GLS UNIVERSITY

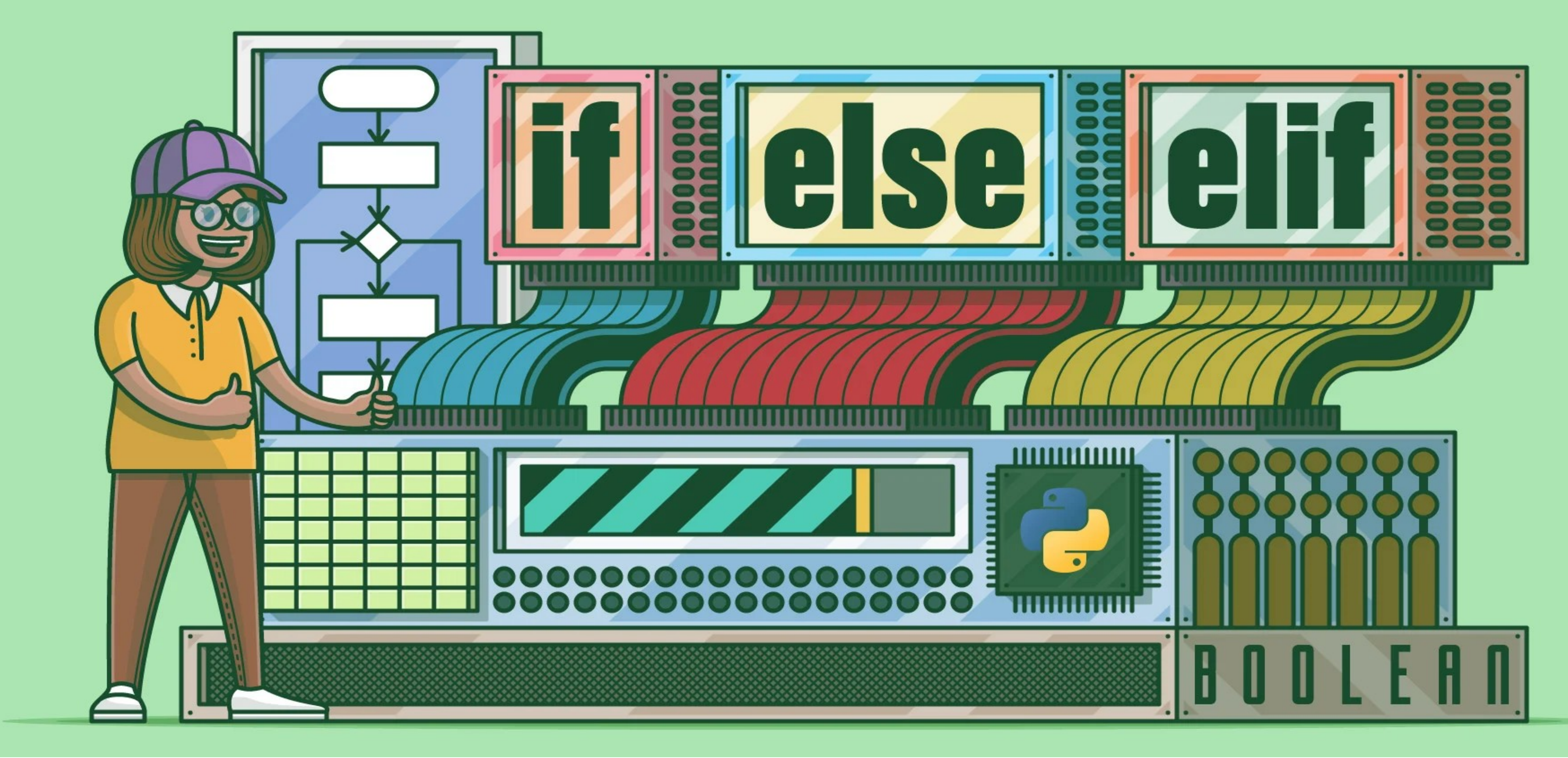
Faculty of Computer Applications & Information Technology

