

Memory Management in Java

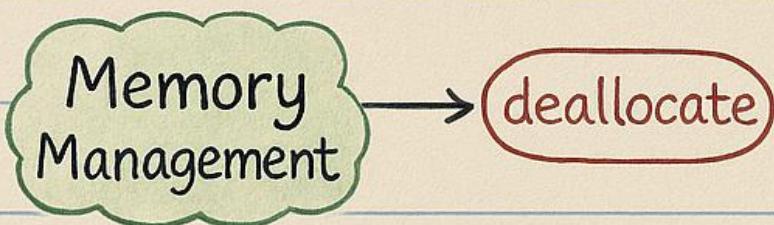
**Complete Beginner
to Advanced Notes**

Page No.	Topic
Cover Page	Cover Page
1	Title / Submission Page
2	Introduction to Java Memory Management
3	Memory Management: Java vs C/C++
4	JVM Architecture Overview
5	Components of JVM (ClassLoader, Execution Engine, JNI, etc.)
6	Java Runtime Data Areas – Overview
7	Method Area / Metaspace
8	Heap Memory (Young & Old Generations)
9	Stack Memory (Frames, Variables, Lifespan)
10	Stack vs Heap – Detailed Comparison (Diagram)
11	Program Counter Register & Native Method Stack
12	Object Creation & Memory Allocation Process
13	Garbage Collection – Introduction & Fundamentals
14	Object Reachability & Reference Types (Strong, Soft, Weak)
15	Garbage Collection Types (Minor, Major, Full GC)
16	Generational Garbage Collection (Eden, Survivor Spaces)
17	Garbage Collection Algorithms (Mark-Sweep, Copy, Compact)
18	HotSpot Garbage Collectors (Serial, Parallel, CMS)
19	Modern Collectors (G1, ZGC, Shenandoah)
20	GC Performance Concepts (Pause Time, Latency, Throughput)
21	JVM Memory Tuning & JVM Parameters (-Xms, -Xmx, etc.)
22	Memory Leak – Causes, Detection, and Prevention
23	Java Memory Monitoring Tools (jconsole, jvisualvm, jmap, jstat)
24	Best Practices for Efficient Memory Usage
25	Optimization Techniques – Examples & Summary
26	One-Page Revision Cheat Sheet (full mind map style)
27	Important MCQ Questions
28	Conclusion

→ Introduction to Java Memory Management

What is Memory Management?

- Allocation of memory to objects
- deallocation of memory from objects



Why Java Memory Management is important?

- Efficient memory utilization
- Performance improvement
- Automatic garbage collection

In Java, memory management is the process of managing the memory used and available by a Java program.

→ Key Components

- JVM (Java Virtual Machine)
- Runtime Data Areas
- Garbage Collector

Example:

```
int[] numbers = new int[5];
```