Day21_Pattern_&_For_Else_In_Loops

June 10, 2025

Today I focused on mastering Python's `for` loops, `for-else` structure, and different types of **pattern printing** that are commonly used. These concepts build the foundation for solving logic-based coding problems and help in writing efficient, readable code.

[1]:
```
## 1. Skip numbers divisible by 3 or 5
for i in range(1, 51):
    if i % 3 == 0 or i % 5 == 0:
        continue
    print(i)
# Skips all values divisible by 3 or 5

print("---")
```

```
1
2
4
7
8
11
13
14
16
17
19
22
23
26
28
29
31
32
34
37
38
41
43
44
46
47
```

```
49
---
```

[2]:
```
## 2. Skip numbers divisible by both 3 and 5 (i.e., 15, 30, 45...)
for i in range(1, 50):
    if i % 3 == 0 and i % 5 == 0:
        continue
    print(i)
print("end")

print("---")
```

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
16
17
18
19
20
21
22
23
24
25
26
27
28
29
31
32
33
34
35
36
37
38
```

```
39
40
41
42
43
44
46
47
48
49
end
---
```

[3]:
```
## 3. Print only odd numbers
for i in range(1, 51):
    if i % 2 == 0:
        continue
    else:
        print(i)
print("bye")

print("---")
```

```
1
3
5
7
9
11
13
15
17
19
21
23
25
27
29
31
33
35
37
39
41
43
45
47
49
```

```
bye
---
```

1 Print Pattern

Tips About Patterns in Python

- Patterns are created using **nested loops**, where the outer loop handles rows and the inner loop handles columns or symbols.
- You can use `print("#", end=" ")` to keep symbols on the same line.
- Adjusting the number of iterations and `end` spacing helps create different pattern shapes (squares, triangles, pyramids).
- Changing loop conditions allows for reverse patterns, alignment, and stepwise effects.
- Always use `print()` after the inner loop to move to a new line.

[4]:
```python
## 4. Manual pattern print
print("# # # #")
print("# # # #")
print("# # # #")
print("# # # #")

# Not best way for large pattern and not a way to print, big no
```

```
# # # #
# # # #
# # # #
# # # #
```

[5]:
```python
## 5. Using loop to print same pattern
for i in range(1, 5):
    i = i + 1
    print("# # # #")

print("---")
```

```
# # # #
# # # #
# # # #
# # # #
---
```

[6]:
```python
## 6. Conditional printing inside loop
for i in range(1, 5):
    if i <= 5:
        print("# # # #")

print("---")
```

```
# # # #
# # # #
# # # #
# # # #
---
```

[7]:
```
## 7. Print single column
for j in range(4):
    print("#")

print("---")
```

```
#
#
#
#
---
```

[8]:
```
## 8. Print full square with spacing
for j in range(4):
    print("# # # #")

print("---")
```

```
# # # #
# # # #
# # # #
# # # #
---
```

[10]:
```
## 9. Inline hash symbols
for j in range(4):
    print("#", end=" ")
print()
for j in range(4):
    print("#", end=" ")
print()

print("---")
```

```
# # # #
# # # #
---
```

[11]:
```
## 10. Variable spacing between hashes
for j in range(4):
    print("#", end=" ")
print()

for j in range(4):
    print("#", end="  ")
print()

for j in range(4):
    print("#", end="  ")
print()

for j in range(4):
    print("#", end="  ")
print()

print("---")
```

```
# # # #
#  #  #  #
#  #  #  #
#  #  #  #
---
```

[12]:
```
## 11. Nested loop square
for i in range(4):
    for j in range(4):
        print("#", end="  ")
    print()

print("---")
```

```
#  #  #  #
#  #  #  #
#  #  #  #
#  #  #  #
---
```

[13]:
```
## 12. Right-angle triangle
for i in range(1, 5):
    print("# " * i)
```

```
#
# #
# # #
# # # #
```

[14]:
```
## 13. Triangle using condition
for i in range(1, 5):
    for j in range(4):
        if i > j:
            print("#", end=" ")
    print()
print("---")
```