Integrated MCA Programme

Semester V

Unit 2: Python Programming Constructs and Data Structures

Conditional Statements

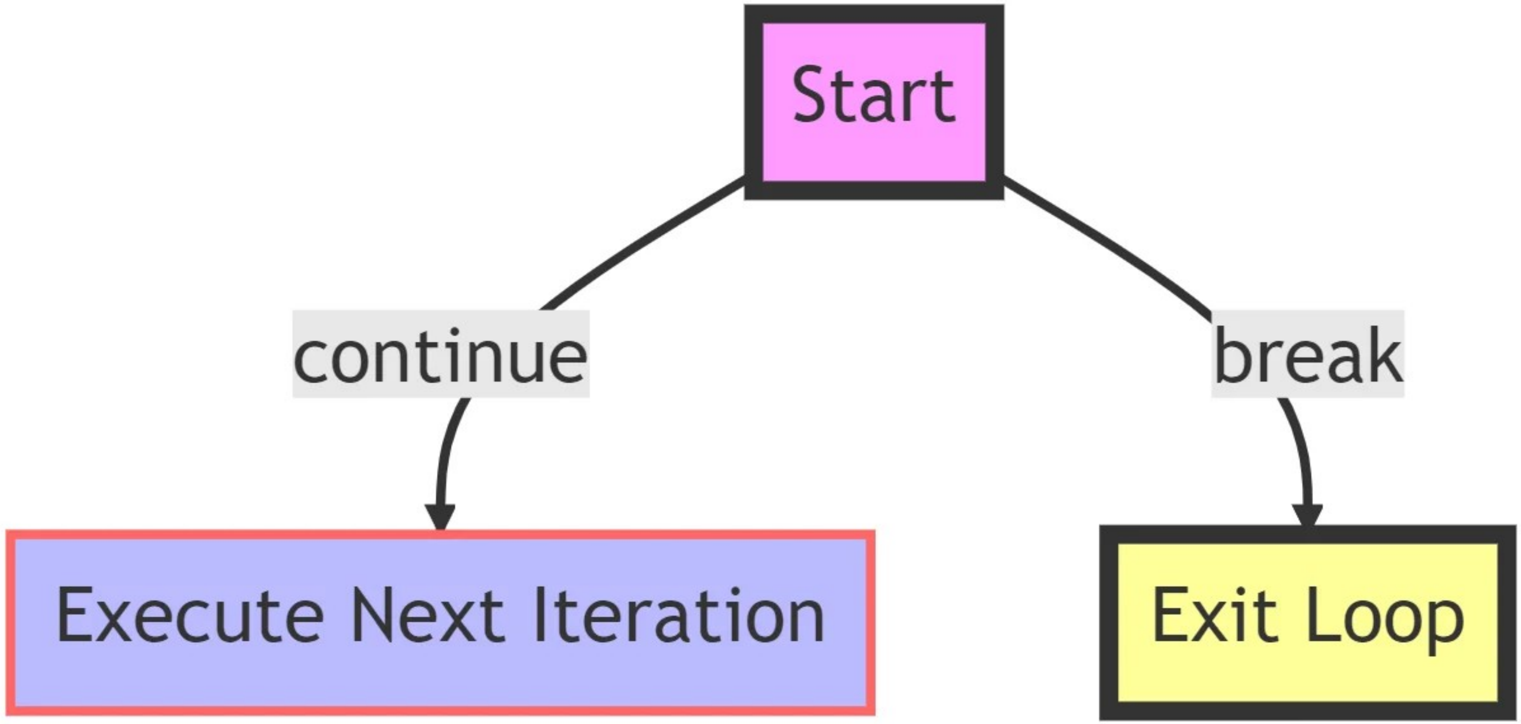


Conditional Statements

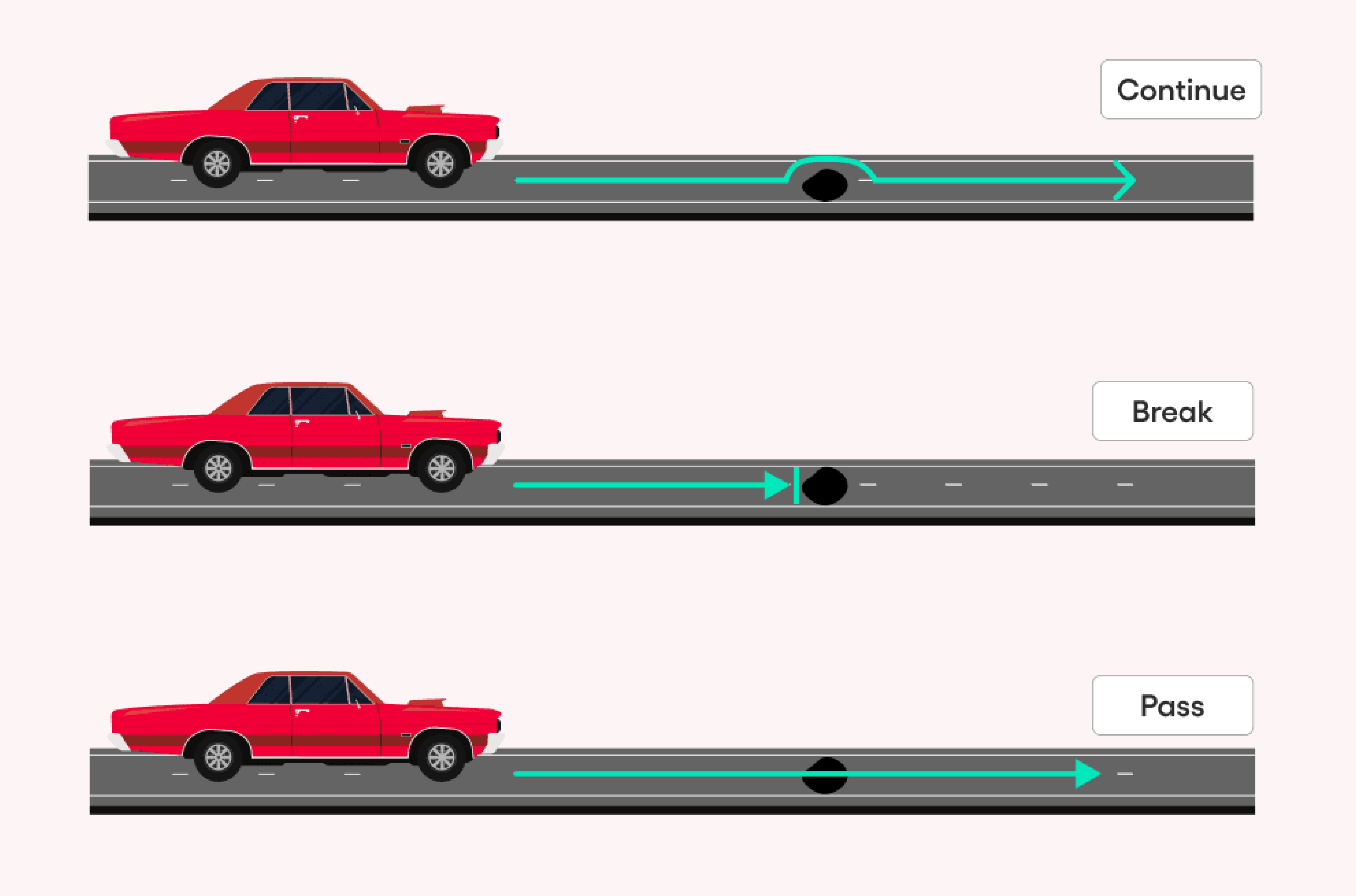
Python supports several types of loops:

- **for loop**: Iterates over a sequence (e.g., list, tuple, string, dictionary).
- **while loop:** Executes a block of code as long as a specified condition is true.
- **Nested loops:** One loop inside another loop.
- Comprehensions (e.g., list comprehensions, dictionary comprehensions): Concise ways to create lists, dictionaries, etc., using loops.

break and continue …

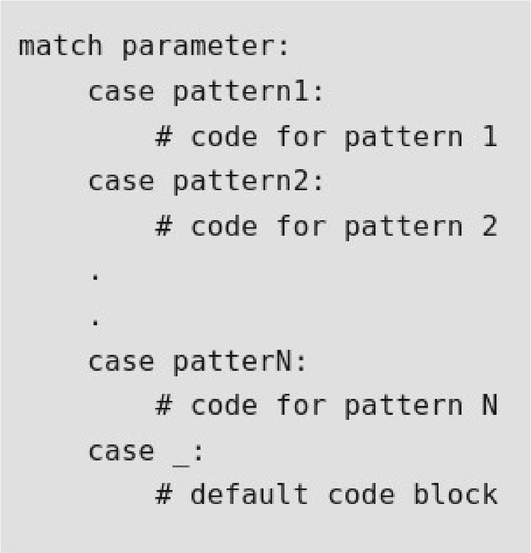


break and continue ...

