Python Programming

Interview Practice Guide

Operators, Modules, Control Flow & Loops

With Code Examples and Outputs

1. Arithmetic Operators

Arithmetic operators perform mathematical operations like addition, subtraction, multiplication, division, modulus, exponentiation, and floor division. These are fundamental for calculations.

```
# Example 1: Basic Addition
a = 10
b = 5
result = a + b
print(f'{a} + {b} = {result}')
```

```
Output: 10 + 5 = 15
```

```
# Example 2: Subtraction
x = 20
y = 8
print(f'{x} - {y} = {x - y}')
```

```
Output: 20 - 8 = 12
```

```
# Example 3: Multiplication
num1 = 7
num2 = 6
product = num1 * num2
print(f'{num1} × {num2} = {product}')
```

```
Output: 7 × 6 = 42
```

```
# Example 4: Division (float result)
a = 15
b = 4
print(f'{a} / {b} = {a / b}')
```

```
Output: 15 / 4 = 3.75
```

```
# Example 5: Floor Division
a = 17
b = 5
print(f'{a} // {b} = {a // b}')
```

```
Output: 17 // 5 = 3
```

```
# Example 6: Modulus (remainder)
a = 17
b = 5
print(f'{a} % {b} = {a % b}')
```

```
Output: 17 % 5 = 2
```

```
# Example 7: Exponentiation (power)
base = 2
exp = 8
print(f'{base}^{exp} = {base ** exp}')
```

```
Output: 2^8 = 256
```

```
# Example 8: Negative numbers
a = -10
b = 3
print(f'{a} + {b} = {a + b}')
```

```
Output: -10 + 3 = -7
```

```
# Example 9: Mixed operations (BODMAS)
result = 2 + 3 * 4
print(f'2 + 3 * 4 = {result}')
```

```
Output: 2 + 3 * 4 = 14
```

```
# Example 10: Parentheses priority
result = (2 + 3) * 4
print(f'(2 + 3) * 4 = {result}')
```

```
Output: (2 + 3) * 4 = 20
```

```
# Example 11: Check even/odd
num = 17
if num % 2 == 0:
    print(f'{num} is Even')
else:
    print(f'{num} is Odd')
```

```
Output: 17 is Odd
```

```
# Example 12: Swap without temp variable
a, b = 5, 10
print(f'Before: a={a}, b={b}')
a, b = b, a
print(f'After: a={a}, b={b}')
```

```
Output: Before: a=5, b=10
After: a=10, b=5
```

```
# Example 13: Calculate average
num1, num2, num3 = 85, 90, 78
avg = (num1 + num2 + num3) / 3
print(f'Average: {avg:.2f}')
```

```
Output: Average: 84.33
```

```
# Example 14: Compound interest
p, r, t = 1000, 5, 2
ci = p * ((1 + r/100) ** t - 1)
print(f'Compound Interest: {ci:.2f}')
```

```
Output: Compound Interest: 102.50
```

```
# Example 15: Temperature conversion
celsius = 25
fahrenheit = (celsius * 9/5) + 32
print(f'{celsius}°C = {fahrenheit}°F')
```

```
Output: 25°C = 77.0°F
```

2. Relational (Comparison) Operators

Relational operators compare two values and return Boolean (True/False). They include: == (equal), != (not equal), > (greater), < (less), >= (greater or equal), <= (less or equal).

```
# Example 1: Equal to (==)
a = 10
b = 10
print(f'{a} == {b}: {a == b}')
```

```
Output: 10 == 10: True
```

```
# Example 2: Not equal to (!=)
x = 5
y = 8
print(f'{x} != {y}: {x != y}')
```

```
Output: 5 != 8: True
```

```
# Example 3: Greater than (>)
num1 = 15
num2 = 10
print(f'{num1} > {num2}: {num1 > num2}')
```

```
Output: 15 > 10: True
```

```
# Example 4: Less than (<)
age1 = 18
age2 = 21
print(f'{age1} < {age2}: {age1 < age2}')
```

```
Output: 18 < 21: True
```

```
# Example 5: Greater than or equal (>=)
score = 75
pass_mark = 75
print(f'{score} >= {pass_mark}: {score >= pass_mark}')
```

```
Output: 75 >= 75: True
```

```
# Example 6: Less than or equal (<=)
temp = 30
max_temp = 35
print(f'{temp} <= {max_temp}: {temp <= max_temp}')
```

```
Output: 30 <= 35: True
```

```
# Example 7: Comparing strings
str1 = 'apple'
str2 = 'banana'
print(f"'{str1}' < '{str2}': {str1 < str2}")
```

```
Output: 'apple' < 'banana': True
```

```
# Example 8: Float comparison
pi = 3.14159
approx = 3.14
print(f'{pi} > {approx}: {pi > approx}')
```

```
Output: 3.14159 > 3.14: True
```

```
# Example 9: Chaining comparisons
x = 15
result = 10 < x < 20
print(f'10 < {x} < 20: {result}')
```

```
Output: 10 < 15 < 20: True
```

```
# Example 10: Comparing with None
value = None
print(f'value == None: {value == None}')
print(f'value is None: {value is None}')
```

```
Output: value == None: True
value is None: True
```

```
# Example 11: Password validation
password = 'secret123'
correct_pwd = 'secret123'
if password == correct_pwd:
    print('Access Granted')
else:
    print('Access Denied')
```

```
Output: Access Granted
```

```
# Example 12: Age eligibility check
age = 17
if age >= 18:
    print('Eligible to vote')
else:
    print(f'Wait {18-age} more year(s)')
```

```
Output: Wait 1 more year(s)
```

```
# Example 13: Grade comparison
marks = 85
if marks >= 90:
    grade = 'A'
elif marks >= 80:
    grade = 'B'
else:
    grade = 'C'
print(f'Grade: {grade}')
```

```
Output: Grade: B
```

```
# Example 14: Finding maximum
a, b, c = 15, 28, 22
max_num = a
if b > max_num:
    max_num = b
if c > max_num:
    max_num = c
print(f'Maximum: {max_num}')
```

```
Output: Maximum: 28
```

```
# Example 15: String length comparison
name1 = 'John'
name2 = 'Alexander'
if len(name1) < len(name2):
    print(f'{name2} is longer')
```

```
Output: Alexander is longer
```

