Python Notes – Loops & Control Statements

Prepared by: **Dhananjay Kumar**
Trainer | Java Full-Stack & Python | Contact: 7023577968
LinkedIn: Dhananjay Kumar

Today's Agenda

1. Concept of Nested Looping
2. `for` Loop and `range()` method
3. Loop Control Statements: `break`, `continue`, `pass`
4. Pattern Design using Nested Loops

1️⃣ Concept of Nested Looping

A **nested loop** means one loop inside another loop.

- Outer loop runs first.
- For each iteration of the outer loop, the inner loop executes completely.

🔨 Formula:

In [4]:
```
# Example 1: Nested Loop Printing
for i in range(3):        # Outer loop
    for j in range(2):    # Inner loop
        print(f"i={i}, j={j}")
```

```
i=0, j=0
i=0, j=1
i=1, j=0
i=1, j=1
i=2, j=0
i=2, j=1
```

Example 1:

2️⃣ `for` Loop and `range()` Method

A `for` loop is used for iterating over a sequence.
`range()` generates a sequence of numbers.

Syntax:

```
for variable in range(start, stop, step):
    # code
```

In [6]:
```
# Example 2: Using range() in different ways

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

print("\nrange(2, 6):")
for i in range(2, 6):
    print(i)

print("\nrange(1, 10, 2):")
for i in range(1, 10, 2):
    print(i)
```

```
range(5):
0
1
2
3
4

range(2, 6):
2
3
4
5

range(1, 10, 2):
1
3
5
7
9
```

3️⃣ Loop Control Statements

(a) `break` Statement

Used to **exit the loop** immediately.

In [7]:
```python
# Example 3: break usage
for i in range(1, 10):
    if i == 5:
        print("Breaking at", i)
        break
    print(i)
```

```
1
2
3
4
Breaking at 5
```

(b) `continue` Statement

Used to **skip current iteration** and move to the next.

In [8]:
```python
# Example 4: continue usage
for i in range(1, 6):
    if i == 3:
        continue  # skip when i == 3
    print(i)
```

```
1
2
4
5
```

(c) `pass` Statement

Used as a placeholder, does nothing.

In [9]:
```python
# Example 5: pass usage
for i in range(1, 6):
    if i == 3:
        pass  # just a placeholder
    print(i)
```

```
1
2
3
4
5
```

4️⃣ Pattern Design Using Nested Loops

Pattern questions are the most common in interviews and exams.
We use **nested loops** to design them.

⭐ Question 1: Print a square of stars

In [10]:
```
# Pattern: 3x3 square
for i in range(3):
    for j in range(3):
        print("*", end=" ")
    print()   # new line
```

```
* * *
* * *
* * *
```

⭐ Question 2: Print a right-angled triangle

In [11]:
```
# Right-angled triangle
for i in range(1, 6):
    for j in range(i):
        print("*", end=" ")
    print()
```

```
*
* *
* * *
* * * *
* * * * *
```

⭐ Question 3: Print numbers in a pattern

In [12]:
```
# Number pattern
for i in range(1, 6):
    for j in range(1, i+1):
        print(j, end=" ")
    print()
```

```
1
1 2
1 2 3
1 2 3 4
1 2 3 4 5
```

⭐ Question 4: Inverted triangle

In [13]:
```
# Inverted star triangle
for i in range(5, 0, -1):
    for j in range(i):
        print("*", end=" ")
    print()
```

```
* * * * *
* * * *
* * *
* *
*
```

⭐ Question 5: Pyramid Pattern

In [14]:
```
# Pyramid using spaces
rows = 5
for i in range(1, rows+1):
    print(" " * (rows-i), end="")  # spaces
    print("* " * i)
```

```
    *
   * *
  * * *
 * * * *
* * * * *
```

Q6: Print numbers from 1 to 10 using range()

In [16]:
```
for i in range(1, 11):
    print(i, end=" ")

#  Explanation:
# range(1, 11) → generates 1 to 10
```

```
1 2 3 4 5 6 7 8 9 10
```

Q7: Print even numbers from 2 to 20

In [17]:
```
for i in range(2, 21, 2):
```

```
    print(i, end=" ")

#  Explanation:
# range(start=2, stop=21, step=2) → generates even numbers.
```

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

Q8: Print numbers in reverse from 10 to 1

In [18]:
```
for i in range(10, 0, -1):
    print(i, end=" ")

#  Explanation:
# Negative step → countdown.
```

```
10 9 8 7 6 5 4 3 2 1
```