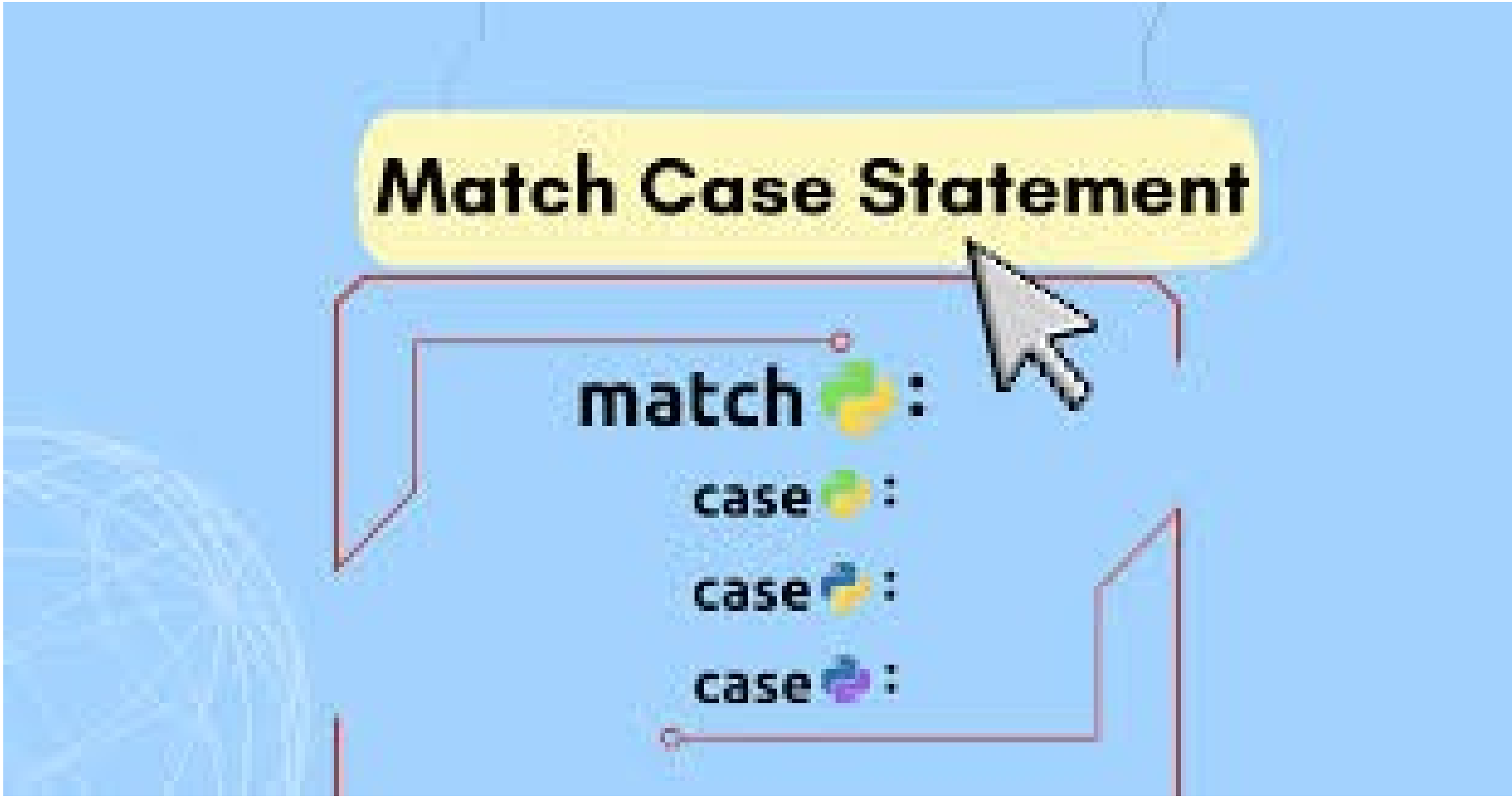


Match Case

- Match Case is the Switch Case of Python which was introduced in Python 3.10.
- The match case statement in Python is initialized with the match keyword followed by the parameter to be matched.
- Various cases are defined using the case keyword and the pattern to match the parameter.
- The “_” is the wildcard character that runs when all the cases fail to match the parameter value.



```
match parameter:
    case pattern1:
        # code for pattern 1
    case pattern2:
        # code for pattern 2
    .
    .
    case patterN:
        # code for pattern N
    case _:
        # default code block
```



Match Case Statement

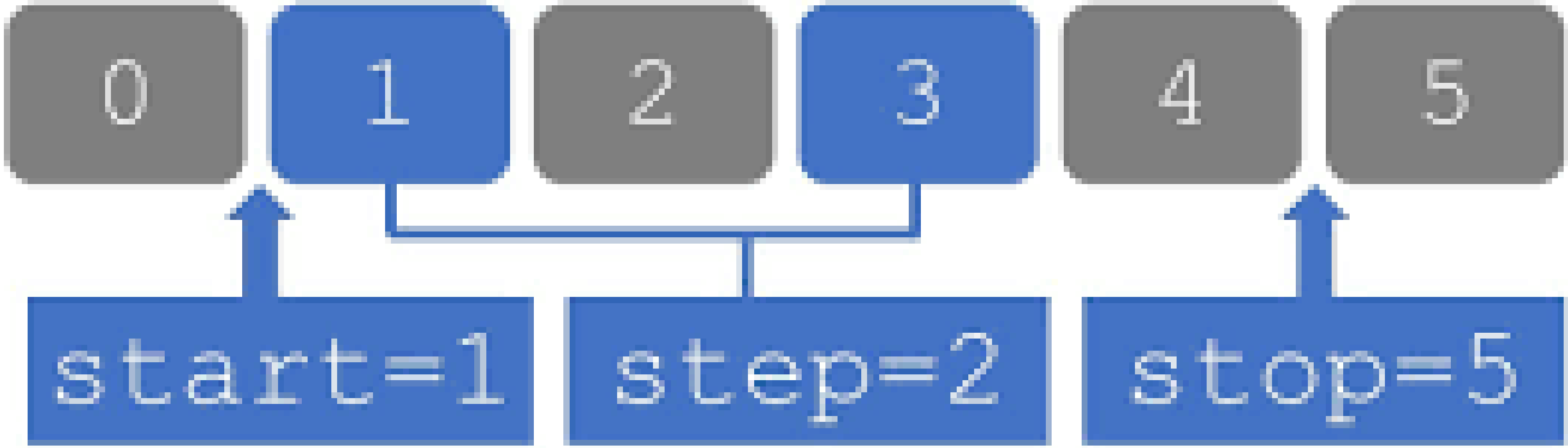
```
match :
    case :
    case :
    case :
```

Range Function

- The range() function returns a sequence of numbers, starting from 0 by default, and increments by 1 (by default), and stops before a specified number.