3. Logical Operators

Logical operators combine conditional statements: and (both conditions true), or (at least one true), not (reverses the result). Used extensively in complex conditions.

```
# Example 1: Logical AND
age = 25
has_license = True
can_drive = age >= 18 and has_license
print(f'Can drive: {can_drive}')
```

```
Output: Can drive: True
```

```
# Example 2: Logical OR
is_weekend = False
is_holiday = True
can_rest = is_weekend or is_holiday
print(f'Can rest: {can_rest}')
```

```
Output: Can rest: True
```

```
# Example 3: Logical NOT
is_raining = False
can_play_outside = not is_raining
print(f'Can play outside: {can_play_outside}')
```

```
Output: Can play outside: True
```

```
# Example 4: Multiple AND conditions
marks = 85
attendance = 90
if marks >= 75 and attendance >= 85:
    print('Eligible for scholarship')
else:
    print('Not eligible')
```

```
Output: Eligible for scholarship
```

```
# Example 5: Combining AND & OR
age = 65
is_student = False
if age < 18 or age > 60 or is_student:
    print('Discount: 20%')
else:
    print('No discount')
```

```
Output: Discount: 20%
```

```
# Example 6: NOT with comparison
password = 'abc123'
if not (password == 'admin'):
    print('Invalid password')
```

```
Output: Invalid password
```

```
# Example 7: Login validation
username = 'john'
password = 'pass123'
if username == 'john' and password == 'pass123':
    print('Login Successful')
else:
    print('Invalid credentials')
```

```
Output: Login Successful
```

```
# Example 8: Range checking with AND
score = 75
if score >= 60 and score <= 100:
    print('Valid score')
else:
    print('Invalid score')
```

```
Output: Valid score
```

```
# Example 9: Short-circuit evaluation
x = 0
y = 10
result = (x != 0) and (y / x > 2)
print(f'Result: {result}')
```

```
Output: Result: False
```

```
# Example 10: Complex condition
temp = 25
humidity = 70
if (temp > 20 and temp < 30) and humidity < 80:
    print('Weather: Pleasant')
else:
    print('Weather: Uncomfortable')
```

```
Output: Weather: Pleasant
```

```
# Example 11: Eligibility check
age = 16
has_guardian = True
can_watch = age >= 18 or (age >= 13 and has_guardian)
print(f'Can watch movie: {can_watch}')
```

```
Output: Can watch movie: True
```

```
# Example 12: NOT with membership
fruits = ['apple', 'banana']
if not 'orange' in fruits:
    print('Orange not available')
```

```
Output: Orange not available
```

```
# Example 13: Multiple OR conditions
day = 'Saturday'
if day == 'Saturday' or day == 'Sunday':
    print('Weekend!')
else:
    print('Weekday')
```

```
Output: Weekend!
```

```
# Example 14: Nested logical operators
x = 15
y = 20
z = 25
result = (x < y) and (y < z) or (x == 15)
print(f'Result: {result}')
```

```
Output: Result: True
```

```
# Example 15: Boundary validation
value = 50
if not (value < 0 or value > 100):
    print('Value within range')
else:
    print('Value out of range')
```

```
Output: Value within range
```

4. Bitwise Operators

Bitwise operators work on bits (binary representation) of numbers. They include: & (AND), | (OR), ^ (XOR), ~ (NOT), << (left shift), >> (right shift). Important for low-level programming.

```
# Example 1: Bitwise AND (&)
a = 5  # Binary: 0101
b = 3  # Binary: 0011
result = a & b  # Binary: 0001
print(f'{a} & {b} = {result}')
```

```
Output: 5 & 3 = 1
```

```
# Example 2: Bitwise OR (|)
a = 5  # Binary: 0101
b = 3  # Binary: 0011
result = a | b  # Binary: 0111
print(f'{a} | {b} = {result}')
```

```
Output: 5 | 3 = 7
```

```
# Example 3: Bitwise XOR (^)
a = 5  # Binary: 0101
b = 3  # Binary: 0011
result = a ^ b  # Binary: 0110
print(f'{a} ^ {b} = {result}')
```

```
Output: 5 ^ 3 = 6
```

```
# Example 4: Bitwise NOT (~)
a = 5  # Binary: 0101
result = ~a  # Binary: ...11111010 (two's complement)
print(f'~{a} = {result}')
```

```
Output: ~5 = -6
```

```
# Example 5: Left shift (<<)
a = 5  # Binary: 0101
result = a << 2  # Binary: 010100
print(f'{a} << 2 = {result}')
```

```
Output: 5 << 2 = 20
```

```
# Example 6: Right shift (>>)
a = 20  # Binary: 10100
result = a >> 2  # Binary: 00101
print(f'{a} >> 2 = {result}')
```

```
Output: 20 >> 2 = 5
```

```
# Example 7: Check if number is even/odd using &
num = 17
if num & 1:
    print(f'{num} is Odd')
else:
    print(f'{num} is Even')
```

```
Output: 17 is Odd
```

```
# Example 8: Swap using XOR
a, b = 10, 20
print(f'Before: a={a}, b={b}')
a = a ^ b
b = a ^ b
a = a ^ b
print(f'After: a={a}, b={b}')
```

```
Output: Before: a=10, b=20
After: a=20, b=10
```

```
# Example 9: Multiply by 2 using left shift
num = 7
result = num << 1
print(f'{num} * 2 = {result}')
```

```
Output: 7 * 2 = 14
```

```
# Example 10: Divide by 2 using right shift
num = 16
result = num >> 1
print(f'{num} / 2 = {result}')
```

```
Output: 16 / 2 = 8
```

```python
# Example 11: Check if power of 2
num = 16
is_power_of_2 = (num & (num - 1)) == 0 and num != 0
print(f'{num} is power of 2: {is_power_of_2}')
```

```
Output: 16 is power of 2: True
```

```python
# Example 12: Count set bits
num = 13  # Binary: 1101
count = bin(num).count('1')
print(f'Set bits in {num}: {count}')
```

```
Output: Set bits in 13: 3
```

```python
# Example 13: Toggle specific bit
num = 5  # Binary: 0101
position = 1
result = num ^ (1 << position)  # Toggle bit at position 1
print(f'After toggle: {result}')
```

```
Output: After toggle: 7
```

```python
# Example 14: Check if bit is set
num = 10  # Binary: 1010
position = 3
is_set = (num & (1 << position)) != 0
print(f'Bit {position} is set: {is_set}')
```

```
Output: Bit 3 is set: True
```

```python
# Example 15: Clear rightmost set bit
num = 12  # Binary: 1100
result = num & (num - 1)  # Binary: 1000
print(f'After clearing: {result}')
```

```
Output: After clearing: 8
```

