🚀 21-Day Python Challenge – Day 5

Loops in Python

🎯 Day 5 Learning Objectives

By the end of today's class, you will be able to:

- Understand what a loop is and why we use it.
- Write programs using the **for loop**.
- Write programs using the **while loop**.
- Use **break** to stop a loop.
- Use **continue** to skip an iteration.
- Solve real-world problems using loops.

📓 What is a Loop?

A **loop** is used to execute the same block of code **multiple times**.

Real-Life Examples

- A teacher calls the attendance of **50 students**.
- A clock keeps ticking every second.
- Traffic signals repeat their colours continuously.
- A mobile alarm rings repeatedly until you stop it.

Instead of writing the same code many times, Python uses **loops**.

Without Loop

If you want to print **"Welcome to Python"** five times:

```
print("Welcome to Python")
print("Welcome to Python")
print("Welcome to Python")
print("Welcome to Python")
print("Welcome to Python")
```

This works, but it is repetitive.

With Loop

```
for i in range(5):
    print("Welcome to Python")
```

Output

```
Welcome to Python
Welcome to Python
Welcome to Python
Welcome to Python
Welcome to Python
```

One line of loop replaces many repeated statements.

Types of Loops

Python provides two main loops:

1. **for Loop**
2. **while Loop**

We will also learn:

- **break**
- **continue**

Part 1: for Loop

The **for loop** repeats a block of code a fixed number of times.

Syntax

```
for variable in range(number):
    statements
```

Understanding `range()`

Example

```
range(5)
```

Python automatically generates

```
0 1 2 3 4
```

Notice that **5 is not included**.

Example 1

```
for i in range(5):
    print(i)
```

Output

```
0
1
2
3
4
```

Example 2

```
for i in range(1,6):
    print(i)
```

Output

```
1
2
3
4
5
```

Explanation

```
range(1,6)

Start = 1
Stop = 6 (not included)

Output

1 2 3 4 5
```

Example 3

Print your name 10 times.

```
for i in range(10):
    print("Rahul")
```

Practical 1

Print numbers from 1 to 20.

```
for i in range(1,21):
    print(i)
```

Practical 2

Print even numbers from 2 to 20.

```
for i in range(2,21,2):
    print(i)
```

Explanation

```
range(2,21,2)

Start = 2

Stop = 21

Step = 2

Output

2 4 6 8 10 12 14 16 18 20
```

Practical 3

Print multiplication table of 5.

```
for i in range(1,11):
    print("5 x", i, "=", 5*i)
```

Output

```
5 x 1 = 5
5 x 2 = 10
...
5 x 10 = 50
```

Part 2: while Loop

The **while loop** repeats until the condition becomes **False**.

Syntax

```
while condition:
    statements
```

Example 1

```
count = 1

while count <= 5:
    print(count)
    count = count + 1
```

Output

```
1
2
3
4
5
```

How It Works

```
count = 1

↓

Is count <= 5?

↓

Yes

↓

Print 1

↓

Increase count

↓

Repeat
```

Example 2

Print your name five times.

```
count = 1

while count <= 5:
    print("GK InfoTech Solutions")
    count += 1
```

Practical 4

Print numbers from 10 to 1.

```
count = 10

while count >= 1:
    print(count)
    count -= 1
```

Part 3: break Statement

The **break** statement immediately stops the loop.

Example

```
for i in range(1,11):

    if i == 6:
        break

    print(i)
```

Output

```
1
2
3
4
5
```

Explanation

When **i becomes 6**, the loop stops immediately.

Practical 5

```
password = ""

while True:

    password = input("Enter Password: ")

    if password == "python123":
        print("Login Successful")
        break

    print("Wrong Password")
```

Part 4: continue Statement

The **continue** statement skips the current iteration and moves to the next iteration.

Example

```
for i in range(1,11):

    if i == 5:
        continue

    print(i)
```

Output

```
1
2
3
4
6
7
8
9
10
```

Notice that **5 is skipped**.

Practical 6

Print only odd numbers.

```
for i in range(1,21):

    if i % 2 == 0:
        continue

    print(i)
```

Real-Time Programs

Program 1: Sum of First 10 Numbers

```
total = 0

for i in range(1,11):
    total += i

print("Total =", total)
```

Output

```
Total = 55
```

Program 2: Factorial of a Number

```
num = int(input("Enter Number: "))

fact = 1

for i in range(1,num+1):
    fact *= i

print("Factorial =", fact)
```

Program 3: Password Verification

```
while True:

    password = input("Password: ")

    if password == "admin123":
        print("Access Granted")
        break

    print("Try Again")
```

Program 4: Print Squares

```
for i in range(1,11):
    print(i, "Square =", i*i)
```

Program 5: Countdown Timer

```
count = 10

while count >= 1:
    print(count)
    count -= 1

print("Time Up!")
```

Practice Programs

1. Print numbers from 1 to 100.
2. Print even numbers from 1 to 50.
3. Print odd numbers from 1 to 50.
4. Print your name 20 times.
5. Print the multiplication table of any number.
6. Find the sum of numbers from 1 to 100.
7. Find the factorial of a number.
8. Print squares of numbers from 1 to 20.
9. Print numbers in reverse from 20 to 1.
10. Create a password verification program using `while`.

Mini Project

Student Attendance

```
students = int(input("Enter Number of Students: "))

for i in range(1, students+1):

    name = input("Enter Student Name: ")

    print("Attendance Marked for", name)
```

Assignment

Write Python programs to:

1. Print numbers from 1 to 50.
2. Print even numbers from 2 to 100.
3. Print odd numbers from 1 to 99.
4. Print the multiplication table of any number.
5. Find the sum of the first 100 natural numbers.
6. Find the factorial of a given number.
7. Print the square and cube of numbers from 1 to 10.
8. Create a countdown timer from 20 to 1.
9. Create a password checker using `while` and `break`.
10. Print numbers from 1 to 20, skipping multiples of 3 using `continue`.

Interview Questions

1. What is a loop in Python?
2. Why do we use loops?
3. What is the difference between a `for` loop and a `while` loop?
4. What is the purpose of the `range()` function?
5. What are the three arguments of `range(start, stop, step)`?
6. What does the `break` statement do?
7. What does the `continue` statement do?
8. What happens if you forget to update the variable in a `while` loop?
9. Which loop is better when the number of repetitions is known?
10. Give one real-life example where loops are useful.

🏆 Day 5 Challenge

Student Marks Processing System

Write a program that:

- Accepts the number of students.
- Uses a **for loop** to enter each student's name and marks.
- Displays whether each student **Passed** or **Failed** (`marks >= 35`).
- Counts the total number of passed and failed students.
- Prints the final summary.

Example Output:

```
Student 1: Rahul - Pass
Student 2: Anitha - Fail
Student 3: Ravi - Pass

-------------------------
Total Students : 3
Passed         : 2
Failed         : 1
```

🌟 Day 5 Success Message

Congratulations! 🎉

Today, you learned one of the most powerful concepts in programming—**Loops**. They help you repeat tasks efficiently, making your code shorter, faster, and easier to maintain.

Remember:

- ✅ Use a **`for` loop** when you know **how many times** to repeat.
- ✅ Use a **`while` loop** when you repeat **until a condition changes**.
- ✅ Use **`break`** to stop a loop immediately.
- ✅ Use **`continue`** to skip the current iteration and move to the next.

"Great programmers don't write more code—they write smarter code. Loops help you automate repetitive tasks and think like a programmer."

🚀 **Next Lesson (Day 6): Strings – Indexing, Slicing, String Methods, and Formatting.**