- Syntax: range(start, stop, step)
  - start: Optional. An integer number specifying at which position to start. Default is 0
  - stop: Required. An integer number specifying at which position to stop (not included).
  - step: Optional. An integer number specifying the incrementation. Default is 1

Range Start + Stop + Step



0 1 2 3 4 5

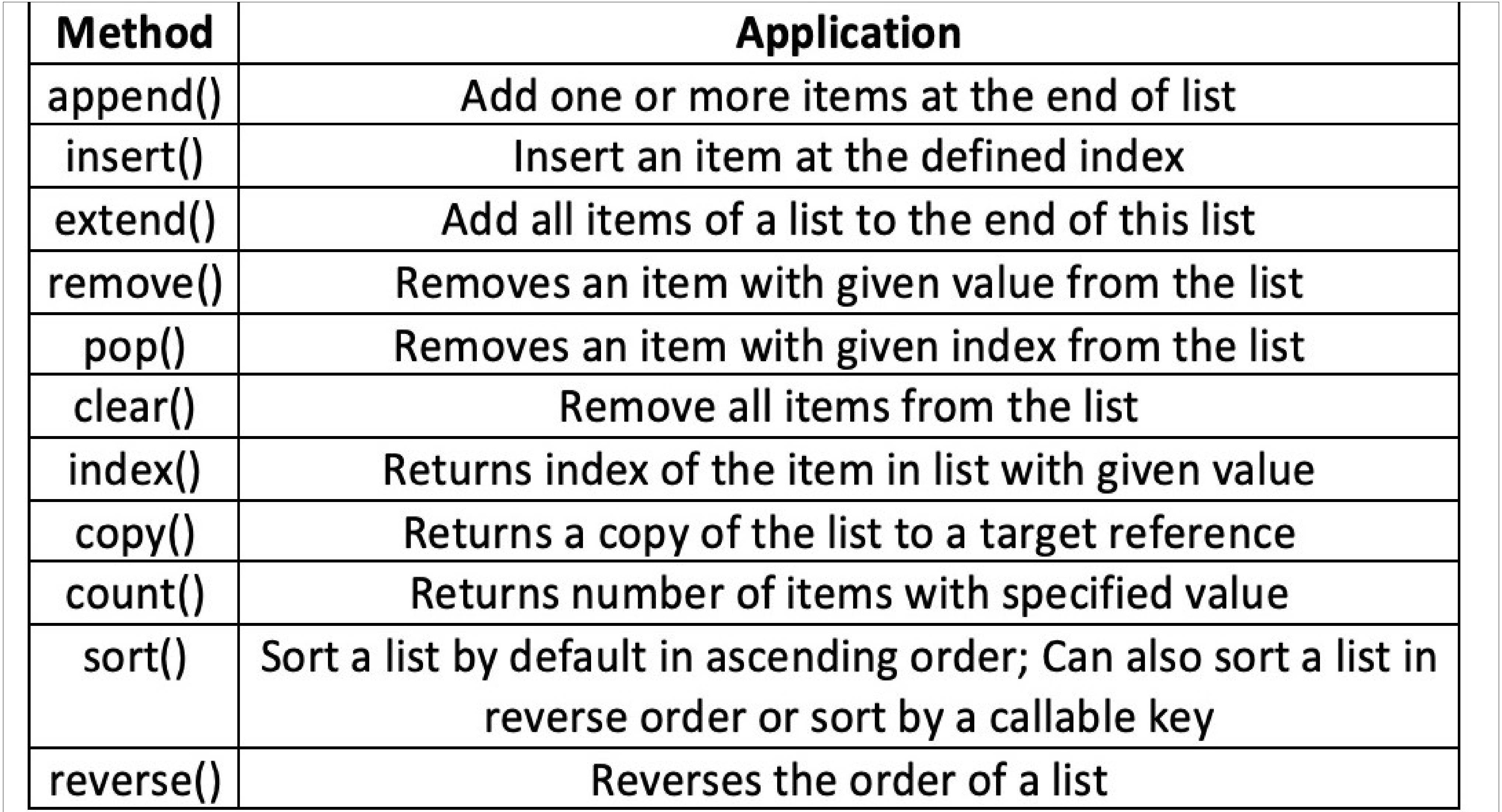
start=1

step=2

stop=5

Lists

- Lists are used to store multiple items in a single variable.
- Lists are built-in data types in Python used to store collections of data.
- Syntax:
  - list_name = [“item 1”,”item 2”, … , “item n”]
- List items are ordered, changeable, and allow duplicate values.
- List items are indexed, the first item has index [0], the second item has index [1] etc.
- The items have a defined order, and that order will not change.
- If you add new items to a list, the new items will be placed at the end of the list.
- Lists can have items with same value



Method	Application
append()	Add one or more items at the end of list
insert()	Insert an item at the defined index
extend()	Add all items of a list to the end of this list
remove()	Removes an item with given value from the list
pop()	Removes an item with given index from the list
clear()	Remove all items from the list
index()	Returns index of the item in list with given value
copy()	Returns a copy of the list to a target reference
count()	Returns number of items with specified value
sort()	Sort a list by default in ascending order; Can also sort a list in reverse order or sort by a callable key
reverse()	Reverses the order of a list

Tuple

- Tuples are used to store multiple items in a single variable.
- Tuple is built-in data types in Python
- Tuples are written with round brackets.
- Eg: courses = (“imca”, “imscit”, “bca”)
- Tuple items are indexed, the first item has index [0], the second item has index [1] etc.
- Tuples are sequences of objects.
- Tuples are also called containers or collections because a single tuple can contain or collect an arbitrary number of other objects.

