🧠 What is a List?

A **list** in Python is an **ordered, mutable (changeable)** collection of elements.
Lists are written inside **square brackets `[ ]`**.

```
my_list = [10, 20, 30, "apple", True]
```

⚙️ 1. Creating Lists

```
list1 = [1, 2, 3, 4]
list2 = ["apple", "banana", "cherry"]
list3 = [1, "hello", 3.5, True]
empty_list = []           # Empty list
nested_list = [1, [2, 3], 4]
```

📥 2. Accessing Elements (Indexing)

```
my_list = [10, 20, 30, 40]
print(my_list[0])    # 10
print(my_list[-1])   # 40 (last element)
```

✂️ 3. Slicing

```
my_list = [10, 20, 30, 40, 50]
print(my_list[1:4])   # [20, 30, 40]
print(my_list[:3])    # [10, 20, 30]
print(my_list[::2])   # [10, 30, 50]
```

✏️ 4. Changing or Updating Elements

```
my_list = [10, 20, 30]
my_list[1] = 200
print(my_list)   # [10, 200, 30]
```

➕ 5. Adding Elements

a) `append()` → Add at the end

```
fruits = ["apple", "banana"]
fruits.append("cherry")
print(fruits)  # ['apple', 'banana', 'cherry']
```

b) `insert()` → Add at specific index

```
fruits = ["apple", "banana"]
fruits.insert(1, "orange")
print(fruits)  # ['apple', 'orange', 'banana']
```

c) `extend()` → Add multiple elements

```
fruits = ["apple", "banana"]
fruits.extend(["grape", "mango"])
print(fruits)  # ['apple', 'banana', 'grape', 'mango']
```

➖ 6. Removing Elements

a) `remove()` → Remove by value

```
fruits = ["apple", "banana", "cherry"]
fruits.remove("banana")
print(fruits)  # ['apple', 'cherry']
```

b) `pop()` → Remove by index (default last)

```
fruits = ["apple", "banana", "cherry"]
fruits.pop(1)
print(fruits)  # ['apple', 'cherry']
```

c) `del` → Delete using keyword

```
fruits = ["apple", "banana", "cherry"]
del fruits[0]
print(fruits)  # ['banana', 'cherry']
```

d) `clear()` → Remove all elements

```
fruits = ["apple", "banana"]
fruits.clear()
print(fruits)  # []
```

🧩 7. List Methods (Detailed)

Method	Description	Example
append(x)	Add item `x` at end	`list.append(5)`
insert(i, x)	Add `x` at position `i`	`list.insert(1, "apple")`
extend(iterable)	Add multiple items	`list.extend([1,2,3])`
remove(x)	Remove first occurrence of `x`	`list.remove("apple")`
pop([i])	Remove item at index `i` (or last)	`list.pop(2)`
clear()	Remove all items	`list.clear()`
index(x)	Return index of `x`	`list.index(20)`
count(x)	Count occurrences of `x`	`list.count(10)`

Method	Description	Example
sort()	Sort list ascending	`list.sort()`
reverse()	Reverse list order	`list.reverse()`
copy()	Return shallow copy	`new_list = list.copy()`

🧮 8. Mathematical & Logical Operations

a) Concatenation

```
a = [1, 2, 3]
b = [4, 5]
print(a + b)    # [1, 2, 3, 4, 5]
```

b) Repetition

```
a = [1, 2]
print(a * 3)    # [1, 2, 1, 2, 1, 2]
```

c) Membership

```
a = [10, 20, 30]
print(20 in a)        # True
print(50 not in a)    # True
```

🔁 9. Iterating Through a List

```
fruits = ["apple", "banana", "cherry"]
for fruit in fruits:
    print(fruit)
```

🧮 10. Built-in Functions with Lists

```
nums = [5, 2, 9, 1]
print(len(nums))   # 4
print(min(nums))   # 1
print(max(nums))   # 9
print(sum(nums))   # 17
```

🧰 11. Copying Lists (Important for Beginners)

```
a = [1, 2, 3]
b = a.copy()   # Creates new copy
b.append(4)
print(a)  # [1, 2, 3]
print(b)  # [1, 2, 3, 4]
```

🔄 12. Nested Lists

```
matrix = [[1, 2, 3], [4, 5, 6]]
print(matrix[0][1])  # 2
```

🧠 13. List Comprehension (Short Way to Create Lists)

```
# Create list of squares
squares = [x**2 for x in range(5)]
print(squares)  # [0, 1, 4, 9, 16]
```

✅ Summary Table

Operation	Example	Result
Append	`list.append(5)`	Add element
Insert	`list.insert(1, x)`	Add at position
Extend	`list.extend([x,y])`	Add multiple
Remove	`list.remove(x)`	Delete by value
Pop	`list.pop()`	Delete last
Clear	`list.clear()`	Empty list
Index	`list.index(x)`	Find position
Count	`list.count(x)`	Count items
Sort	`list.sort()`	Arrange ascending
Reverse	`list.reverse()`	Reverse order
Copy	`list.copy()`	Duplicate