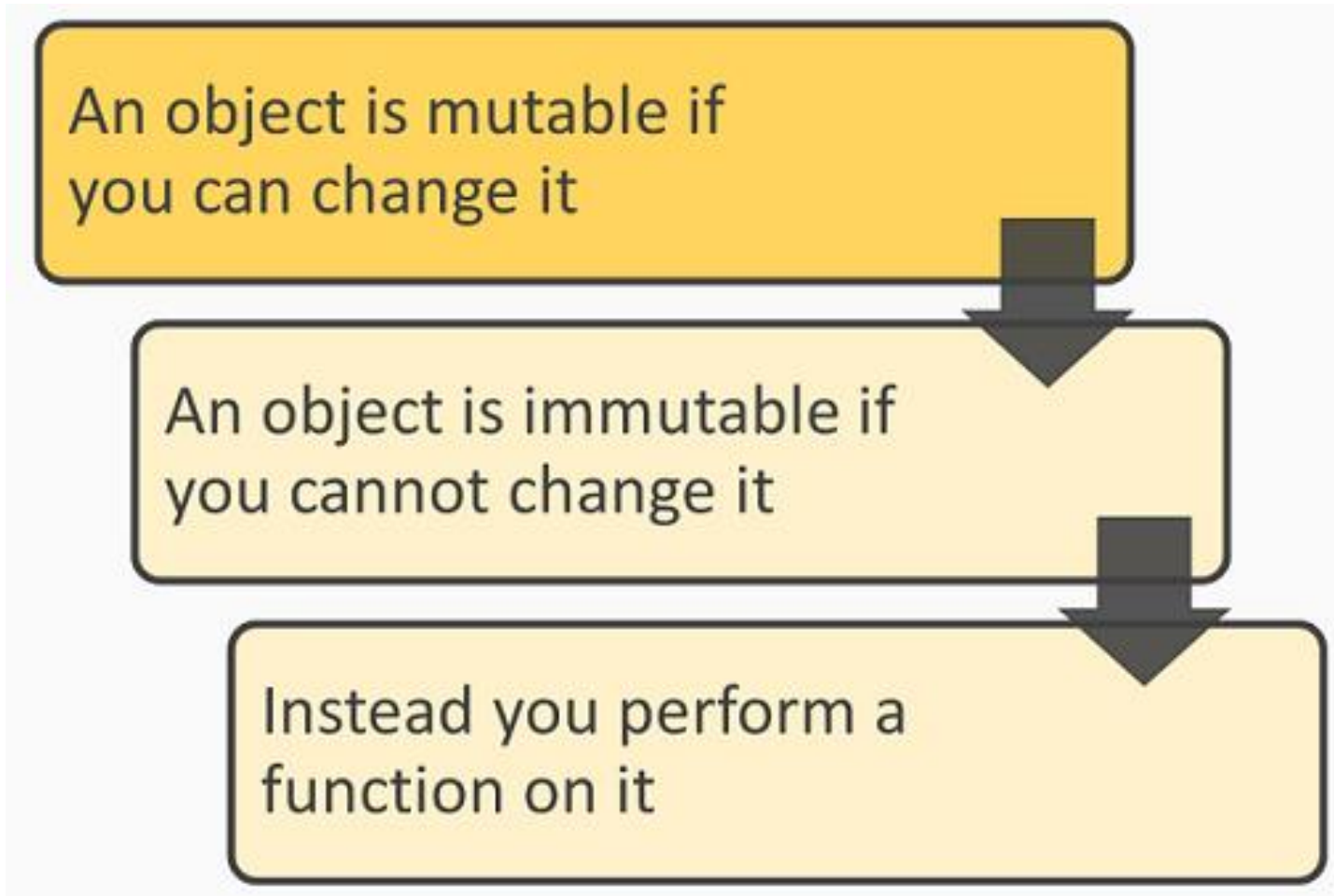


Others you can create yourself