Characteristics of Tuple Objects

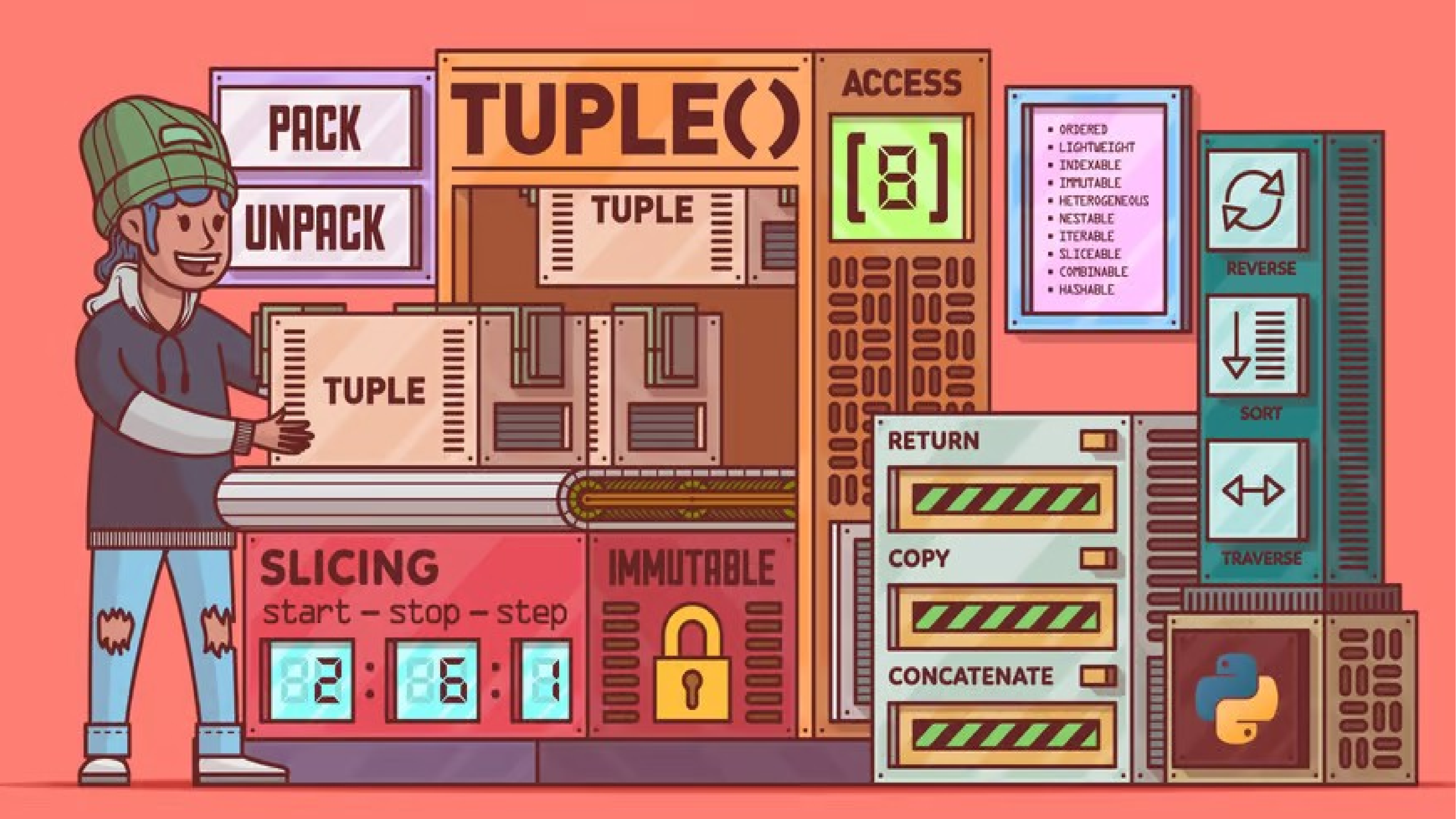
- **Ordered**: They contain elements that are sequentially arranged according to their specific insertion order.
- **Lightweight**: They consume relatively small amounts of memory compared to other sequences like lists.
- **Indexable** through a zero-based index: They allow you to access their elements by integer indices that start from zero.
- **Immutable**: They don't support in-place mutations or changes to their contained elements. They don't support growing or shrinking operations.
- **Heterogeneous**: They can store objects of different data types and domains, including mutable objects.

