Python

Learn to repeat actions efficiently in Python

About the Author

- Created By: Mohammad Salman
- Experience: 19 Years +
- Designation: Corporate Trainer .NET

Microsoft CERTIFIED Professional

Microsoft Specialist

Microsoft CERTIFIED Solutions Developer

Icons Used

Questions

Tools

Hands-on Exercise

Coding Standards

Questions?

Reference

Try it Out

Informative Slide

Mandatory Slide

Welcome Break

LOOPS IN PYTHON (BEGINNER LEVEL)

- Learn to repeat actions efficiently in Python
- By: Mohd Salman

LEARNING OBJECTIVES

- • Understand what loops are and why they are used
- • Use for and while loops correctly
- • Apply break, continue, and else with loops
- • Write simple programs using loops

WHAT ARE LOOPS?

- Loops help us repeat a block of code multiple times.
- They make code shorter, efficient, and readable.

- Without loop:
- print('Hello')
- print('Hello')
- print('Hello')

- With loop:
- for i in range(3):
- ```
      print('Hello')
  ```
  ```

TYPES OF LOOPS IN PYTHON

- 1. for loop – used to iterate over a sequence (list, tuple, string, range)
- 2. while loop – repeats until condition becomes False

FOR LOOP – SYNTAX

- Syntax:

```
for variable in sequence:
    # code block
```

Example:

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

FOR LOOP WITH LISTS AND STRINGS

- Example 1:

```
fruits = ['apple', 'banana', 'cherry']
for f in fruits:
    print(f)
```

Example 2:

```
for letter in 'Python':
    print(letter)
```

WHILE LOOP – SYNTAX

- Syntax:
- while condition:

  # code block

Example:

```
count = 1
while count <= 5:
    print('Count is:', count)
    count += 1
```

WHILE LOOP – EXAMPLE

- Example: Printing even numbers from 1 to 10

```
num = 2
while num <= 10:
    print(num)
    num += 2
```

BREAK STATEMENT

- Used to exit the loop immediately.

- Example:

```
for i in range(5):
    if i == 3:
        break
    print(i)
```

CONTINUE STATEMENT

Used to skip the current iteration.

Example:

```
for i in range(5):
    if i == 2:
        continue
    print(i)
```

FOR-ELSE LOOP

Syntax:

```
for item in sequence:
    # code
else:
    # runs if loop not broken
```

Example:

```
for i in range(3):
    print(i)
else:
    print('Loop completed!')
```

NESTED LOOPS

- Loop inside another loop.

- Example:

```
for i in range(1, 4):
    for j in range(1, 4):
        print(i, j)
```

USING RANGE() FUNCTION

Syntax:

range(start, stop, step)

Examples:

range(5) → 0,1,2,3,4

range(2, 6) → 2,3,4,5

range(1, 10, 2) → 1,3,5,7,9

PRACTICE EXAMPLES

- 1. Print numbers 1 to 10 using for loop
- 2. Print even numbers 2 to 20 using while loop
- 3. Print multiplication table of 5
- 4. Print each character of your name
- 5. Print all numbers divisible by 3 and 5 between 1–100

SIMPLE REAL-LIFE EXAMPLE

- Program: Countdown before game starts

```
import time
count = 5
while count > 0:
    print('Game starts in', count)
    time.sleep(1)
    count -= 1
print('Go!')
```

- for loop – iterate over sequence
- while loop – repeat until condition false
- break – stop loop early
- continue – skip iteration
- else – after normal loop finish

LOOP SUMMARY TABLE

QUICK QUIZ

1. Difference between for and while?
2. What if you forget increment in while?
3. What does break do?
4. Output?

```
for i in range(3):
    print(i)
else:
    print('Done')
```

SUMMARY

- Loops make repetitive tasks easy
- for loop – use for sequence
- while loop – use for condition
- break, continue, else – control flow
- Practice is key!

HANDS-ON LAB TASKS

1. Factorial using while loop
2. Count vowels in a word using for loop
3. Print pattern:

*

**

***

****

4. Sum of numbers from 1–100

- Code. Run. Repeat. That's how loops and learning both work.

THANK YOU!