5. Assignment Operators

Assignment operators assign values to variables. Beyond simple =, compound operators like +=, -=, *=, /=, //=, %=, **=, &=, |=, ^=, <<=, >>= combine operation with assignment.

```
# Example 1: Simple assignment (=)
x = 10
print(f'x = {x}')
```

```
Output: x = 10
```

```
# Example 2: Add and assign (+=)
count = 5
count += 3  # count = count + 3
print(f'count = {count}')
```

```
Output: count = 8
```

```
# Example 3: Subtract and assign (-=)
balance = 100
balance -= 30  # balance = balance - 30
print(f'balance = {balance}')
```

```
Output: balance = 70
```

```
# Example 4: Multiply and assign (*=)
price = 50
price *= 2  # price = price * 2
print(f'price = {price}')
```

```
Output: price = 100
```

```
# Example 5: Divide and assign (/=)
total = 100
total /= 4  # total = total / 4
print(f'total = {total}')
```

```
Output: total = 25.0
```

```
# Example 6: Floor divide and assign (//=)
num = 17
num //= 5  # num = num // 5
print(f'num = {num}')
```

```
Output: num = 3
```

```
# Example 7: Modulus and assign (%=)
value = 17
value %= 5  # value = value % 5
print(f'value = {value}')
```

```
Output: value = 2
```

```
# Example 8: Exponent and assign (**=)
base = 2
base **= 3  # base = base ** 3
print(f'base = {base}')
```

```
Output: base = 8
```

```
# Example 9: Bitwise AND and assign (&=)
flags = 12  # 1100
flags &= 10  # 1010
print(f'flags = {flags}')
```

```
Output: flags = 8
```

```
# Example 10: Bitwise OR and assign (|=)
perms = 4  # 0100
perms |= 2  # 0010
print(f'perms = {perms}')
```

```
Output: perms = 6
```

```
# Example 11: Bitwise XOR and assign (^=)
state = 5  # 0101
state ^= 3  # 0011
print(f'state = {state}')
```

```
Output: state = 6
```

```
# Example 12: Left shift and assign (<<=)
num = 3
```

```
num <<= 2  # num = num << 2
print(f'num = {num}')
```

```
Output: num = 12
```

```
# Example 13: Right shift and assign (>>=)
num = 16
num >>= 2  # num = num >> 2
print(f'num = {num}')
```

```
Output: num = 4
```

```
# Example 14: Multiple assignments
x = y = z = 0
x += 5
y += 10
z += 15
print(f'x={x}, y={y}, z={z}')
```

```
Output: x=5, y=10, z=15
```

```
# Example 15: Accumulating values
total = 0
for i in range(1, 6):
    total += i
print(f'Sum 1 to 5: {total}')
```

```
Output: Sum 1 to 5: 15
```

6. Membership Operators

Membership operators test if a value exists in a sequence (string, list, tuple, set, dictionary). The operators are: in (returns True if found) and not in (returns True if not found).

```
# Example 1: in operator with list
fruits = ['apple', 'banana', 'cherry']
print('banana' in fruits)
```

```
Output: True
```

```
# Example 2: not in operator with list
fruits = ['apple', 'banana', 'cherry']
print('mango' not in fruits)
```

```
Output: True
```

```
# Example 3: in operator with string
text = 'Python Programming'
print('Python' in text)
```

```
Output: True
```

```
# Example 4: Substring checking
email = 'user@example.com'
if '@' in email and '.' in email:
    print('Valid email format')
else:
    print('Invalid email')
```

```
Output: Valid email format
```

```
# Example 5: in with tuple
numbers = (1, 2, 3, 4, 5)
print(3 in numbers)
```

```
Output: True
```

```
# Example 6: not in with set
vowels = {'a', 'e', 'i', 'o', 'u'}
print('b' not in vowels)
```

```
Output: True
```

```
# Example 7: Dictionary key checking
student = {'name': 'John', 'age': 20}
print('name' in student)
```

```
Output: True
```

```
# Example 8: Case-sensitive string check
text = 'Hello World'
print('hello' in text)
```

```
Output: False
```

```
# Example 9: Check character in string
password = 'Pass123!'
has_special = '!' in password or '@' in password
print(f'Has special char: {has_special}')
```

```
Output: Has special char: True
```

```
# Example 10: Validation using in
valid_colors = ['red', 'green', 'blue']
user_color = 'yellow'
if user_color in valid_colors:
    print('Valid color')
else:
    print('Invalid color')
```

```
Output: Invalid color
```

```
# Example 11: Multiple membership checks
text = 'Python is awesome'
words = ['Python', 'Java', 'C++']
for word in words:
    if word in text:
        print(f'{word} found')
```

```
Output: Python found
```

```
# Example 12: Checking range
age = 25
if age in range(18, 60):
    print('Working age')
else:
    print('Not working age')
```

```
Output: Working age
```

```
# Example 13: Empty sequence check
my_list = []
if not my_list:
    print('List is empty')
```

```
Output: List is empty
```

```
# Example 14: File extension validation
filename = 'document.pdf'
valid_exts = ['.pdf', '.doc', '.txt']
if any(ext in filename for ext in valid_exts):
    print('Valid file type')
```

```
Output: Valid file type
```

```
# Example 15: Remove duplicates check
numbers = [1, 2, 3, 4, 5]
new_num = 3
if new_num not in numbers:
    numbers.append(new_num)
    print('Added')
else:
    print('Already exists')
```

```
Output: Already exists
```