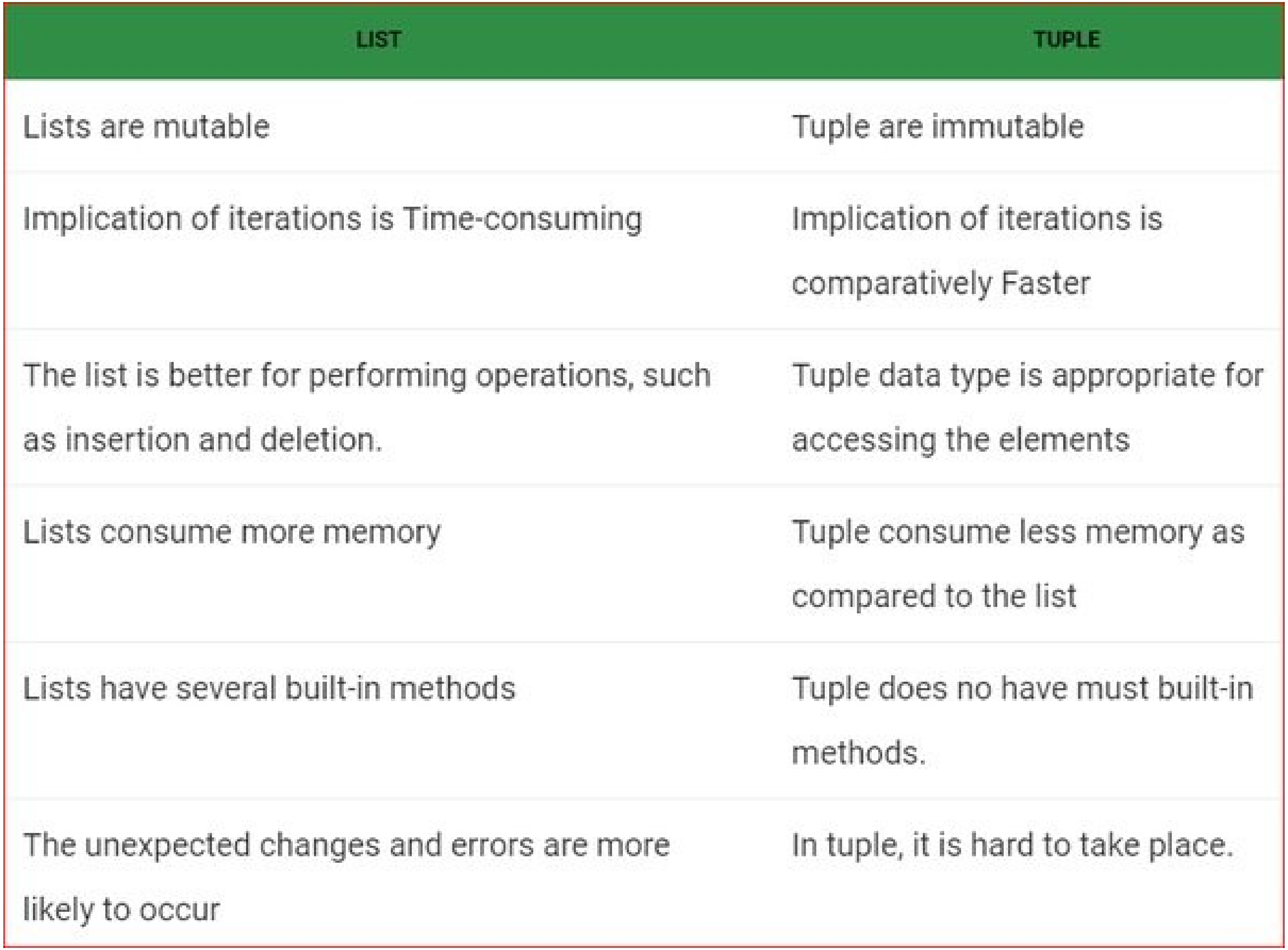
Characteristics of Tuple Objects

- **Nestable**: They can contain other tuples, so you can have tuples of tuples.
- **Iterable**: They support iteration, so you can traverse them using a loop or comprehension while you perform operations with each of their elements.
- **Sliceable**: They support slicing operations, meaning that you can extract a series of elements from a tuple.
- **Combinable**: They support concatenation operations, so you can combine two or more tuples using the concatenation operators, which creates a new tuple.
- **Hashable**: They can work as keys in dictionaries when all the tuple items are immutable.

Methods of Tuple Objects

Function	Description
all()	Return True if all elements of the tuple are true (or if the tuple is empty).
any()	Return True if any element of the tuple is true. If the tuple is empty, return False.
enumerate()	Return an enumerate object. It contains the index and value of all the items of tuple as pairs.
len()	Return the length (the number of items) in the tuple.
max()	Return the largest item in the tuple.
min()	Return the smallest item in the tuple
sorted()	Take elements in the tuple and return a new sorted list (does not sort the tuple itself).
sum()	Retrun the sum of all elements in the tuple.
tuple()	Convert an iterable (list, string, set, dictionary) to a tuple.


PACK
UNPACK
TUPLE()
TUPLE
TUPLE
ACCESS
[8]
• ORDERED
• LIGHTWEIGHT
• INDEXABLE
• IMMUTABLE
• HETEROGENEOUS
• NESTABLE
• ITERABLE
• SLICEABLE
• COMBINABLE
• HASHABLE
REVERSE
SORT
TRAVERSE
RETURN
COPY
CONCATENATE
SLICING
start – stop – step
02 : 06 : 1
IMMUTABLE



LIST	TUPLE
Lists are mutable	Tuple are immutable
Implication of iterations is Time-consuming	Implication of iterations is comparatively Faster
The list is better for performing operations, such as insertion and deletion.	Tuple data type is appropriate for accessing the elements
Lists consume more memory	Tuple consume less memory as compared to the list
Lists have several built-in methods	Tuple does no have must built-in methods.
The unexpected changes and errors are more likely to occur	In tuple, it is hard to take place.

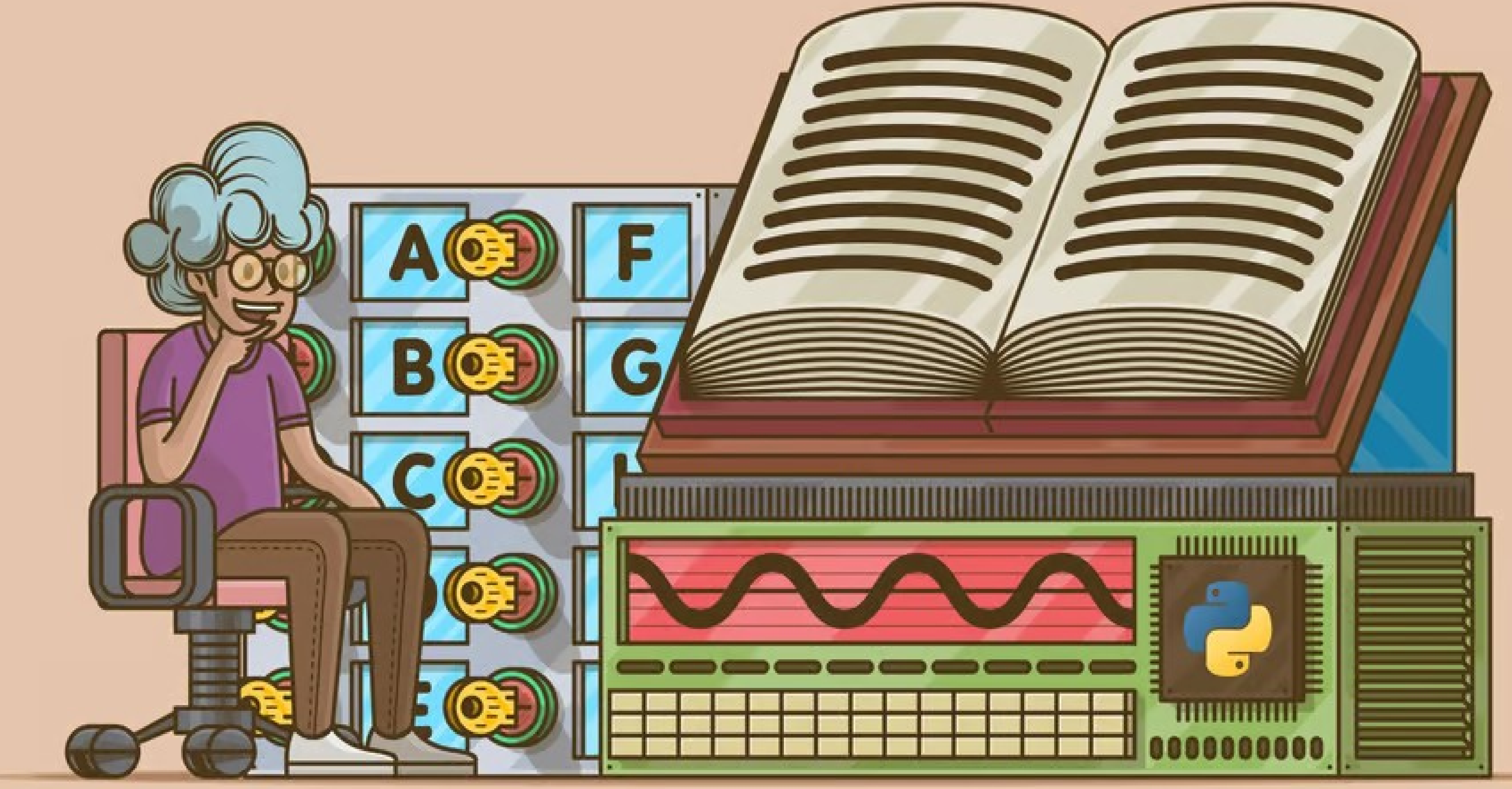
Dictionaries

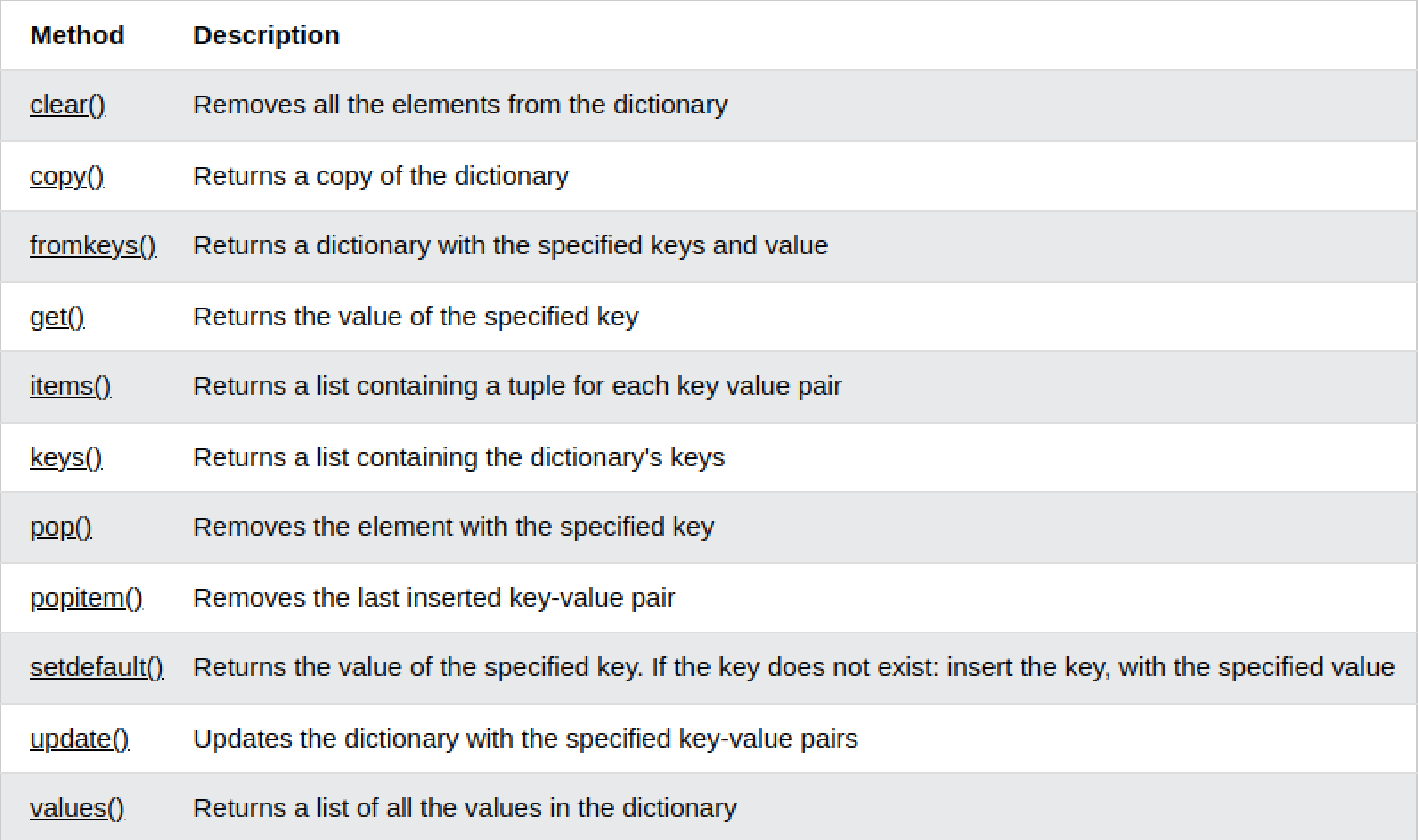
- Dictionaries are used to store data values in key:value pairs.
- A dictionary is a collection which is ordered, changeable and do not allow duplicates.
- Dictionary items are presented in **“key:value”** pairs, and can be referred to by using the key names.
- An empty dictionary can be created by just placing curly braces{}.
- Python Dictionary Syntax

  ```
  dict_var = {key1 : value1, key2 : value2, …..}
  ```

Dictionaries

- Values in a dictionary can be of any data type and can be duplicated, whereas keys can’t be repeated and must be immutable.
- Dictionary keys are case sensitive
- Dictionaries are a useful data structure for storing data in Python because they are capable of imitating real-world data arrangements where a certain value exists for a given key.