```
#
# #
# # #
# # # #
---
```

[15]:
```
## 14. Using list(range())
print(list(range(5)))

print("---")
```

```
[0, 1, 2, 3, 4]
---
```

[16]:
```
## 15. Left aligned triangle
for i in range(4):
    for j in range(i):
        print("#", end="  ")
    print()

print("---")
```

```

#
#  #
#  #  #
---
```

[17]:
```
## 16. Triangle pattern with +1
for i in range(4):
    for j in range(i + 1):
        print("#", end="  ")
    print()

print("---")
```

```
#
#  #
#  #  #
#  #  #  #
---
```

[18]:
```
## 17. Reverse triangle pattern
for i in range(4):
    for j in range(4 - i):
        print("#", end="  ")
    print()
print("---")
```

```
#   #   #   #
#   #   #
#   #
#
---
```

2 For - Else

Tips About For-Else in Python

- `for-else` is a unique Python feature not available in many other languages.
- The `else` block executes **only if the loop completes without encountering a `break`**.
- Common use cases:
  - Searching for a value in a list
  - Checking if a number is prime
  - Validating conditions without additional flags
- Helps eliminate the need for an external boolean flag (`found = False` pattern).

[19]:
```
## 18. For-Else in Python - find first divisible by 5
nums = [12, 15, 18, 21, 26, 30, 40]
for num in nums:
    if num % 5 == 0:
        print(num)
```

```
15
30
40
```

[20]:
```
## 19. Multiple divisible values
nums = [12, 14, 18, 21, 25, 30, 35]
for num in nums:
    if num % 5 == 0:
        print(num)
```

```
25
30
35
```

[21]:
```
## 20. Using break - stop at first divisible by 5
nums = [12, 14, 18, 21, 20, 25]
for num in nums:
    if num % 5 == 0:
        print(num)
        break
```

```
20
```

[22]:
```
## 21. For-Else example
nums = [7, 14, 18, 21, 23, 27, 29]
for num in nums:
    if num % 5 == 0:
        print(num)
        break
else:
    print('Number Not Found')
```

```
Number Not Found
```

[ ]:
```
## 22. Prime number check
num = 14
for i in range(2, num):
    if num % i == 0:
        print('Not prime Number')
        break
else:
    print('Prime Number')
```

3 Pattern Examples

[23]:
```
## 1. Right-Angled Triangle
for i in range(1, 6):
    for j in range(i):
        print("*", end=" ")
    print()
```

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

[24]:
```
## 2. Inverted Right-Angled Triangle
for i in range(5, 0, -1):
    for j in range(i):
        print("*", end=" ")
    print()
```

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

[25]:
```
## 3. Pyramid Pattern
for i in range(1, 6):
    print(" " * (5 - i) + "* " * i)
```

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

[26]:
```
## 4. Number Triangle
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
```

[27]:
```
## 5. Floyd's Triangle
num = 1
for i in range(1, 6):
    for j in range(i):
        print(num, end=" ")
        num += 1
    print()
```

```
1
2 3
4 5 6
7 8 9 10
11 12 13 14 15
```

4 For-Else Examples

[28]:
```
## 1. Prime Number Checker
num = 17
for i in range(2, num):
    if num % i == 0:
        print("Not Prime")
        break
else:
    print("Prime")
```

```
Prime
```

[29]:
```python
## 2. Search for Item in List
items = [12, 24, 35, 47, 60]
for item in items:
    if item == 47:
        print("Found 47")
        break
else:
    print("47 Not Found")
```

```
Found 47
```

[30]:
```python
## 3. Check All Items are Even
nums = [2, 4, 6, 8, 10]
for n in nums:
    if n % 2 != 0:
        print("Not all numbers are even")
        break
else:
    print("All numbers are even")
```

```
All numbers are even
```

[31]:
```python
## 4. Password Check Simulation
attempts = ["admin", "root", "guest"]
for attempt in attempts:
    if attempt == "root":
        print("Access granted")
        break
else:
    print("Access denied")
```

```
Access granted
```