Python Basics: Common Data Type - List and List Methods

Section 1: Learn

What is a List?

A **list** is a built-in data type in Python that is used to **store multiple values** in a single variable. It is like a **collection of items**.

Why Use Lists?

- To store multiple items together.
- To avoid creating many variables.
- You can add, remove, and change values easily.

How to Create a List?

```
fruits = ["apple", "banana", "mango"]

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

mixed = ["hello", 10, 2.5]
```

- Lists are created using **square brackets** `[]`.
- Items are separated by **commas**.

Interesting Fact:

In the early 2000s, Indian software companies working on billing systems used lists to store item names, prices, and customer data to print invoices. Lists made it easy to handle such collections.

Section 2: Practice

1. Creating and Printing a List

```
names = ["Amit", "Rahul", "Priya"]

print(names)
```

Output:

```
['Amit', 'Rahul', 'Priya']
```

---

2. Accessing Items in a List

```
print(names[0])   # First item

print(names[2])   # Third item
```

Output:

```
Amit

Priya
```

---

3. Changing Item in a List

```
names[1] = "Rohit"

print(names)
```

Output:

```
['Amit', 'Rohit', 'Priya']
```

---

4. Common List Methods

a) append() - Add item at end

```
names.append("Kiran")

print(names)
```

Output:

```
['Amit', 'Rohit', 'Priya', 'Kiran']
```

b) insert() - Add item at specific position

```
names.insert(1, "Sita")

print(names)
```

Output:

```
['Amit', 'Sita', 'Rohit', 'Priya', 'Kiran']
```

c) remove() - Remove specific item

```
names.remove("Rohit")

print(names)
```

Output:

```
['Amit', 'Sita', 'Priya', 'Kiran']
```

d) pop() - Remove item from end (or given index)

```
names.pop()

print(names)
```

Output:

```
['Amit', 'Sita', 'Priya']
```

e) len() - Find number of items

```
print(len(names))
```

Output:

```
3
```

f) sort() - Sort the list in ascending order

```
numbers = [4, 2, 9, 1]

numbers.sort()

print(numbers)
```

Output:

```
[1, 2, 4, 9]
```

g) reverse() - Reverse the list

```
numbers.reverse()

print(numbers)
```

Output:

```
[9, 4, 2, 1]
```

h) index() - Find the index of an item

```
print(names.index("Priya"))
```

Output:

```
2
```

i) count() - Count how many times an item appears

```
marks = [70, 80, 70, 90]

print(marks.count(70))
```

Output:

```
2
```

j) clear() - Remove all items from the list

```
names.clear()

print(names)
```

Output:

```
[]
```

k) copy() - Make a copy of a list

```
original = ["pen", "pencil"]

copy_list = original.copy()

print(copy_list)
```

Output:

```
['pen', 'pencil']
```

l) extend() - Add elements of another list

```
list1 = [1, 2]

list2 = [3, 4]

list1.extend(list2)

print(list1)
```

Output:

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

Real-Life Example: Grocery List

```
grocery = ["rice", "wheat", "dal"]

grocery.append("sugar")

print("Grocery List:", grocery)
```

Output:

```
Grocery List: ['rice', 'wheat', 'dal', 'sugar']
```

Practice Problems (Try Yourself)

1. Create a list of your 3 favourite movies and print it.

2. Add a new movie using append() and display the updated list.

3. Replace the second item with another movie.

4. Remove the last item using pop().

5. Use len() to find the number of movies.

6. Sort a list of numbers.

7. Count how many times a value appears in a list.

8. Copy a list and modify the copy.
9. Merge two lists using extend().

Section 3: Know More (FAQs)

Q1. Can a list contain different data types?

Yes. A list can have strings, numbers, booleans, etc.

```
mixed = ["apple", 10, True]
```

Q2. Are lists ordered in Python?

Yes. Items are stored in the same order as they were added.

Q3. What happens if I use an invalid index?

Python will show an IndexError if the index is out of range.

Q4. What is the difference between remove() and pop()?

- remove() deletes the **item by value**.
- pop() deletes the **item by index** (default is last).

Q5. Can I store another list inside a list?

Yes, you can. This is called a **nested list**.

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

Q6. What does the sort() method do to strings?

It sorts them in **alphabetical order**.

```
names = ["Zara", "Amit", "Lata"]

names.sort()
```

```
print(names)
```

Output:

```
['Amit', 'Lata', 'Zara']
```

Q7. Can sort() sort numbers and strings in same list?

No. It will give an error if list has **mixed data types**.

Lists are like storage boxes in Python. Very handy for managing groups of values. Mastering lists will help you in handling all kinds of real-world data easily!