A
F
B
G
C



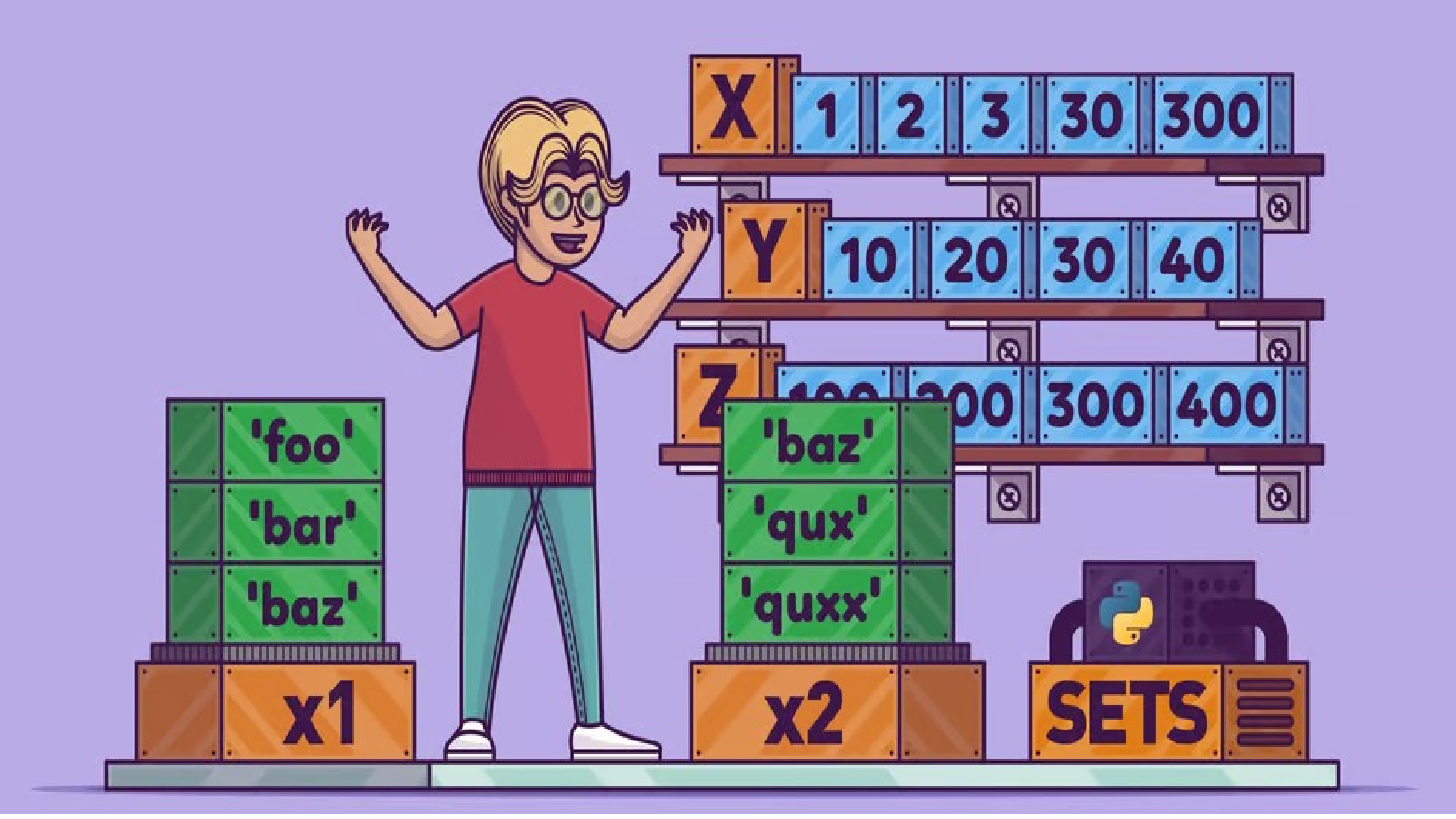
Method	Description
clear()	Removes all the elements from the dictionary
copy()	Returns a copy of the dictionary
fromkeys()	Returns a dictionary with the specified keys and value
get()	Returns the value of the specified key
items()	Returns a list containing a tuple for each key value pair
keys()	Returns a list containing the dictionary's keys
pop()	Removes the element with the specified key
popitem()	Removes the last inserted key-value pair
setdefault()	Returns the value of the specified key. If the key does not exist: insert the key, with the specified value
update()	Updates the dictionary with the specified key-value pairs
values()	Returns a list of all the values in the dictionary

Set and Frozenset: Set

- Sets are used to store multiple items in a single variable.
- Set is one of the built-in data types in Python.
- A set is a collection which is unordered, unchangeable, and unindexed.
- Set items are unchangeable, but you can remove items and add new items.
- Duplicates are not allowed.
- ***The values "True" and 1 are considered the same value in sets, and are treated as duplicates***
- ***False and 0 is considered the same value:***

Set

- Sets are unordered that means it is not sure in which order the items will appear.
- A set cannot have mutable elements like lists, sets or dictionaries as its elements.
- Creating an empty set is a bit tricky because Empty curly braces {} will make an empty dictionary in Python.
- To make a set without any elements, set() function is used without any argument.


X
1
2
3
30
300
Y
10
20
30
40
Z
300
400
'foo'
'bar'
'baz'
x1
'baz'
'qux'
'quxx'
x2
SETS

Method	Description
add()	Adds an element to the set
clear()	Removes all the elements from the set
copy()	Returns a copy of the set
difference()	Returns a set containing the difference between two or more sets
difference_update()	Removes the items in this set that are also included in another, specified set
discard()	Remove the specified item
intersection()	Returns a set, that is the intersection of two or more sets
intersection_update()	Removes the items in this set that are not present in other, specified set(s)
isdisjoint()	Returns whether two sets have a intersection or not
issubset()	Returns whether another set contains this set or not
issuperset()	Returns whether this set contains another set or not
pop()	Removes an element from the set
remove()	Removes the specified element
symmetric_difference()	Returns a set with the symmetric differences of two sets
symmetric_difference_update()	inserts the symmetric differences from this set and another
union()	Return a set containing the union of sets
update()	Update the set with another set, or any other iterable

Frozenset

- A variation of sets that provides a version of sets with immutability is Frozen Sets.
- Frozen Sets are created through the method frozensets() which takes iterable data and converts them into a frozen set.
- The term "Iterable Data" means sequential data that can be iterated upon.
- Frozenset is a data type in python that is immutable and unordered. It is used when an unchangeable set is required, such as an element of another set or as a key in a dictionary.