7. If-Else Statements

If-else statements execute code blocks based on conditions. The structure includes: if (condition), elif (else if - multiple conditions), and else (when all conditions are false).

```
# Example 1: Simple if statement
age = 20
if age >= 18:
    print('You are an adult')
```

```
Output: You are an adult
```

```
# Example 2: if-else
num = 15
if num % 2 == 0:
    print('Even')
else:
    print('Odd')
```

```
Output: Odd
```

```
# Example 3: if-elif-else (grading)
marks = 85
if marks >= 90:
    grade = 'A'
elif marks >= 80:
    grade = 'B'
elif marks >= 70:
    grade = 'C'
else:
    grade = 'F'
print(f'Grade: {grade}')
```

```
Output: Grade: B
```

```
# Example 4: Nested if-else
num = 15
if num > 0:
    if num % 2 == 0:
        print('Positive Even')
    else:
        print('Positive Odd')
else:
    print('Negative or Zero')
```

```
Output: Positive Odd
```

```
# Example 5: Multiple conditions with and
username = 'admin'
password = 'pass123'
if username == 'admin' and password == 'pass123':
    print('Login Successful')
else:
    print('Invalid Credentials')
```

```
Output: Login Successful
```

```
# Example 6: Ternary operator
age = 17
status = 'Adult' if age >= 18 else 'Minor'
print(f'Status: {status}')
```

```
Output: Status: Minor
```

```
# Example 7: Leap year checker
year = 2024
if (year % 4 == 0 and year % 100 != 0) or (year % 400 == 0):
    print(f'{year} is a Leap Year')
else:
    print(f'{year} is not a Leap Year')
```

```
Output: 2024 is a Leap Year
```

```
# Example 8: Finding maximum of three
a, b, c = 15, 28, 22
if a >= b and a >= c:
    max_val = a
elif b >= c:
    max_val = b
else:
    max_val = c
```

```
print(f'Maximum: {max_val}')
```

```
Output: Maximum: 28
```

```
# Example 9: Triangle validity
a, b, c = 3, 4, 5
if a + b > c and b + c > a and c + a > b:
    print('Valid Triangle')
else:
    print('Invalid Triangle')
```

```
Output: Valid Triangle
```

```
# Example 10: Vowel or Consonant
char = 'a'
if char.lower() in 'aeiou':
    print(f'{char} is a Vowel')
else:
    print(f'{char} is a Consonant')
```

```
Output: a is a Vowel
```

```
# Example 11: Discount calculation
purchase = 1500
if purchase >= 2000:
    discount = 20
elif purchase >= 1000:
    discount = 10
else:
    discount = 0
final = purchase * (1 - discount/100)
print(f'Final Amount: {final}')
```

```
Output: Final Amount: 1350.0
```

```
# Example 12: BMI calculator
weight = 70  # kg
height = 1.75  # meters
bmi = weight / (height ** 2)
if bmi < 18.5:
    category = 'Underweight'
elif bmi < 25:
    category = 'Normal'
elif bmi < 30:
    category = 'Overweight'
else:
    category = 'Obese'
print(f'BMI: {bmi:.1f} - {category}')
```

```
Output: BMI: 22.9 - Normal
```

```
# Example 13: Number sign checker
num = -5
if num > 0:
    print('Positive')
elif num < 0:
    print('Negative')
else:
    print('Zero')
```

```
Output: Negative
```

```
# Example 14: Divisibility check
num = 15
if num % 3 == 0 and num % 5 == 0:
    print('Divisible by both 3 and 5')
elif num % 3 == 0:
    print('Divisible by 3 only')
elif num % 5 == 0:
    print('Divisible by 5 only')
else:
    print('Not divisible by 3 or 5')
```

```
Output: Divisible by both 3 and 5
```

```
# Example 15: Character type checker
ch = '5'
if ch.isalpha():
    print('Letter')
elif ch.isdigit():
    print('Digit')
elif ch.isspace():
    print('Space')
```

```
else:
    print('Special Character')
```

```
Output: Digit
```

8. Math Module

The math module provides mathematical functions and constants. It includes functions for trigonometry, logarithms, square roots, rounding, constants like pi and e, and more.

```
# Example 1: Square root
import math
num = 25
result = math.sqrt(num)
print(f'sqrt({num}) = {result}')
```

```
Output: sqrt(25) = 5.0
```

```
# Example 2: Power function
import math
base, exp = 2, 3
result = math.pow(base, exp)
print(f'{base}^{exp} = {result}')
```

```
Output: 2^3 = 8.0
```

```
# Example 3: Ceiling and floor
import math
num = 4.3
print(f'ceil({num}) = {math.ceil(num)}')
print(f'floor({num}) = {math.floor(num)}')
```

```
Output: ceil(4.3) = 5
floor(4.3) = 4
```

```
# Example 4: Factorial
import math
n = 5
result = math.factorial(n)
print(f'{n}! = {result}')
```

```
Output: 5! = 120
```

```
# Example 5: Pi constant
import math
radius = 5
area = math.pi * radius ** 2
print(f'Area of circle: {area:.2f}')
```

```
Output: Area of circle: 78.54
```

```
# Example 6: Trigonometric functions
import math
angle_deg = 45
angle_rad = math.radians(angle_deg)
print(f'sin(45°) = {math.sin(angle_rad):.4f}')
print(f'cos(45°) = {math.cos(angle_rad):.4f}')
```

```
Output: sin(45°) = 0.7071
cos(45°) = 0.7071
```

```
# Example 7: Logarithms
import math
num = 100
print(f'log10({num}) = {math.log10(num)}')
print(f'ln({num}) = {math.log(num):.4f}')
```

```
Output: log10(100) = 2.0
ln(100) = 4.6052
```

```
# Example 8: Absolute value
import math
num = -15.7
result = math.fabs(num)
print(f'|{num}| = {result}')
```

```
Output: |-15.7| = 15.7
```

```
# Example 9: GCD (Greatest Common Divisor)
import math
a, b = 48, 18
gcd = math.gcd(a, b)
print(f'GCD({a}, {b}) = {gcd}')
```

```
Output: GCD(48, 18) = 6
```

```
# Example 10: Hypotenuse
import math
a, b = 3, 4
c = math.hypot(a, b)
print(f'Hypotenuse: {c}')
```

```
Output: Hypotenuse: 5.0
```

```
# Example 11: Degrees to radians
import math
degrees = 180
radians = math.radians(degrees)
print(f'{degrees}° = {radians:.4f} radians')
```

```
Output: 180° = 3.1416 radians
```

```
# Example 12: Exponential function
import math
x = 2
result = math.exp(x)
print(f'e^{x} = {result:.4f}')
```

```
Output: e^2 = 7.3891
```

```
# Example 13: Check if number is finite
import math
num1 = 100
num2 = float('inf')
print(f'{num1} is finite: {math.isfinite(num1)}')
print(f'inf is finite: {math.isfinite(num2)}')
```

```
Output: 100 is finite: True
inf is finite: False
```

```
# Example 14: Truncate decimal
import math
num = 7.89
result = math.trunc(num)
print(f'trunc({num}) = {result}')
```

```
Output: trunc(7.89) = 7
```

```
# Example 15: Distance formula
import math
x1, y1 = 0, 0
x2, y2 = 3, 4
dist = math.sqrt((x2-x1)**2 + (y2-y1)**2)
print(f'Distance: {dist}')
```

```
Output: Distance: 5.0
```

