Let’s break down wait(), notify(), and notifyAll() in **Java multithreading** — these are core synchronization tools tied to the **Object class**, commonly used for inter-thread communication.

**🔸 What Are They?**

**🧵 wait()**

* **Used by a thread to pause execution** and release the **lock** on the object.
* The thread **waits** until another thread calls notify() or notifyAll() on the same object.

**🧵 notify()**

* **Wakes up one** of the threads that are waiting on that object's monitor.

**🧵 notifyAll()**

* **Wakes up all** threads that are waiting on that object's monitor.

**🔐 Must Be Called Inside a synchronized Block**

If not, you’ll get:

IllegalMonitorStateException

**🔁 Life Cycle Example: Producer-Consumer (Simplified)**

class Store {

int product = 0;

public synchronized void produce() throws InterruptedException {

while (product >= 1) {

wait(); // wait until consumer consumes

}

product++;

System.out.println("Produced: " + product);

notify(); // notify the consumer

}

public synchronized void consume() throws InterruptedException {

while (product == 0) {

wait(); // wait until producer produces

}

System.out.println("Consumed: " + product);

product--;

notify(); // notify the producer

}

}

**👷 Usage:**

public class Main {

public static void main(String[] args) {

Store store = new Store();

Thread producer = new Thread(() -> {

try {

for (int i = 0; i < 5; i++) store.produce();

} catch (InterruptedException e) {}

});

Thread consumer = new Thread(() -> {

try {

for (int i = 0; i < 5; i++) store.consume();

} catch (InterruptedException e) {}

});

producer.start();

consumer.start();

}

}

**⚠️ Common Points to Remember**

| **Concept** | **Details** |
| --- | --- |
| 🔐 Lock required? | Yes, use inside synchronized block or method. |
| 🧘‍♂️ wait() | Releases lock, goes to waiting state. |
| 🧨 notify() | Wakes up **one** waiting thread. |
| 🌍 notifyAll() | Wakes up **all** waiting threads (use when multiple consumers/producers). |
| ❗ Loop with while | Always use while(condition) instead of if for re-check after wake-up. |

**✅ Interview Tip:**

* wait() is like **“pause & release the monitor”**.
* notify() is like **“you can continue now”**, but **only one thread** gets that message.
* notifyAll() is like **“everyone wake up”**, but only one will acquire the lock first.

Let me know if you want a visual representation or comparison with Condition.await()/signal() from java.util.concurrent.locks.