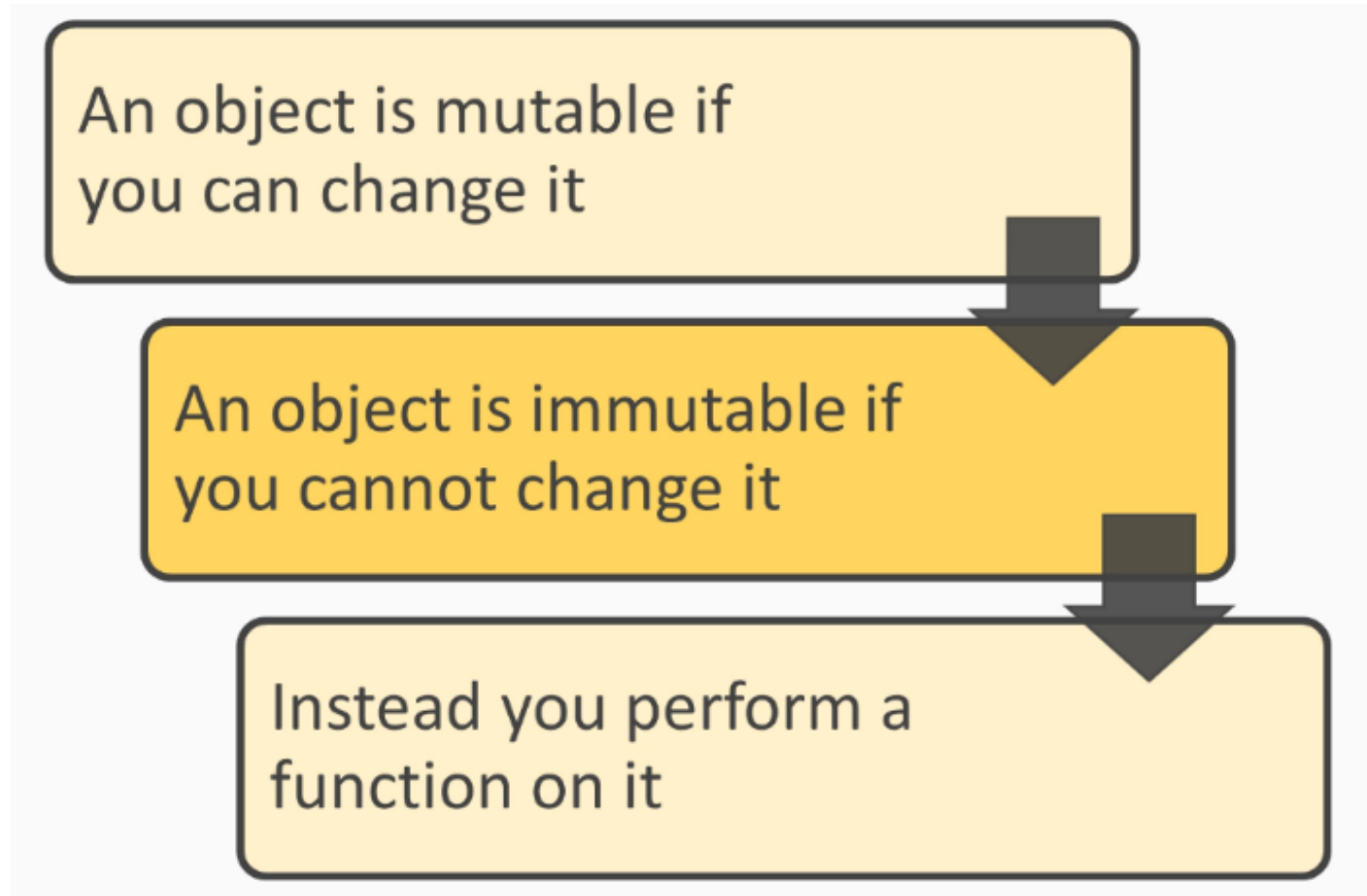


Functional Programming
Relies Heavily on
Iterators!

