Why Loop Through a List?

You often need to **do something with every item** in a list — like:

- Print each name
- Check or filter data
- Calculate totals
- Send messages to users

You can use a **`for` loop** to handle each item one by one.

Looping with `for` Loop

Syntax:

```
for item in list_name:
    # do something with item
```

Example:

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

for fruit in fruits:
    print(fruit)
```

Output:

```
apple
banana
```

```
mango
```

Loop with Custom Messages

```
names = ["Anjali", "Ravi", "Tina"]

for name in names:
    print(f"Hello, {name}!")
```

Output:

```
Hello, Anjali!

Hello, Ravi!

Hello, Tina!
```

Looping Through Numbers in a List

```
marks = [75, 80, 92, 65]
total = 0

for mark in marks:
    total += mark

print("Total Marks:", total)
```

Output:

```
Total Marks: 312
```

Looping with `range()` and Indexes

If you want to access **both the index and the value**:

```
colors = ["red", "blue", "green"]

for i in range(len(colors)):
    print(f"Index {i} = {colors[i]}")
```

Output:

```
Index 0 = red
Index 1 = blue
Index 2 = green
```

Real-Life Use Case

Filter all items that match a condition:

```
numbers = [10, 15, 22, 30, 5]

for num in numbers:
    if num > 20:
        print(num)
```

Output:

```
22
30
```

Practice Questions

1. Create a list of 5 names and print "Hello `<name>`" to each

2. Print all numbers greater than 50 from a list
3. Add all marks in a list and print total
4. Print even numbers only from `[1, 2, 3, ..., 20]`
5. Loop through a list of hobbies and print them with bullet points
6. Ask the user to enter 5 items (one by one), store them in a list using `.append()`, then loop and print
7. Print index + value for this list: `["math", "science", "english"]`

What are List Functions?

List functions are **ready-made tools in Python** that help you:

- Organize your list
- Count items
- Copy a list
- Remove or reverse values

They make your life much easier when working with list data.

Most Common List Functions

`sort()` → Sorts list in ascending order

```
numbers = [3, 1, 5, 2]
numbers.sort()
print(numbers)
```

Output:

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

It sorts the original list itself (permanently changes it).

`reverse()` → Reverses the order of the list

```
names = ["Anjali", "Ravi", "Tina"]
names.reverse()
print(names)
```

Output:

```
['Tina', 'Ravi', 'Anjali']
```

`copy()` → Makes a copy of the list

```
list1 = [10, 20, 30]
list2 = list1.copy()
print(list2)
```

Output:

```
[10, 20, 30]
```

Now changing `list2` won't affect `list1`.

`count(value)` → Counts how many times a value appears

```
marks = [80, 90, 80, 70]
print(marks.count(80))
```

Output:

CopyEdit
2

`index(value)` → Finds the first position of a value

```
colors = ["red", "blue", "green"]
print(colors.index("blue"))
```

Output:

1

`extend()` → Joins two lists

```
a = [1, 2]
b = [3, 4]
a.extend(b)
print(a)
```

Output:

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

Practice Questions

1. Create a list of 5 numbers and sort them using `sort()`
2. Reverse the list `["cat", "dog", "lion"]`
3. Count how many times `"apple"` appears in this list: `["apple", "mango", "apple", "grape"]`
4. Find the index of `"banana"` in `["apple", "banana", "cherry"]`
5. Copy a list of 3 colors to a new variable
6. Combine two lists: one of fruits and one of vegetables using `extend()`
7. Try this: `list1 = [10, 20]; list2 = list1.copy(); list2.append(30)` — what is in both lists now?
8. Create a list of marks, and sort them in descending order (hint: sort + reverse)

What is a Nested List?

A **nested list** is a list **that contains other lists** as its elements.

> Think of it like a **table with rows and columns**.

Example:

```
students = [
    ["Amit", 18],
    ["Pooja", 19],
```

```
    ["Ravi", 20]
]
```

Here, each inner list represents a student's name and age.

Accessing Items in Nested Lists

You use **two indexes**:

```
print(students[0])      # ['Amit', 18]
print(students[0][0])   # 'Amit'
print(students[1][1])   # 19
```

Index	Inner List	Inner Index 0	Inner Index 1
0	["Amit", 18]	"Amit"	18
1	["Pooja", 19]	"Pooja"	19
2	["Ravi", 20]	"Ravi"	20
