# ■ Complexity & Data Types Cheat Sheet

### Time Complexity by Input Size

■ If **n** ≤ **$10^3$ (1000)**
- O($n^2$) or even O($n^3$) may be acceptable.
- Example: brute force, nested loops.

■ If **n** ≤ **10$\overset{4}{\blacksquare}$ (10,000)**
- O($n^2$) might still pass (~10■ ops, borderline).
- Prefer O(n log n) or better.

■ If **n** ≤ **10$\overset{5}{\blacksquare}$ (100,000)**
- O($n^2$) is too slow (~$10^1$■ ops).
- You need O(n log n) or faster.

■ If **n** ≤ **10$\overset{6}{\blacksquare}$ (1 million)**
- O(n log n) is OK.
- O($n^2$) is impossible.

■ If **n** ≤ **10$\overset{8}{\blacksquare}$ (100 million)**
- Only O(n) or O(log n) will run.

■ If **n ~ 10$\overset{9}{\blacksquare}$ or more**
- Only O(log n), O(1), or math-based solutions are possible.

### Data Types Cheat Sheet

- -$2^{31}$ to $2^{31}$-1 → int in Java (32-bit)
- -2■$^3$ to 2■$^3$-1 → long in Java (64-bit)
- Bigger than that → BigInteger

**■ Quick Rules:**
1. Check max n → decide algorithm complexity.
2. Check value ranges → choose correct data type.
3. Think scalability → avoid nested loops for n ≥ 10■. 6