Bit Manipulation Complete Cheat Sheet

1. Basic Bitwise Operators

AND (&) → 1 if both bits are 1
OR (|) → 1 if at least one bit is 1
XOR (^) → 1 if bits are different
NOT (~) → flips bits
Left Shift (<<) → shifts bits left
Right Shift (>>) → shifts bits right

2. Common Bit Tricks

Get i-th bit: (n >> i) & 1
Set i-th bit: n | (1 << i)
Clear i-th bit: n & ~(1 << i)
Toggle i-th bit: n ^ (1 << i)
Remove rightmost set bit: n & (n - 1)
Isolate rightmost set bit: n & (-n)
Count set bits: while (n) { n &= (n-1); count++; }
Check power of 2: (n > 0 && (n & (n - 1)) == 0)
Swap without temp: a^=b; b^=a; a^=b;

3. XOR Properties

a ^ 0 = a
a ^ a = 0
a ^ b ^ a = b
Used in unique element problems and parity checks.

4. Useful Built-in Functions (C++)

__builtin_popcount(x) → count set bits
__builtin_clz(x) → count leading zeros
__builtin_ctz(x) → count trailing zeros
__builtin_parity(x) → 1 if odd no. of bits set

5. Advanced Techniques

Power set generation: for(mask=0;mask<(1<Brian Kernighan's Algorithm for counting bits
Reverse bits, Mask DP, Subset sum, Parity

6. Mathematical Uses

Multiply by 2: n << 1
Divide by 2: n >> 1
Even/Odd check: n & 1
Modulo power of 2: n & (m - 1)
Find log2(n): use __builtin_clz()

7. Key Interview Problems

Single Number (Leetcode 136): XOR all numbers
Missing Number: XOR 1..n and array elements
Power of Two: (n & (n-1))==0
Counting Bits (Leetcode 338): dp[i] = dp[i&(i-1)] + 1
Max XOR Pair: Trie + XOR