9. Keyword Module

The keyword module provides functions to work with Python keywords. It helps check if a string is a Python keyword and provides a list of all keywords in the current Python version.

```
# Example 1: List all keywords
import keyword
keywords = keyword.kwlist
print(f'Total keywords: {len(keywords)}')
print(f'First 5: {keywords[:5]}')
```

```
Output: Total keywords: 35
First 5: ['False', 'None', 'True', 'and', 'as']
```

```
# Example 2: Check if string is keyword
import keyword
word1 = 'if'
word2 = 'hello'
print(f"'{word1}' is keyword: {keyword.iskeyword(word1)}")
print(f"'{word2}' is keyword: {keyword.iskeyword(word2)}")
```

```
Output: 'if' is keyword: True
'hello' is keyword: False
```

```
# Example 3: Validate variable name
import keyword
var_name = 'class'
if keyword.iskeyword(var_name):
    print(f"'{var_name}' cannot be used as variable")
else:
    print(f"'{var_name}' can be used")
```

```
Output: 'class' cannot be used as variable
```

```
# Example 4: Count control flow keywords
import keyword
control_keywords = ['if', 'elif', 'else', 'for', 'while']
count = sum(1 for k in control_keywords if k in keyword.kwlist)
print(f'Control flow keywords: {count}')
```

```
Output: Control flow keywords: 5
```

```
# Example 5: Display keywords in columns
import keyword
keywords = keyword.kwlist
for i in range(0, len(keywords), 5):
    print(', '.join(keywords[i:i+5]))
```

```
Output: False, None, True, and, as
assert, async, await, break, class
continue, def, del, elif, else
except, finally, for, from, global
if, import, in, is, lambda
nonlocal, not, or, pass, raise
return, try, while, with, yield
```

```
# Example 6: Check multiple names
import keyword
names = ['for', 'variable', 'def', 'my_var']
for name in names:
    status = 'keyword' if keyword.iskeyword(name) else 'valid'
    print(f'{name}: {status}')
```

```
Output: for: keyword
variable: valid
def: keyword
my_var: valid
```

```
# Example 7: Soft keywords check
import keyword
soft_kw = keyword.softkwlist
print(f'Soft keywords: {soft_kw}')
```

```
Output: Soft keywords: ['_', 'case', 'match', 'type']
```

```
# Example 8: Filter out keywords from list
import keyword
words = ['name', 'if', 'age', 'for', 'class']
valid = [w for w in words if not keyword.iskeyword(w)]
print(f'Valid names: {valid}')
```

```
Output: Valid names: ['name', 'age']
```

```
# Example 9: Keywords containing 'for'
import keyword
matches = [k for k in keyword.kwlist if 'for' in k]
print(f'Keywords with "for": {matches}')
```

```
Output: Keywords with "for": ['for']
```

```
# Example 10: Keywords starting with specific letter
import keyword
letter = 'a'
keywords_with_a = [k for k in keyword.kwlist if k.startswith(letter)]
print(f'Keywords starting with "{letter}": {keywords_with_a}')
```

```
Output: Keywords starting with "a": ['and', 'as', 'assert', 'async', 'await']
```

```
# Example 11: Boolean keywords
import keyword
bool_kw = [k for k in keyword.kwlist if k in ['True', 'False']]
print(f'Boolean keywords: {bool_kw}')
```

```
Output: Boolean keywords: ['False', 'True']
```

```
# Example 12: Exception-related keywords
import keyword
exc_kw = [k for k in keyword.kwlist if k in ['try', 'except', 'finally', 'raise']]
print(f'Exception keywords: {exc_kw}')
```

```
Output: Exception keywords: ['except', 'finally', 'raise', 'try']
```

```
# Example 13: Variable name validator function
import keyword
def is_valid_var_name(name):
    if keyword.iskeyword(name):
        return False
    if not name.isidentifier():
        return False
    return True

print(is_valid_var_name('my_var'))
print(is_valid_var_name('for'))
```

```
Output: True
False
```

```
# Example 14: Length of longest keyword
import keyword
longest = max(keyword.kwlist, key=len)
print(f'Longest keyword: {longest} ({len(longest)} chars)')
```

```
Output: Longest keyword: continue (8 chars)
```

```
# Example 15: Keywords vs identifiers comparison
import keyword
test_words = ['return', 'result', 'if', 'number']
for word in test_words:
    is_kw = keyword.iskeyword(word)
    is_id = word.isidentifier()
    print(f'{word}: keyword={is_kw}, identifier={is_id}')
```

```
Output: return: keyword=True, identifier=True
result: keyword=False, identifier=True
if: keyword=True, identifier=True
number: keyword=False, identifier=True
```

10. Random Module

The random module generates pseudo-random numbers for various purposes. It includes functions for random integers, floats, choices, shuffling, and sampling from sequences.

```
# Example 1: Random float [0.0, 1.0)
import random
random.seed(42)  # For reproducible output
num = random.random()
print(f'Random float: {num:.4f}')
```

```
Output: Random float: 0.6394
```

```
# Example 2: Random integer in range
import random
random.seed(42)
num = random.randint(1, 10)
print(f'Random int [1-10]: {num}')
```

```
Output: Random int [1-10]: 2
```

```
# Example 3: Random float in range
import random
random.seed(42)
num = random.uniform(10.0, 20.0)
print(f'Random float [10-20]: {num:.2f}')
```

```
Output: Random float [10-20]: 16.39
```

```
# Example 4: Random choice from list
import random
random.seed(42)
fruits = ['apple', 'banana', 'cherry', 'date']
choice = random.choice(fruits)
print(f'Random fruit: {choice}')
```

```
Output: Random fruit: cherry
```

```
# Example 5: Random sample (without replacement)
import random
random.seed(42)
numbers = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
sample = random.sample(numbers, 3)
print(f'Random sample: {sample}')
```

```
Output: Random sample: [9, 6, 4]
```

```
# Example 6: Shuffle a list
import random
random.seed(42)
cards = ['A', 'K', 'Q', 'J', '10']
random.shuffle(cards)
print(f'Shuffled: {cards}')
```

```
Output: Shuffled: ['10', 'Q', 'A', 'K', 'J']
```

```
# Example 7: Random choices with replacement
import random
random.seed(42)
colors = ['red', 'green', 'blue']
choices = random.choices(colors, k=5)
print(f'Choices: {choices}')
```

```
Output: Choices: ['green', 'blue', 'red', 'green', 'red']
```

```
# Example 8: Weighted random choice
import random
random.seed(42)
items = ['common', 'rare', 'legendary']
weights = [70, 25, 5]
choice = random.choices(items, weights=weights, k=1)[0]
print(f'Item: {choice}')
```

```
Output: Item: rare
```

```
# Example 9: Random boolean
import random
random.seed(42)
result = random.choice([True, False])
print(f'Random boolean: {result}')
```

```
Output: Random boolean: False
```

```
# Example 10: Dice roll simulation
import random
random.seed(42)
dice1 = random.randint(1, 6)
dice2 = random.randint(1, 6)
print(f'Dice: {dice1} and {dice2}, Sum: {dice1+dice2}')
```

```
Output: Dice: 2 and 1, Sum: 3
```

```
# Example 11: Random password generator
import random
import string
random.seed(42)
chars = string.ascii_letters + string.digits
password = ''.join(random.choices(chars, k=8))
print(f'Password: {password}')
```

```
Output: Password: YhQfLVJw
```

```
# Example 12: Random even number
import random
random.seed(42)
even = random.randrange(0, 101, 2)
print(f'Random even: {even}')
```

```
Output: Random even: 8
```

```
# Example 13: Lottery number generator
import random
random.seed(42)
numbers = sorted(random.sample(range(1, 50), 6))
print(f'Lottery numbers: {numbers}')
```

```
Output: Lottery numbers: [4, 6, 9, 13, 38, 39]
```

```
# Example 14: Random color (RGB)
import random
random.seed(42)
r = random.randint(0, 255)
g = random.randint(0, 255)
b = random.randint(0, 255)
print(f'RGB: ({r}, {g}, {b})')
```

```
Output: RGB: (163, 1, 119)
```

```
# Example 15: Coin flip simulation
import random
random.seed(42)
flips = [random.choice(['Heads', 'Tails']) for _ in range(5)]
heads = flips.count('Heads')
print(f'Flips: {flips}')
print(f'Heads: {heads}, Tails: {5-heads}')
```

```
Output: Flips: ['Tails', 'Tails', 'Heads', 'Tails', 'Heads']
Heads: 2, Tails: 3
```

11. Datetime Module

The datetime module provides classes for manipulating dates and times. It includes date, time, datetime, timedelta objects for working with dates, times, and time intervals.

```
# Example 1: Current date and time
import datetime
now = datetime.datetime.now()
print(f'Current: {now.strftime("%Y-%m-%d %H:%M:%S")}')
```

```
Output: Current: 2024-02-15 10:30:45
```

```
# Example 2: Current date only
import datetime
today = datetime.date.today()
print(f'Today: {today}')
```

```
Output: Today: 2024-02-15
```

```
# Example 3: Create specific date
import datetime
date = datetime.date(2024, 12, 25)
print(f'Christmas: {date}')
```

```
Output: Christmas: 2024-12-25
```

```
# Example 4: Extract date components
import datetime
today = datetime.date.today()
print(f'Year: {today.year}')
print(f'Month: {today.month}')
print(f'Day: {today.day}')
```

```
Output: Year: 2024
Month: 2
Day: 15
```

```
# Example 5: Day of week
import datetime
today = datetime.date.today()
day_name = today.strftime('%A')
print(f'Today is: {day_name}')
```

```
Output: Today is: Thursday
```

```
# Example 6: Add days to date
import datetime
today = datetime.date.today()
future = today + datetime.timedelta(days=7)
print(f'After 7 days: {future}')
```

```
Output: After 7 days: 2024-02-22
```

```
# Example 7: Calculate age
import datetime
birthdate = datetime.date(2000, 5, 15)
today = datetime.date.today()
age = today.year - birthdate.year
if today.month < birthdate.month or (today.month == birthdate.month and today.day < birthdate.day):
    age -= 1
print(f'Age: {age} years')
```

```
Output: Age: 23 years
```

```
# Example 8: Difference between dates
import datetime
date1 = datetime.date(2024, 1, 1)
date2 = datetime.date(2024, 12, 31)
diff = date2 - date1
print(f'Days between: {diff.days}')
```

```
Output: Days between: 365
```

```
# Example 9: Format date
import datetime
today = datetime.date.today()
formatted = today.strftime('%d/%m/%Y')
print(f'Formatted: {formatted}')
```

```
Output: Formatted: 15/02/2024
```

```
# Example 10: Parse string to date
import datetime
date_str = '2024-12-25'
date_obj = datetime.datetime.strptime(date_str, '%Y-%m-%d')
print(f'Parsed date: {date_obj.date()}')
```

```
Output: Parsed date: 2024-12-25
```

```
# Example 11: Time object
import datetime
time = datetime.time(14, 30, 45)
print(f'Time: {time}')
```

```
Output: Time: 14:30:45
```

```
# Example 12: Calculate time difference
import datetime
time1 = datetime.datetime(2024, 2, 15, 10, 0, 0)
time2 = datetime.datetime(2024, 2, 15, 15, 30, 0)
diff = time2 - time1
print(f'Time difference: {diff}')
```

```
Output: Time difference: 5:30:00
```

```
# Example 13: First day of month
import datetime
today = datetime.date.today()
first_day = today.replace(day=1)
print(f'First day of month: {first_day}')
```

```
Output: First day of month: 2024-02-01
```

```
# Example 14: Is leap year?
import datetime
year = 2024
is_leap = (year % 4 == 0 and year % 100 != 0) or (year % 400 == 0)
print(f'{year} is leap year: {is_leap}')
```

```
Output: 2024 is leap year: True
```

```
# Example 15: Days until birthday
import datetime
today = datetime.date.today()
birthday = datetime.date(today.year, 5, 15)
if birthday < today:
    birthday = birthday.replace(year=today.year + 1)
days = (birthday - today).days
print(f'Days until birthday: {days}')
```

```
Output: Days until birthday: 90
```

12. While Loop

A while loop repeatedly executes a block of code as long as a condition is True. It's useful when the number of iterations is not known beforehand. Must ensure condition becomes False to avoid infinite loops.

```
# Example 1: Basic while loop
count = 1
while count <= 5:
    print(count, end=' ')
    count += 1
```

```
Output: 1 2 3 4 5
```

```
# Example 2: Print even numbers
num = 2
while num <= 10:
    print(num, end=' ')
    num += 2
```

```
Output: 2 4 6 8 10
```

```
# Example 3: Sum of first n numbers
n = 5
total = 0
i = 1
while i <= n:
    total += i
    i += 1
print(f'Sum: {total}')
```

```
Output: Sum: 15
```

```
# Example 4: Countdown
count = 5
while count > 0:
    print(count, end=' ')
    count -= 1
print('Go!')
```

```
Output: 5 4 3 2 1 Go!
```

```
# Example 5: Factorial calculation
num = 5
factorial = 1
i = 1
while i <= num:
    factorial *= i
    i += 1
print(f'{num}! = {factorial}')
```

```
Output: 5! = 120
```

```
# Example 6: Reverse a number
num = 12345
reversed_num = 0
while num > 0:
    digit = num % 10
    reversed_num = reversed_num * 10 + digit
    num //= 10
print(f'Reversed: {reversed_num}')
```

```
Output: Reversed: 54321
```

```
# Example 7: Count digits
num = 12345
count = 0
while num > 0:
    count += 1
    num //= 10
print(f'Digits: {count}')
```

```
Output: Digits: 5
```

```
# Example 8: Fibonacci series
a, b = 0, 1
count = 0
print('Fibonacci:', end=' ')
while count < 7:
    print(a, end=' ')
    a, b = b, a + b
    count += 1
```

```
Output: Fibonacci: 0 1 1 2 3 5 8
```

```
# Example 9: Find GCD
a, b = 48, 18
while b != 0:
    a, b = b, a % b
print(f'GCD: {a}')
```

```
Output: GCD: 6
```

```
# Example 10: Power calculation
base, exp = 2, 5
result = 1
while exp > 0:
    result *= base
    exp -= 1
print(f'Result: {result}')
```

```
Output: Result: 32
```

```
# Example 11: Check palindrome number
num = 121
original = num
reversed_num = 0
while num > 0:
    reversed_num = reversed_num * 10 + num % 10
    num //= 10
if original == reversed_num:
    print('Palindrome')
else:
    print('Not Palindrome')
```

```
Output: Palindrome
```

```
# Example 12: Sum of digits
num = 12345
total = 0
while num > 0:
    total += num % 10
    num //= 10
print(f'Sum of digits: {total}')
```

```
Output: Sum of digits: 15
```

```
# Example 13: Menu-driven program
choice = 1
while choice != 0:
    print('1. Option 1')
    print('0. Exit')
    choice = 0  # Simulating user input
    if choice == 1:
        print('Option 1 selected')
print('Exited')
```

```
Output: 1. Option 1
0. Exit
Exited
```

```
# Example 14: Prime number check
num = 17
i = 2
is_prime = True
while i <= num // 2:
    if num % i == 0:
        is_prime = False
        break
    i += 1
print(f'{num} is prime: {is_prime}')
```

```
Output: 17 is prime: True
```

```
# Example 15: Multiplication table
num = 5
i = 1
while i <= 5:
    print(f'{num} x {i} = {num * i}')
    i += 1
```

```
Output: 5 x 1 = 5
5 x 2 = 10
5 x 3 = 15
5 x 4 = 20
5 x 5 = 25
```

13. While Loop with Else

Python's while loop can have an optional else clause that executes when the loop completes normally (not interrupted by break). The else block doesn't execute if the loop is terminated by break.

```
# Example 1: Basic while-else
count = 1
while count <= 3:
    print(count, end=' ')
    count += 1
else:
    print('\nLoop completed')
```

```
Output: 1 2 3
Loop completed
```

```
# Example 2: Search in list (found)
numbers = [1, 2, 3, 4, 5]
target = 3
i = 0
while i < len(numbers):
    if numbers[i] == target:
        print(f'Found {target} at index {i}')
        break
    i += 1
else:
    print(f'{target} not found')
```

```
Output: Found 3 at index 2
```

```
# Example 3: Search in list (not found)
numbers = [1, 2, 3, 4, 5]
target = 10
i = 0
while i < len(numbers):
    if numbers[i] == target:
        print(f'Found {target}')
        break
    i += 1
else:
    print(f'{target} not found')
```

```
Output: 10 not found
```

```
# Example 4: Password validation
attempts = 0
max_attempts = 3
while attempts < max_attempts:
    password = 'wrong'  # Simulating input
    if password == 'correct':
        print('Access granted')
        break
    attempts += 1
else:
    print('Too many attempts')
```

```
Output: Too many attempts
```

```
# Example 5: Prime number check
num = 29
i = 2
while i <= num // 2:
    if num % i == 0:
        print(f'{num} is not prime')
        break
    i += 1
else:
    print(f'{num} is prime')
```

```
Output: 29 is prime
```

```
# Example 6: Process until condition
count = 0
while count < 5:
    count += 1
    if count == 3:
        continue
    print(count, end=' ')
else:
    print('\nAll processed')
```

```
Output: 1 2 4 5
All processed
```

```
# Example 7: No break - else executes
i = 1
while i <= 3:
    print(f'Iteration {i}')
    i += 1
else:
    print('Finished normally')
```

```
Output: Iteration 1
Iteration 2
Iteration 3
Finished normally
```

```
# Example 8: With break - else doesn't execute
i = 1
while i <= 5:
    if i == 3:
        print('Breaking at 3')
        break
    i += 1
else:
    print('This will not print')
```

```
Output: Breaking at 3
```

```
# Example 9: Sum until threshold
total = 0
num = 1
while total < 10:
    total += num
    num += 1
else:
    print(f'Total reached: {total}')
```

```
Output: Total reached: 10
```

```
# Example 10: Validate all elements
numbers = [2, 4, 6, 8]
i = 0
while i < len(numbers):
    if numbers[i] % 2 != 0:
        print('Odd number found')
        break
    i += 1
else:
    print('All numbers are even')
```

```
Output: All numbers are even
```

```
# Example 11: Count down with else
count = 3
while count > 0:
    print(count, end=' ')
    count -= 1
else:
    print('\nLiftoff!')
```

```
Output: 3 2 1
Liftoff!
```

```
# Example 12: String character search
text = 'Python'
char = 'x'
i = 0
while i < len(text):
    if text[i] == char:
        print(f"'{char}' found at index {i}")
        break
    i += 1
else:
    print(f"'{char}' not in string")
```

```
Output: 'x' not in string
```

```
# Example 13: Process list elements
items = ['a', 'b', 'c']
i = 0
while i < len(items):
    print(items[i], end=' ')
    i += 1
else:
```

```
    print('\nAll items processed')
```

```
Output: a b c
All items processed
```

```
# Example 14: Number guessing game
secret = 7
guess = 5
attempts = 0
while attempts < 3:
    if guess == secret:
        print('Correct!')
        break
    attempts += 1
    guess += 1
else:
    print('Out of attempts')
```

```
Output: Correct!
```

```
# Example 15: Factorial with while-else
num = 4
factorial = 1
i = 1
while i <= num:
    factorial *= i
    i += 1
else:
    print(f'{num}! = {factorial}')
```

```
Output: 4! = 24
```

