List

A list in Python is used to store multiple items in a single variable. Lists are ordered, mutable, and allow duplicate values.

List Characteristics

- Ordered: Items have a defined order.
- Mutable: Can be changed after creation.
- Allows Duplicates: Same values can appear multiple times.
- Supports Different Data Types: Strings, integers, and even other lists.

List Methods

Here is a complete list of methods available for lists in Python:

1. append(): Adds an element to the end of the list.
2. extend(): Adds elements from an iterable to the list.
3. insert(): Inserts an element at a specified index.
4. remove(): Removes the first occurrence of an element.
5. pop(): Removes and returns an element at a given index (defaults to last).
6. clear(): Removes all elements from the list.
7. index(): Returns the index of the first occurrence of an element.
8. count(): Returns the number of occurrences of an element.
9. sort(): Sorts the list in place (ascending order by default)
10. reverse(): Reverses the list in place.
11. copy(): Returns a *shallow* copy of the list.

In [8]:
```
my_list = ["apple", "banana", "cherry"]
print(my_list)
print(type(my_list))
```

```
['apple', 'banana', 'cherry']
<class 'list'>
```

In [9]:
```
# Accessing List Elements
print(my_list[0])  # Output: apple
print(my_list[1])  # Output: banana
print(my_list[2])  # Output: cherry
print(my_list[-1])  # Output: cherry
print(my_list[-2])  # Output: banana
print(my_list[-3])  # Output: apple
```

```
apple
banana
cherry
cherry
banana
apple
```

In [10]:
```
# Slicing
print(my_list[1:3])  # Output: ['banana', 'cherry']
print(my_list[:2])  # Output: ['apple', 'banana']
print(my_list[1:])  # Output: ['banana', 'cherry']

['banana', 'cherry']
['apple', 'banana']
['banana', 'cherry']
```

In [11]:
```
# Changing an Item
print(my_list)
my_list[1] = "orange"
print(my_list)
```

```
['apple', 'banana', 'cherry']
['apple', 'orange', 'cherry']
```

In [12]:
```
# 1. append(): Adds an element to the end of the list.
colors = ["red", "blue"]
colors.append("green")
print(colors)  # ['red', 'blue', 'green']
```

```
['red', 'blue', 'green']
```

In [13]:
```
# 2. extend(): Adds elements from another list (or iterable).
colors = ["red", "blue"]
colors.extend(["yellow", "purple"])
print(colors)  # ['red', 'blue', 'yellow', 'purple']
```

```
['red', 'blue', 'yellow', 'purple']
```

In [15]:
```
# 3. insert(): Inserts an element at a specific index.

colors = ["purple", "magenta"]
colors.insert(1, "green")
print(colors)  # ['purple', 'green', 'magenta']
```

```
['purple', 'green', 'magenta']
```

In [16]:
```
# 4. remove(): Removes the first occurrence of a value.
colors = ["red", "blue", "red"]
colors.remove("red")
print(colors)  # ['blue', 'red']
```

```
['blue', 'red']
```

In [17]:
```
# 5. pop(): Removes and returns item at index (default is last).
colors = ["red", "blue", "green"]
color = colors.pop()
print(color)  # green
print(colors)  # ['red', 'blue']

green
['red', 'blue']
```

In [18]:
```
# 6. clear(): Removes all elements from the list.
colors = ["red", "blue"]
colors.clear()
print(colors)  # []
```

```
[]
```

In [19]:
```
# 7. index(): Returns the index of the first matching value.
colors = ["red", "blue", "green"]
i = colors.index("blue")
print(i)  # 1
```

```
1
```

In [20]:
```
# 8. count(): Counts how many times a value appears.
colors = ["red", "blue", "red"]
print(colors.count("red"))  # 2
```

```
2
```

In [22]:
```
# 9. sort(): Sorts the list in place (ascending by default).

numbers = [5, 2, 9, 1]
numbers.sort()
print(numbers)  # [1, 2, 5, 9]

# Descending sort
numbers.sort(reverse=True)
print(numbers)  # [9, 5, 2, 1]
```

```
[1, 2, 5, 9]
[9, 5, 2, 1]
```

In [23]:
```
# 10. reverse(): Reverses the list in place.
colors = ["red", "blue", "green"]
colors.reverse()
print(colors)  # ['green', 'blue', 'red']
```

```
['green', 'blue', 'red']
```

In [24]:
```
# 11. copy(): Returns a shallow copy of the list.
colors = ["red", "blue"]
new_colors = colors.copy()
print(new_colors)  # ['red', 'blue']
```

```
['red', 'blue']
```

In [ ]:
```
# Looping Through a List
new_list = ["blue", "green", "red", "orange"]
for item in new_list:
    print(item)
```

```
blue
green
red
orange
```

In [ ]:
```
# Joining Lists
list1 = [1, 2, 3]
list2 = (4, 5, 6)
combined = list1 + list2  # [1, 2, 3, 4, 5, 6]
print(combined)
```

```
[1, 2, 3, 4, 5, 6]
```

In [ ]:
```
list1 = ["a", "b", "c", "d", "e", "f", "g", "h"]
print(list1)
print(type(list1))
```

```
['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h']
<class 'list'>
```

Tuple

A tuple in Python is an immutable ordered collection of elements, which means once created, you cannot modify, add, or remove elements from it.