Mutability – To change or Not change

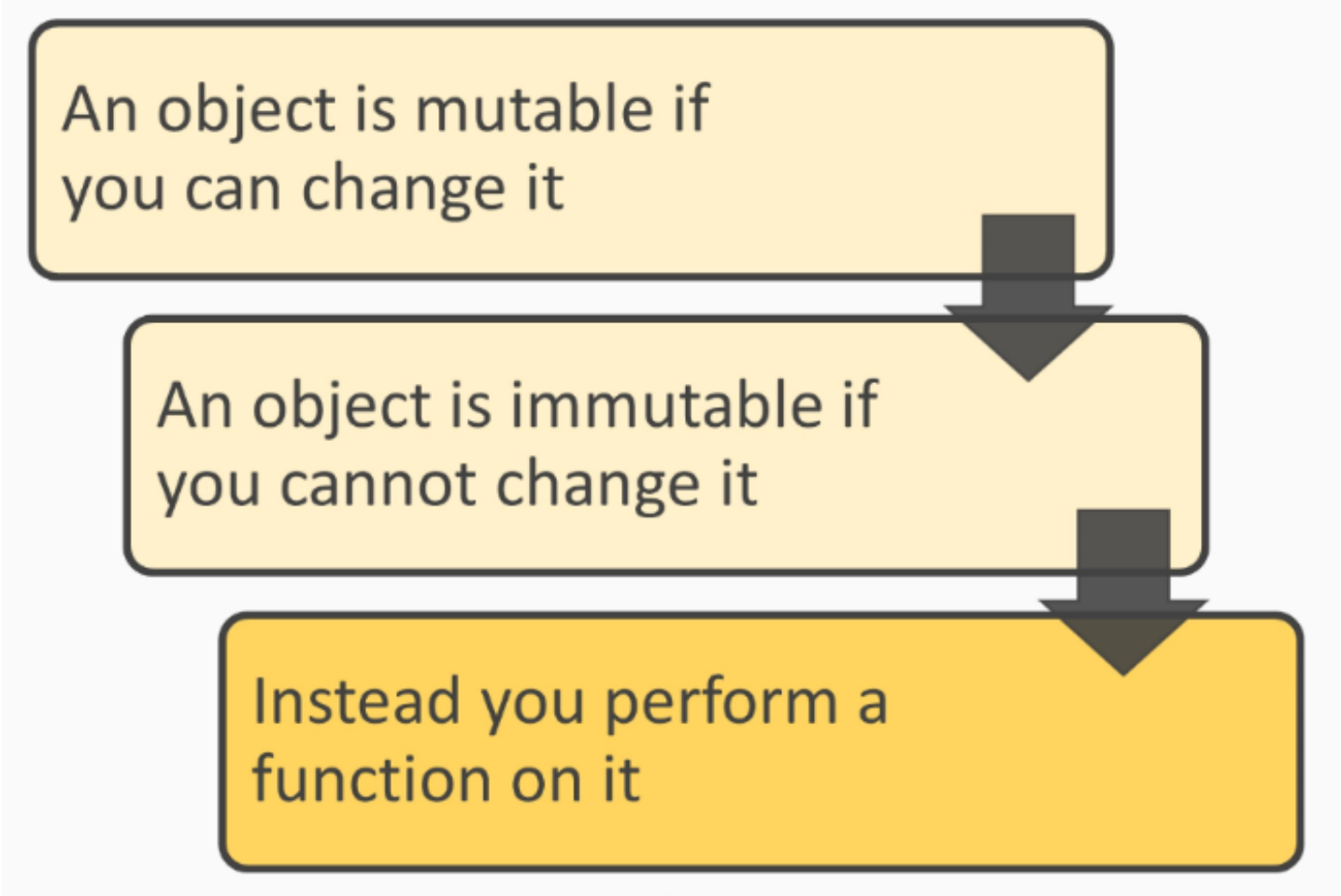


Mutability – To change or Not change



Mutability – To change or Not change

An object is mutable if
you can change it



```
graph TD; A[An object is mutable if you can change it] --> B[An object is immutable if you cannot change it]; B --> C[Instead you perform a function on it];
```

An object is immutable if
you cannot change it

Instead you perform a
function on it

Functional Programming
Doesn't Mutate Objects!

Order

An iterator is ordered if the elements are in a predictable order



An iterator is unordered if the elements are in an unpredictable order

Order

An iterator is ordered if the elements are in a predictable order



An iterator is unordered if the elements are in an unpredictable order

A List Is Mutable
and Ordered

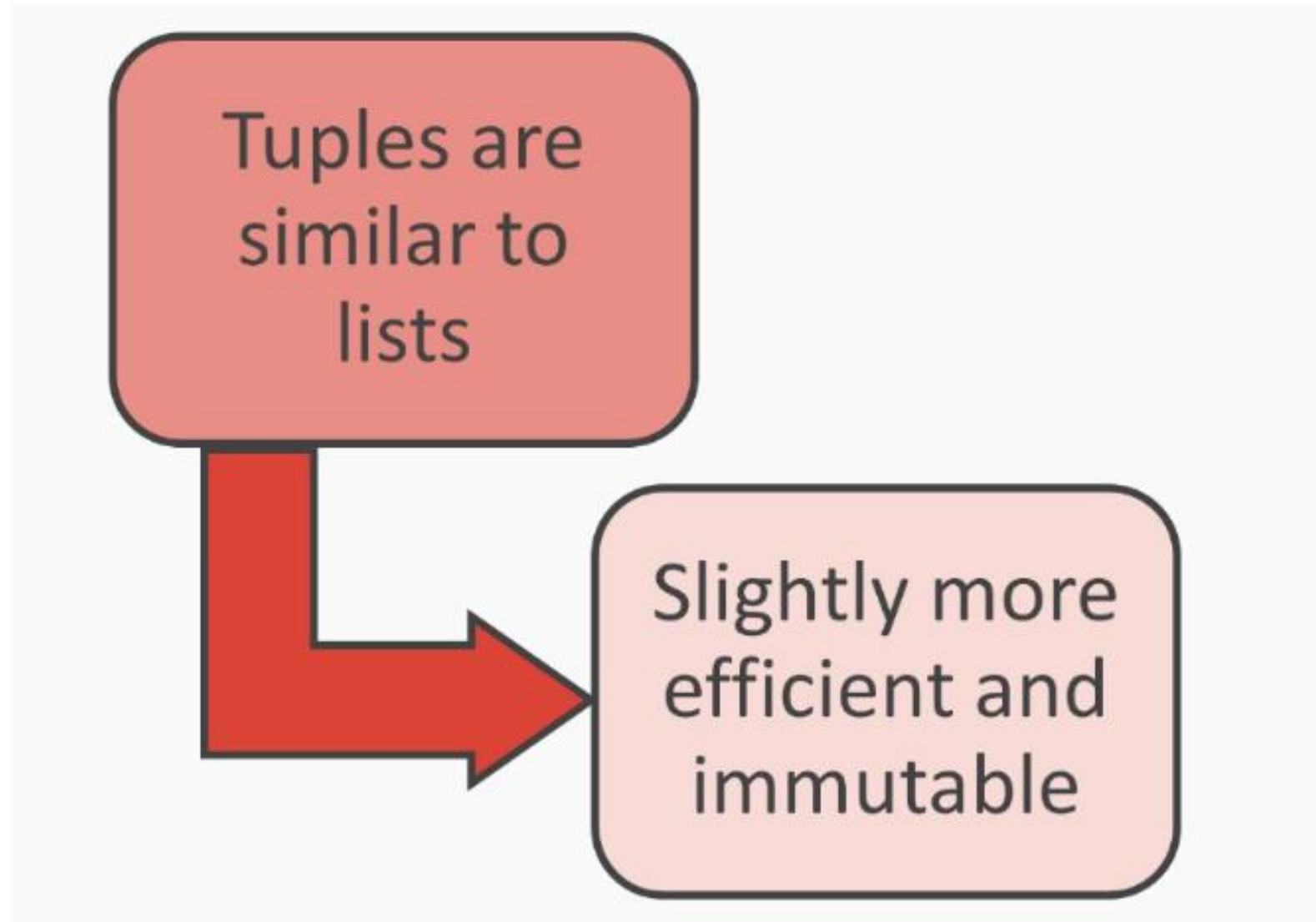
Using a Tuple – Immutable Sequences
of Elements with a Fixed Order

Tuple

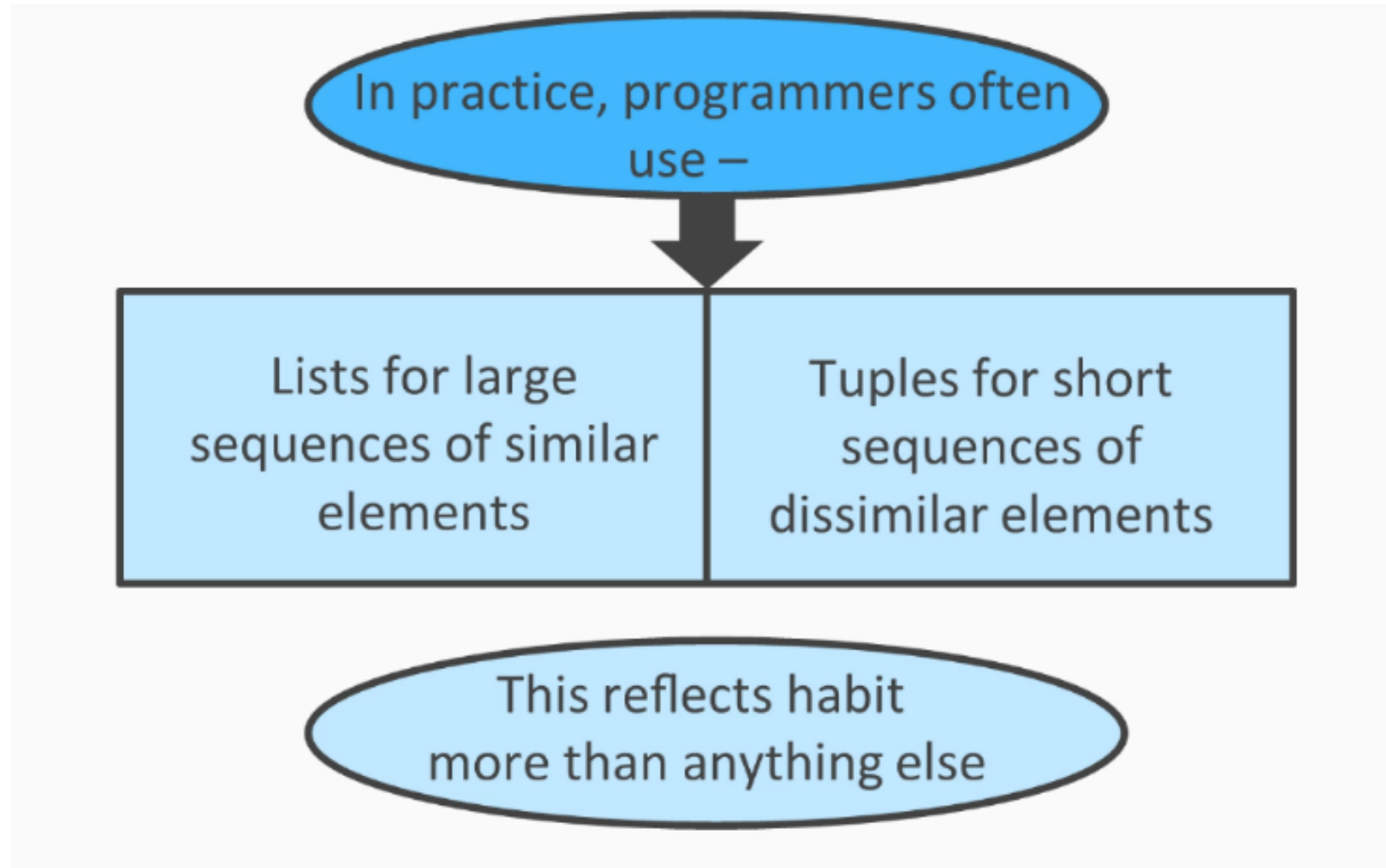
- What a tuple object is
- How you can use tuple objects in real code
- Some more things you can do with sequences (not only tuples)

