C++ Logical & Bitwise Operators – Concise Notes

1. Two Categories of Operators
Bitwise Operators: Work on individual bits of integers.
- &, |, ^, ~

Logical Operators: Work on truth values (true/false).
- &&, ||, !

2. Bitwise Operators
& → AND (bit-by-bit)
| → OR (bit-by-bit)
^ → XOR (1 if bits differ)
~ → NOT (flips every bit, including sign bit)

3. Logical Operators
&& → Logical AND (short-circuit)
|| → Logical OR (short-circuit)
! → Logical NOT
Note: Logical operators return only 0 or 1.

4. Key Difference
a & b → Bitwise operation on bits
a && b → Logical operation on truth values
Logical operators may skip evaluation (short-circuit).

5. Logic Gates Using Operators
AND → a & b / a && b
OR → a | b / a || b
XOR → a ^ b
NOT → ~a / !a
NAND → ~(a & b) / !(a && b)
NOR → ~(a | b) / !(a || b)
XNOR → ~(a ^ b)

6. Sign Bit
The leftmost bit in a signed integer.
0 → positive number
1 → negative number

7. Two's Complement Representation
Used by almost all modern systems.
Rule: ~x = -x - 1
Negative numbers are represented by flipping bits and adding 1.

8. Why ~ Changes Sign
~ flips all bits including the sign bit.
Result depends on integer size (8-bit, 32-bit, etc.).

9. Important Takeaways
- Use bitwise operators for low-level bit manipulation.
- Use logical operators for conditions and control flow.
- Never confuse & with && or | with ||.
- Sign bit flip is a side effect, not intent.