**📚 What is a List in Python?**

A **list** is a **mutable**, **ordered**, **indexed** collection that allows **duplicate values**. It can contain elements of **different data types**.

my\_list = [10, "hello", 3.14, True]

**🧠 Key Properties**

| **Property** | **Description** |
| --- | --- |
| Ordered | Items are stored in a defined order (insertion order) |
| Mutable | Can change elements after creation |
| Heterogeneous | Can store mixed data types |
| Duplicates | Allows repeated elements |

**🧩 Indexing and Slicing**

**🔢 Indexing**

lst = [1, 2, 3, 4, 5]

print(lst[0]) # 1

print(lst[-1]) # 5

**✂️ Slicing**

print(lst[1:4]) # [2, 3, 4]

print(lst[:3]) # [1, 2, 3]

print(lst[::-1]) # [5, 4, 3, 2, 1] (reverse)

**🛠️ Built-in List Methods**

**📌 Adding Items**

lst = [1, 2, 3]

lst.append(4) # [1, 2, 3, 4]

lst.insert(1, 10) # [1, 10, 2, 3, 4]

lst.extend([5, 6]) # [1, 10, 2, 3, 4, 5, 6]

**📌 Removing Items**

lst.remove(10) # Removes first occurrence of 10

lst.pop() # Removes last element

lst.pop(1) # Removes element at index 1

lst.clear() # Removes all elements

**📌 Searching**

lst = [1, 2, 3, 2, 4]

print(lst.index(2)) # 1 (first occurrence)

print(lst.count(2)) # 2

**📌 Sorting and Reversing**

nums = [4, 2, 9, 1]

nums.sort() # [1, 2, 4, 9]

nums.sort(reverse=True) # [9, 4, 2, 1]

nums.reverse() # Reverse the list in place

**📌 Copying**

a = [1, 2, 3]

b = a.copy()

print(b) # [1, 2, 3]

**📌 Other Useful Methods**

lst = [1, 2, 3]

print(len(lst)) # 3

print(max(lst)) # 3

print(min(lst)) # 1

print(sum(lst)) # 6

**🔧 List Operations**

| **Operation** | **Syntax** | **Example** |
| --- | --- | --- |
| Concatenation | + | [1, 2] + [3, 4] → [1, 2, 3, 4] |
| Repetition | \* | [1] \* 3 → [1, 1, 1] |
| Membership | in, not in | 2 in [1,2,3] → True |
| Iteration | for x in list: | See below |

for item in [1, 2, 3]:

print(item)

**🧪 List Comprehensions**

Powerful shorthand for creating new lists.

squares = [x\*x for x in range(5)] # [0, 1, 4, 9, 16]

evens = [x for x in range(10) if x%2==0] # [0, 2, 4, 6, 8]

**🔄 Nested Lists**

matrix = [[1, 2], [3, 4]]

print(matrix[0][1]) # 2

**💥 Mutable Nature**

Lists can be changed in place:

lst = [1, 2, 3]

lst[0] = 10

print(lst) # [10, 2, 3]

**🧠 Interview Questions (With Answers)**

**✅ Basic-Level**

1. **What is a list in Python?**
   * A mutable, ordered collection of elements.
2. **How are lists different from tuples?**
   * Lists are mutable, tuples are not.
3. **How do you reverse a list?**
   * lst[::-1] or lst.reverse()
4. **How to remove duplicates from a list?**
   * list(set(lst)) (Unordered)

**✅ Intermediate-Level**

1. **Difference between append() and extend()?**
   * append() adds a single item; extend() merges another iterable.
2. **What happens if you use list.remove(x) and x is not found?**
   * It throws a ValueError.
3. **How do you sort a list without modifying the original?**
   * sorted(lst)
4. **How to flatten a nested list?**

nested = [[1, 2], [3, 4]]

flat = [item for sublist in nested for item in sublist] # [1, 2, 3, 4]

**✅ Advanced-Level**

1. **How does list slicing work internally (memory)?**
   * It creates a new list (copy), not a view.
2. **How to find common elements between two lists?**

a = [1, 2, 3]

b = [2, 3, 4]

common = list(set(a) & set(b)) # [2, 3]

1. **How to remove all occurrences of a value?**

lst = [1, 2, 2, 3]

lst = [x for x in lst if x != 2]

1. **List vs Array vs Set vs Tuple – Differences?**

* Arrays (from numpy): fixed-type
* Sets: unordered, unique
* Tuples: immutable
* Lists: mutable, ordered

**📘 Practice Problems**

1. Reverse a list without using reverse() or slicing.
2. Find all even numbers from a list.
3. Merge two sorted lists into one sorted list.
4. Write a program to find frequency of elements in a list.

**✅ Summary**

* Lists are core to Python — used in almost every data structure & algorithm problem.
* Know the **difference between append, insert, extend, remove, pop**.
* Understand **list comprehension** – a favorite in interviews.
* Be able to **manipulate nested lists**, especially in real-world data parsing.