Final choice:

```java
Map<Integer, Integer> map = new HashMap<>();
```

Copy code

8️⃣ Golden 5-step checklist (use EVERY problem)

When you open a LeetCode problem:

1. 🔍 **Max input size?**
2. ⏱️ **Allowed time complexity?**
3. 🧠 **Best data structure?**
4. 🔢 **int or long?**
5. ⚠️ **Edge cases from constraints?**

If you answer these, **80% of the problem is already solved**.

9️⃣ Want to practice this skill?

If you want, I can:

- Walk through **1 real LeetCode problem** step by step
- Teach **constraint → algorithm patterns**
- Give you **constraint decoding exercises**
- Show **Java-specific pitfalls**

Just tell me 👍

Teach constraint → algorithm patterns

Perfect—this is **the skill that separates beginners from strong LeetCode solvers**.
I'll give you a **clear constraint → algorithm pattern map**, with **Java-specific thinking** and **mental shortcuts** you can apply instantly.

🔥 Constraint → Algorithm Patterns (LeetCode Master Map)

1️⃣ `n ≤ 20` → Brute Force / Backtracking

Typical constraints

```
1 ≤ n ≤ 20
```

Copy code

What it screams

- Exponential is OK
- Try **all subsets / permutations**

Algorithms

- Backtracking
- Bitmasking (`1 << n`)
- DFS

Example problems

- Subsets
- Permutations
- Combination Sum

Java hint

java

```
for (int mask = 0; mask < (1 << n); mask++)
```

Copy code

2️⃣ `n ≤ 100` → $O(n^3)$ / DP

Typical constraints

```
1 ≤ n ≤ 100
```

Copy code

What it screams

- DP with 2D/3D arrays
- Floyd–Warshall
- Interval DP

Algorithms

- Dynamic Programming
- Matrix DP
- Graph all-pairs

Example problems

- Longest Palindromic Subsequence
- Matrix Chain Multiplication

3️⃣ `n ≤ 1,000` → $O(n^2)$

Typical constraints

```
1 ≤ n ≤ 10^3
```

Copy code

What it screams

- Nested loops are fine
- Greedy + DP

Algorithms

- LIS ($O(n^2)$)
- Two pointers
- Sliding window (worst case)

Java hint

```java
for (int i = 0; i < n; i++)
  for (int j = 0; j < i; j++)
```

Copy code

4️⃣ `n ≤ 10^5` → O(n log n) / O(n) (MOST COMMON)

Typical constraints

```
1 ≤ n ≤ 10^5
```

Copy code

What it screams

🚨 **NO double loops**

Algorithms

- Sorting
- HashMap / HashSet
- Two pointers
- Sliding Window
- Prefix Sum
- Binary Search
- Greedy

Example problems

- Two Sum
- Subarray Sum Equals K
- Merge Intervals
- Top K Elements

Java tools

```java
Arrays.sort(arr);
HashMap<Integer, Integer>
```

```
PriorityQueue<Integer>
```

Copy code

---

5 `n ≤ 10^6` → Strict O(n)

Typical constraints

```
1 ≤ n ≤ 10^6
```

Copy code

What it screams

- Single pass only
- No sorting
- Arrays preferred over HashMap

Algorithms

- Counting
- Prefix arrays
- Greedy
- Kadane's Algorithm

---

6 Very large values (`nums[i] ≤ 10^9`) → Coordinate tricks

Typical constraints

```
css

-10^9 ≤ nums[i] ≤ 10^9
```

Copy code

What it screams

- No direct indexing
- HashMap or sorting
- Coordinate compression

7️⃣ Range queries → Prefix Sum / Fenwick / Segment Tree

Typical constraints

```css
Q ≤ 10^5
```

Copy code

Patterns

Query type	Algorithm
Static sum	Prefix Sum
Update + sum	Fenwick Tree
Min/Max	Segment Tree

8️⃣ Subarray / substring problems → Sliding Window

Typical phrases

- “longest”
- “at most k”
- “continuous”
- “subarray”

Constraint clue

```nginx
n ≤ 10^5
```

Copy code

Algorithm

- Sliding Window (two pointers)

Example

- Longest Substring Without Repeating Characters

9️⃣ Graph problems → n + edges decide

Typical constraints

```
1 ≤ n ≤ 10^5
1 ≤ edges ≤ 10^5
```

Copy code

Decision table

Graph type	Algorithm
Unweighted	BFS
Weighted	Dijkstra
DAG	Topological Sort
Cycle check	DFS / Union Find

🔟 DP + constraints → state count rule

Rule

```mathematica
Total states × transitions ≤ 10^7
```

Copy code

Example

```css
dp[i][j] where n ≤ 1000
```

Copy code

❌ `1000 × 1000 = 10^6` states is OK

❌ 3D DP → usually TLE

1️⃣ Common Constraint → Algorithm Cheat Sheet

Constraint	Use
`n ≤ 20`	Backtracking
`n ≤ 100`	DP
`n ≤ 1000`	$O(n^2)$
`n ≤ 10^5`	HashMap / Sliding Window
Large values	Map / Sort
Many queries	Prefix / Tree
"k-th / top k"	Heap
"minimum / maximum"	Greedy

🧠 The "Pattern Sentence" Trick

Convert constraints into a sentence:

> "n is 10^5, values are large, order matters"
> → **O(n log n)**
> → **Sorting + two pointers**

This becomes automatic with practice.

🚀 Your next step (important)

If you want, I can:

1. Break down **10 common LeetCode patterns** with problems
2. Give **constraint decoding drills**
3. Teach **Java performance traps**
4. Do **live constraint → algorithm conversion**

Just tell me which one you want 🔥