**What is a for Loop?**

A for loop in Python is used to repeat a block of code for each item in a sequence (like a list, string, tuple, or range).

It’s count-based or item-based iteration — you know how many times or over which items the loop will run.

**Basic Syntax**

```
for variable in sequence:
    # code to execute
```

variable → holds the current item from the sequence during each loop

sequence → any iterable object (list, string, tuple, range, etc.)

**Examples**

**Looping Over a List**

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

**Looping Over a String**

```
for char in "Python":
    print(char)
```

**Using range()**

```
for i in range(5):  # 0 to 4
    print(i)
```

**Loop with Start, End, Step**

```
for i in range(1, 10, 2):  # 1, 3, 5, 7, 9
    print(i)
```

**Nested for Loop**

```
for x in range(1, 4):
    for y in range(1, 3):
        print(f"x={x}, y={y}")
```

**Real-Time Scenarios for Using for Loops**

☑ E-commerce → Looping through product lists to display them.

☑ Banking → Processing all transactions in a given day.

☑ Data Analysis → Iterating over rows in a dataset.

☑ Education → Printing all student names with marks.

☑ Games → Checking all players' scores in each round.

☑ IoT → Reading sensor data repeatedly for multiple devices.

☑ Automation → Sending emails to multiple recipients.

**Controlling the Flow in for Loops**

Sometimes, we don’t want a loop to run exactly from start to end —

we may need to skip steps, stop early, or do something only if the loop completes fully.

This is where break, continue, and else come in.

**break → Stop the loop immediately**

Meaning:

Ends the loop entirely, even if there are items left in the sequence.

When to use:

You found what you were searching for (search in a list).

You want to stop processing when a condition is met.

Example:

```
for num in range(10):
    if num == 5:
        print("Found 5, stopping loop!")
        break
    print(num)
```

Output:

0

1

2

3

4

Found 5, stopping loop!

Real-life analogy:

You’re looking for your lost key in drawers — once you find it, you stop opening drawers.

**Continue → Skip the current iteration**

Meaning:

Skips the rest of the code in the current loop and moves to the next item.

When to use:

You want to ignore certain values.

You want to skip processing for specific cases but still run for others.

Example:

```
for num in range(5):
    if num == 2:
        print("Skipping 2")
        continue
    print(num)
```

Output:

0

1

Skipping 2

3

4

Real-life analogy:

You're checking assignments from students — if one student was absent, you skip them and check the next.

**else in Loops → Run if no break is triggered**

Meaning:

The else block runs only if the loop finishes normally without hitting a break.

When to use:

To confirm that a search completed without finding anything.

To run code only if the loop didn’t stop early.

Example:

```
for num in range(5):
    if num == 10:
        break
    print(num)
else:
    print("Loop completed without break!")
```

Output:

0

1

2

3

4

Loop completed without break!

Real-life analogy:

You check all rooms in a hotel for a lost wallet. If you don’t stop early (break), you say "Checked every room, nothing found".

**Summary Table**

| Statement | Purpose | Loop Behavior | Runs Else? |
|---|---|---|---|
| break | Stop loop completely | Ends immediately | ❌ No |
| continue | Skip current iteration | Goes to next item | ☑ Yes |
| else | Code after loop (only if no break) | Runs after loop ends | ☑ Yes |

**Questions:**

1. Write a program that prints “Drink Water” 8 times — once for each hour between 9 AM and 5 PM.
2. A person saves ₹500 each month. Use a for loop to print the total savings after each month for 12 months.
3. Given a list of student names: ["Arjun", "Priya", "Ravi", "Meera"]
   Print a welcome message for each student like:
   "Welcome, Arjun!"
4. You have temperature readings for a week: [32, 34, 31, 29, 33, 35, 30]
   Use a for loop to print each day’s temperature with the text:
   "Day 1: 32°C"
5. Given item prices in a list: [250, 100, 50, 300]
   Print each item price and finally print the total bill.
6. A bus leaves in 5 minutes. Print a countdown:

   Bus leaves in 5 minutes

   Bus leaves in 4 minutes

   ...

   Bus leaves in 1 minute

7. Given a list of emails: ["a@gmail.com", "b@yahoo.com", "c@outlook.com"]

   Print: "Sending email to <email>" for each one.

8. For Diwali coming in 10 days, print:

10 days to Diwali!

9 days to Diwali!

...

1 day to Diwali!

9. Ask the user for a number and print its multiplication table from 1 to 10.
10. Given marks for students: [55, 72, 88, 40, 95]

    Print "Pass" if the mark is ≥ 50, else print "Fail".
11. Instead of a single table, display multiplication tables from 1 to 5 in a grid format.

    1x1=1  2x1=2  3x1=3  4x1=4  5x1=5

    1x2=2  2x2=4  3x2=6  4x2=8  5x2=10

    ...
12. A cinema has 3 rows and 5 seats per row. Display seat labels like Row1-Seat1, Row1-Seat2, ... using nested loops.
13. Create a diamond shape with stars (*) where the number of rows is given by the user.

      *

     ***

    *****

     ***

      *
14. Print all prime numbers between 10 and 50.
15. You have a list of possible passwords. Stop checking when the correct one is found.
16. Create a 3x3 grid of positions (1 to 9) for a Tic-Tac-Toe game.

1 2 3

4 5 6

7 8 9

17. Print a pyramid of numbers where each row contains the same number as the row number.

1

22

333

4444

55555

18. Given a dictionary of student marks: marks = {"Amit": [80, 85, 90], "Priya": [78, 82, 88], "Ravi": [92, 88, 84]} Print each student’s average marks.

19. Given a list of words, print only those that contain at least one vowel.

20. Display dates for January (31 days) in a weekly format — 7 dates per row.