Q9: Print squares of numbers from 1 to 5

In [19]:
```
for i in range(1, 6):
    print(f"{i}^2 = {i*i}")

#  Explanation:
# Each iteration calculates square of i.
```

```
1^2 = 1
2^2 = 4
3^2 = 9
4^2 = 16
5^2 = 25
```

Q10: Calculate sum of first 10 natural numbers

In [21]:
```
total = 0
for i in range(1, 11):
    total += i
print("Sum =", total)

#  Explanation:
# Keep adding i into total.
```

```
Sum = 55
```

Q11: Use break to stop loop when number is 7

In [22]:
```
for i in range(1, 11):
    if i == 7:
        break
    print(i)

#  Explanation:
# Loop stops completely when i == 7.
```

```
1
2
3
4
5
6
```

Q12: Use continue to skip multiples of 3 between 1–10

In [23]:
```
for i in range(1, 11):
    if i % 3 == 0:
        continue
    print(i)

#  Explanation:
# continue skips numbers divisible by 3.
```

```
1
2
4
5
7
8
10
```

Q13: Use pass statement inside loop

In [24]:
```
for i in range(5):
    if i == 2:
        pass  # placeholder, does nothing
    print(i)

#  Explanation:
# pass is used as empty block (no error).
```

```
0
1
2
3
4
```

Q14: Find first even number and stop

In [25]:
```
for i in range(1, 10):
    if i % 2 == 0:
        print("First even:", i)
        break

#  Explanation:
# Loop ends immediately when first even is found.
```

```
First even: 2
```

Q15: Skip printing 5 using continue

In [26]:
```
for i in range(1, 8):
    if i == 5:
        continue
    print(i)

#  Explanation:
# continue skips i == 5.
```

```
1
2
3
4
6
7
```

Python MCQs – Loops, Nested Loops & Patterns

Q1. What will be the output of the following code?

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

```
A) (0,0) (0,1) (1,0) (1,1)

B) (0,0) (1,0) (0,1) (1,1)

C) Infinite loop

D) Error

✅ Answer: A
 Explanation: Outer loop (i) runs 0→1, Inner loop (j) runs 0→1 → total 4 pairs in order.

**Q2.** How many times will `print("Hello")` execute?


for i in range(3):
    for j in range(4):
        print("Hello")



A) 3

B) 4

C) 7

D) 12
```

✅ Answer: D
Explanation: Outer loop runs 3 times, inner loop runs 4 times each → 3×4 = 12 prints.

Q3. Which is true about nested loops in Python?

- A) Inner loop executes only once.
- B) Inner loop runs completely for every iteration of outer loop.
- C) Outer loop executes after inner loop finishes all iterations.
- D) Both B and C

✅ **Answer: D**
Explanation: Inner loop runs fully for each outer iteration, then outer moves ahead.

Q4. What will be printed?

```
for i in range(2, 4):
    for j in range(1, 3):
        print(i+j, end=" ")
```

A) 3 4 4 5

B) 2 3 3 4

C) 4 5 5 6

D) 1 2 3 4

✅ Answer: A
Explanation: (2+1=3, 2+2=4, 3+1=4, 3+2=5).

Q5. Nested loops are mainly used for:

- A) Simple iteration
- B) Printing patterns
- C) Working with multi-dimensional data
- D) Both B and C

✅ **Answer: D**
Explanation: Nested loops are required when handling grids, matrices, or patterns.

Q6. What does `range(5)` generate?

- A) 0,1,2,3,4
- B) 1,2,3,4,5
- C) 5 numbers starting from 1
- D) Error

✅ **Answer: A**
Explanation: By default, `range(n)` starts from 0 → (0 to n-1).

Q7. Output of:

```
for i in range(2, 10, 3):
    print(i, end=" ")
```

A) 2 3 4 5 6 7 8 9

B) 2 5 8

C) 2 5 7 10

D) Error

✅ Answer: B

Q8. Which statement is false?

- A) `range(5)` is same as `range(0,5)`
- B) `range(1,5)` gives 1,2,3,4
- C) `range(5,1,-1)` gives 5,4,3,2
- D) `range(1,5,-1)` gives 1,0

✅ **Answer: D**
Explanation: Negative step with start < stop gives empty sequence.

Q9. Sum of numbers generated by `range(1, 6)`?

- A) 10
- B) 15
- C) 20
- D) 21

✅ **Answer: B**
Explanation: 1+2+3+4+5 = 15.

Q10. Which of the following is valid?

- A) `range(5, 15, 2)`
- B) `range(10, 0, -2)`
- C) `range(0, 10)`
- D) All of the above

✅ **Answer: D**
Explanation: All are valid variations of `range()`.

In [ ]: