Interview Preparation Notes

# 🔹 Bitwise Operators in Python

Bitwise operators work at the binary level (bit by bit operations). They are mostly used in low-level programming, optimization, cryptography, masks, flags, etc.

## Operators:

AND (&): Sets each bit to 1 if both bits are 1.

a = 5 # 0101  
b = 3 # 0011  
print(a & b) # 1 (0001)

OR (|): Sets each bit to 1 if at least one is 1.

print(a | b) # 7 (0111)

XOR (^): Sets each bit to 1 if only one of the bits is 1.

print(a ^ b) # 6 (0110)

NOT (~): Flips all bits → ~n = -(n+1)

print(~a) # -6 (in 2’s complement)

Left Shift (<<): Shifts bits left, fills with 0 → equivalent to multiplying by 2^n.

print(a << 1) # 10 (1010)

Right Shift (>>): Shifts bits right, drops least significant bits → equivalent to integer division by 2^n.

print(a >> 1) # 2 (0010)

## Quick Interview Tips:

• x & 1 → checks if a number is odd (1 if odd, 0 if even).

• Swapping without a temp variable:

x, y = 5, 10  
x = x ^ y  
y = x ^ y  
x = x ^ y  
print(x, y) # 10, 5

# 🔹 Nested If Statements

Definition: An if inside another if. Used when you need multiple levels of decision-making.

## Syntax:

if condition1:  
 if condition2:  
 # code block if both are true  
 else:  
 # code block if only condition1 is true  
else:  
 # code block if condition1 is false

## Example 1: Student grading

marks = 85  
if marks >= 50:  
 if marks >= 90:  
 print("Grade A")  
 elif marks >= 75:  
 print("Grade B")  
 else:  
 print("Grade C")  
else:  
 print("Fail")

## Example 2: Number classification

num = -7  
if num >= 0:  
 if num == 0:  
 print("Zero")  
 else:  
 print("Positive")  
else:  
 print("Negative")

## Quick Interview Notes:

• Nested if helps handle multiple conditions hierarchically.

• Too many nested ifs can reduce readability → consider elif or logical operators (and, or).

• Often tested with range checks or classification problems in interviews.