Key characteristics of a tuple:

1. Ordered: The elements in a tuple have a specific order and can be accessed by their index.
2. Immutable: After creating a tuple, you cannot change its elements, unlike lists, which are mutable.
3. Heterogeneous: A tuple can store elements of different data types (e.g., integers, strings, lists, etc.).
4. Parentheses: Tuples are defined by enclosing the elements in parentheses ().
5. Indexing and Slicing: Just like lists, you can access elements in a tuple using indexing and slicing.

In [30]:
```
cities = ("New York", "Paris", "Tokyo")
print(cities)
print(type(cities))
```

```
('New York', 'Paris', 'Tokyo')
<class 'tuple'>
```

In [28]:
```
# Tuple Items - Indexed
colors = ("red", "blue", "green")
print(colors[0])  # Output: red
print(colors[1])  # Output: blue
print(colors[2])  # Output: green
print(colors[-1])  # Output: green
print(colors[-2])  # Output: blue
print(colors[-3])  # Output: red

red
blue
green
green
blue
red
```

In [29]:
```
# Tuple Slicing
languages = ("Python", "Java", "C++", "JavaScript", "Swift")

print(languages[1:4])  # Output: ('Java', 'C++', 'JavaScript')

print(languages[0:])  # Output: ('Python', 'Java', 'C++', 'JavaScript', 'Swift')


print(languages[:4])  # Output: ('Python', 'Java', 'C++', 'JavaScript')


print(languages[0:5])  # Output: ('Python', 'Java', 'C++', 'JavaScript', 'Swift')

print(languages[-3:-1])  # Output: ('C++', 'JavaScript')
```

```
('Java', 'C++', 'JavaScript')
('Python', 'Java', 'C++', 'JavaScript', 'Swift')
('Python', 'Java', 'C++', 'JavaScript')
('Python', 'Java', 'C++', 'JavaScript', 'Swift')
('C++', 'JavaScript')
```

In [32]:
```
# Checking if Item Exists
sports = ("soccer", "basketball", "tennis")
if "tennis" in sports:
    print("Yes, tennis is in the tuple!")
```

```
Yes, tennis is in the tuple!
```

In [33]:
```
# Tuple Length
animals = ("lion", "tiger", "elephant", "giraffe")
print(len(animals))  # Output: 4
```

```
4
```

In [34]:
```
# Tuple with One Item
single_item = ("hello",)
print(type(single_item))  # Output: <class 'tuple'>
```

```
<class 'tuple'>
```

In [35]:
```
not_a_tuple = ("hello")  # Missing comma
print(type(not_a_tuple))  # Output: <class 'str'>
```

```
<class 'str'>
```

In [36]:
```
# Tuple with Different Data Types
person = ("Alice", 25, True, 5.7)
print(person)
```

```
('Alice', 25, True, 5.7)
```

In [40]:
```
# Changing Tuple Values (Convert to List)
# We can't directly change tuple so first we need to convert to list and change it and then we convert into tuple.
fruits = ("apple", "banana", "cherry")
fruit_list = list(fruits)
fruit_list[1] = "orange"
fruits = tuple(fruit_list)
print(fruits)
```

```
('apple', 'orange', 'cherry')
```

In [41]:
```
# Adding Items to a Tuple
vehicles = ("car", "bike", "bus")
extra = ("train",)
vehicles += extra

print(vehicles)  # Output: ('car', 'bike', 'bus', 'train')
```

```
('car', 'bike', 'bus', 'train')
```

In [43]:
```
# Removing Items from a Tuple
gadgets = ("laptop", "phone", "tablet")
gadget_list = list(gadgets)
gadget_list.remove("phone")
gadgets = tuple(gadget_list)
print(gadgets)  # Output: ('laptop', 'tablet')
```

```
('laptop', 'tablet')
```

In [45]:
```
# Deleting a Tuple
books = ("novel", "magazine", "comics")
del books
# print(books)  # This will cause an error because 'books' is deleted
```

In [47]:
```
# Loop Through a Tuple
countries = ("USA", "UK", "Canada")
for country in countries:
    print(country)
```

```
USA
UK
Canada
```

In [48]:
```
# Joining Tuples
tuple1 = ("a", "b", "c")
tuple2 = (100, 200, 300)

tuple3 = tuple1 + tuple2
print(tuple3)  # Output: ('a', 'b', 'c', 100, 200, 300)
```

```
('a', 'b', 'c', 100, 200, 300)
```

In [49]:
```
# Multiplying Tuples
shapes = ("circle", "square")
new_tuple = shapes * 3

print(new_tuple)  # Output: ('circle', 'square', 'circle', 'square', 'circle', 'square')
```

```
('circle', 'square', 'circle', 'square', 'circle', 'square')
```

Tuple Methods

1. count(): Returns the number of times a specified value occurs in a tuple

2. index(): Searches the tuple for a specified value and returns the position of where it was found

In [50]:
```
# count() returns how many times a value appears in the tuple.
colors = ("red", "blue", "green", "red", "yellow", "red")
red_count = colors.count("red")
print("Count of 'red':", red_count)  # Output: 3
```

```
Count of 'red': 3
```

In [51]:
```
# index() returns the first index where the value appears.
colors = ("red", "blue", "green", "yellow")
first_red_index = colors.index("red")
print("Index of first 'red':", first_red_index)  # Output: 0
```

```
Index of first 'red': 0
```

In [ ]:
```

```