A Tuple Is Immutable
and Ordered

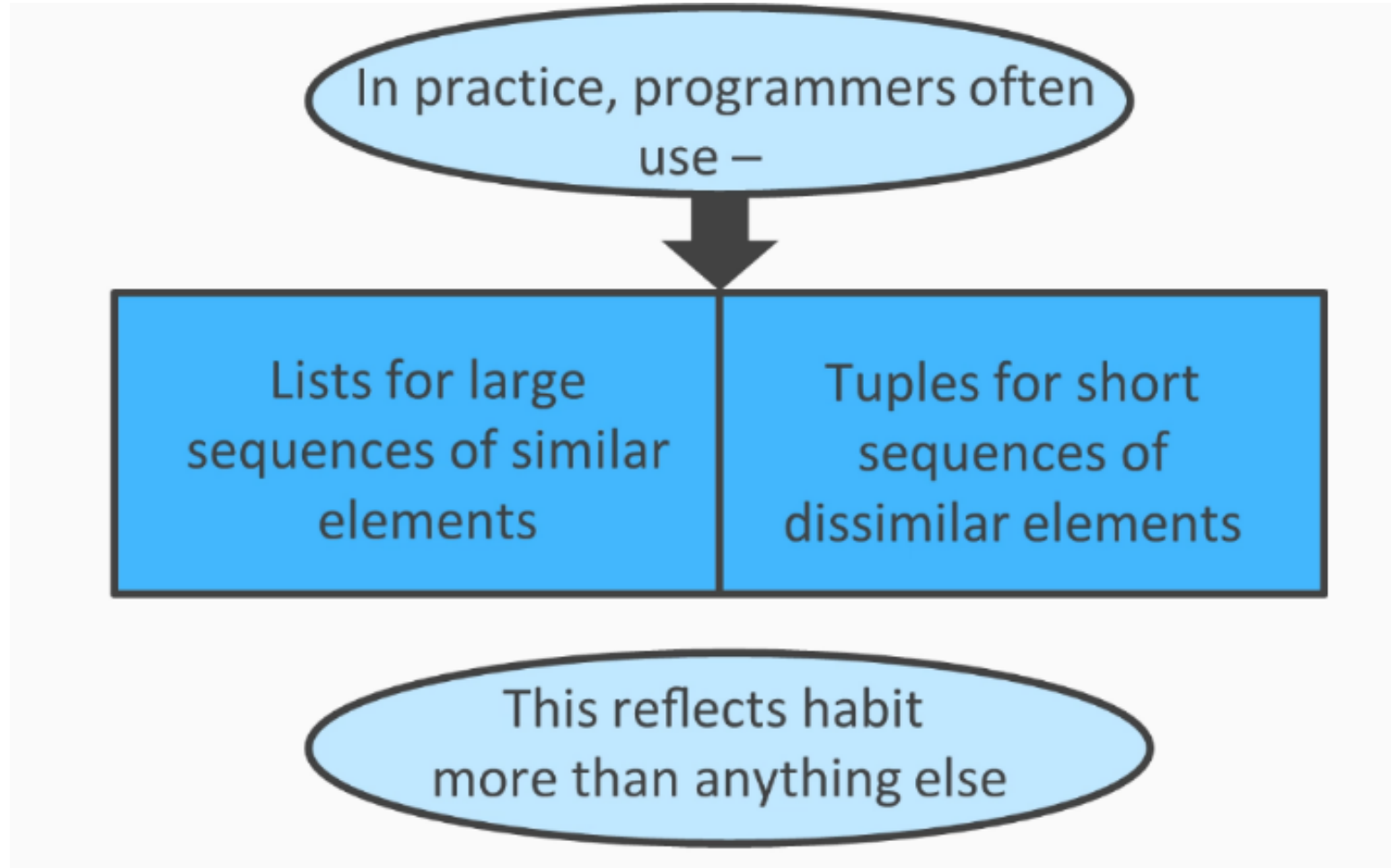
Tuple –The functional way



List or Tuple? –Pick your choice



List or Tuple? –Pick your choice



Using a Dict – Mutable, Key-value
Mappings Without a Fixed Order

Dictionary

- What a dict (dictionary) object is
- How you can use dict objects in real code

A Dict Is a Collection of
Key \Rightarrow Value Mappings

Key an Values

To find an element in a list or tuple, you use the element's position (index).

Elements in a dict don't have a position

Instead, you find elements by their key.

Keys can be (almost) any Python object

Key an Values

To find an element in a list or tuple, you use the element's position (index).

Elements in a dict don't have a position

Instead, you find elements by their key.

Keys can be (almost) any Python object

Key an Values

To find an element in a list or tuple, you use the element's position (index).

Elements in a dict don't have a position

Instead, you find elements by their key.

Keys can be (almost) any Python object

Key an Values

To find an element in a list or tuple, you use the element's position (index).

Elements in a dict don't have a position

Instead, you find elements by their key.

Keys can be (almost) any Python object

No order

Although you can loop through dict objects

The order of the elements is not guaranteed

So don't assume a fixed order

No order

Although you can loop through dict objects

The order of the elements is not guaranteed

So don't assume a fixed order

No order

Although you can loop through dict objects

The order of the elements is not guaranteed

So don't assume a fixed order

Using a Set – Immutable Collections of Unique Elements Without a Fixed Order

Set

- What a set object is
- How you can use set objects in real code

Unpacking Iterators by Assigning to Multiple Variables

Iterator

- Normal iterator unpacking
- Extended iterator unpacking using the `*rest` syntax (Python 3)
- Unpacking nested iterators

Summary

- Explained that iterators are collections of elements and iterators can be mutable and/or ordered
- Explored some built-in iterators such as list, tuple, dict, and set.
- Studies that iterators can be unpacked by having multiple variables on the left-hand side of an assignment