14. For Loop

For loops iterate over sequences (lists, tuples, strings, range objects). They're preferred when the number of iterations is known. Python's for loop is more like a 'foreach' loop in other languages.

```
# Example 1: Iterate over range
for i in range(1, 6):
    print(i, end=' ')
```

```
Output: 1 2 3 4 5
```

```
# Example 2: Iterate over list
fruits = ['apple', 'banana', 'cherry']
for fruit in fruits:
    print(fruit, end=' ')
```

```
Output: apple banana cherry
```

```
# Example 3: Iterate over string
text = 'Python'
for char in text:
    print(char, end=' ')
```

```
Output: P y t h o n
```

```
# Example 4: Range with step
for i in range(0, 11, 2):
    print(i, end=' ')
```

```
Output: 0 2 4 6 8 10
```

```
# Example 5: Sum of list elements
numbers = [10, 20, 30, 40, 50]
total = 0
for num in numbers:
    total += num
print(f'Sum: {total}')
```

```
Output: Sum: 150
```

```
# Example 6: Multiplication table
num = 7
for i in range(1, 6):
    print(f'{num} x {i} = {num * i}')
```

```
Output: 7 x 1 = 7
7 x 2 = 14
7 x 3 = 21
7 x 4 = 28
7 x 5 = 35
```

```
# Example 7: Enumerate for index and value
fruits = ['apple', 'banana', 'cherry']
for index, fruit in enumerate(fruits):
    print(f'{index}: {fruit}')
```

```
Output: 0: apple
1: banana
2: cherry
```

```
# Example 8: Iterate over dictionary
student = {'name': 'John', 'age': 20, 'grade': 'A'}
for key, value in student.items():
    print(f'{key}: {value}')
```

```
Output: name: John
age: 20
grade: A
```

```
# Example 9: Count vowels in string
text = 'Python Programming'
vowels = 'aeiouAEIOU'
count = 0
for char in text:
    if char in vowels:
        count += 1
print(f'Vowels: {count}')
```

```
Output: Vowels: 4
```

```
# Example 10: List comprehension alternative
squares = [x**2 for x in range(1, 6)]
```

```
print(squares)
```

```
Output: [1, 4, 9, 16, 25]
```

```
# Example 11: Find maximum in list
numbers = [15, 28, 22, 35, 19]
max_num = numbers[0]
for num in numbers:
    if num > max_num:
        max_num = num
print(f'Maximum: {max_num}')
```

```
Output: Maximum: 35
```

```
# Example 12: Reverse a string
text = 'Python'
reversed_text = ''
for char in text:
    reversed_text = char + reversed_text
print(f'Reversed: {reversed_text}')
```

```
Output: Reversed: nohtyP
```

```
# Example 13: Pattern printing
for i in range(1, 6):
    print('*' * i)
```

```
Output: *
**
***
****
*****
```

```
# Example 14: Skip even numbers
for i in range(1, 11):
    if i % 2 == 0:
        continue
    print(i, end=' ')
```

```
Output: 1 3 5 7 9
```

```
# Example 15: Find in list and break
numbers = [5, 10, 15, 20, 25]
target = 15
for i, num in enumerate(numbers):
    if num == target:
        print(f'Found {target} at index {i}')
        break
```

```
Output: Found 15 at index 2
```

15. Nested Loops

Nested loops are loops inside loops. The inner loop executes completely for each iteration of the outer loop. Common in matrix operations, pattern printing, and multi-dimensional data processing.

```
# Example 1: Simple nested loop
for i in range(1, 4):
    for j in range(1, 4):
        print(f'({i},{j})', end=' ')
    print()
```

```
Output: (1,1) (1,2) (1,3)
(2,1) (2,2) (2,3)
(3,1) (3,2) (3,3)
```

```
# Example 2: Multiplication table
for i in range(1, 4):
    for j in range(1, 4):
        print(f'{i*j:3}', end=' ')
    print()
```

```
Output:   1   2   3
  2   4   6
  3   6   9
```

```
# Example 3: Right triangle pattern
for i in range(1, 6):
    for j in range(1, i+1):
        print('*', end='')
    print()
```

```
Output: *
**
***
****
*****
```

```
# Example 4: Number pyramid
for i in range(1, 5):
    for j in range(1, i+1):
        print(j, end='')
    print()
```

```
Output: 1
12
123
1234
```

```
# Example 5: Matrix operations
matrix = [[1, 2, 3], [4, 5, 6], [7, 8, 9]]
for row in matrix:
    for elem in row:
        print(f'{elem:2}', end=' ')
    print()
```

```
Output:  1  2  3
 4  5  6
 7  8  9
```

```
# Example 6: Sum of 2D list
matrix = [[1, 2, 3], [4, 5, 6]]
total = 0
for row in matrix:
    for num in row:
        total += num
print(f'Sum: {total}')
```

```
Output: Sum: 21
```

```
# Example 7: Print pairs from two lists
list1 = ['A', 'B']
list2 = [1, 2, 3]
for letter in list1:
    for num in list2:
        print(f'{letter}{num}', end=' ')
print()
```

```
Output: A1 A2 A3 B1 B2 B3
```

```
# Example 8: Search in 2D list
matrix = [[1, 2, 3], [4, 5, 6], [7, 8, 9]]
```

```
target = 5
for i in range(len(matrix)):
    for j in range(len(matrix[i])):
        if matrix[i][j] == target:
            print(f'Found at ({i},{j})')
```

```
Output: Found at (1,1)
```

```
# Example 9: Inverted triangle
for i in range(5, 0, -1):
    for j in range(i):
        print('*', end='')
    print()
```

```
Output: *****
****
***
**
*
```

```
# Example 10: Floyd's triangle
num = 1
for i in range(1, 5):
    for j in range(i):
        print(num, end=' ')
        num += 1
    print()
```

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

```
# Example 11: Diamond pattern
n = 3
for i in range(n):
    print(' ' * (n-i-1) + '*' * (2*i+1))
for i in range(n-2, -1, -1):
    print(' ' * (n-i-1) + '*' * (2*i+1))
```

```
Output:   *
 ***
*****
 ***
  *
```

```
# Example 12: Print all combinations
for i in range(1, 4):
    for j in range(1, 4):
        if i != j:
            print(f'{i}{j}', end=' ')
```

```
Output: 12 13 21 23 31 32
```

```
# Example 13: Transpose matrix
matrix = [[1, 2, 3], [4, 5, 6]]
transpose = []
for j in range(len(matrix[0])):
    row = []
    for i in range(len(matrix)):
        row.append(matrix[i][j])
    transpose.append(row)
for row in transpose:
    print(row)
```

```
Output: [1, 4]
[2, 5]
[3, 6]
```

```
# Example 14: Nested loop with break
for i in range(1, 4):
    for j in range(1, 4):
        if i * j > 4:
            break
        print(f'{i}*{j}={i*j}', end=' ')
    print()
```

```
Output: 1*1=1 1*2=2 1*3=3
2*1=2 2*2=4
3*1=3
```

```
# Example 15: Prime numbers in range
for num in range(2, 20):
```

```
    is_prime = True
    for i in range(2, int(num**0.5) + 1):
        if num % i == 0:
            is_prime = False
            break
    if is_prime:
        print(num, end=' ')
```

```
Output: 2 3 5 7 11 13 17 19
```