**10/07/2025:**

**Multi-threading:**

What is thread?

A thread is the smallest unit of execution in Java program. Java supports multi-threading at the language level using:

1. Thread class.
2. Runnable interface.
3. Executer Service.
4. Advance Thread Maanagement.

**Multi-threading:**

Multi-threading is the ability of a program to run multiple threads(mini program) simultaneously to make better use of CPU resources.

Problem statement:  
Print the numbers using two threads.

Write a java program where, one thread prints even numbers from 1 to 10. Another thread prints odd numbers from 1-10. The program should print numbers in any order(since threads may run independently), but both even and odd numbers must appear.

Our objective in this problem is:

Create two threads, Use thread class or runnable interface, understand how multiple threads run in parallel.

The concept which we are going to cover in this problem is:

1. Thread creation
2. Parallel execution
3. Independent task execution
4. No deterministic output(Because the order of number may vary in each run)

How to create a thread in java?

There are three main ways to create a thread:

1. Extending the thread class.

Create a class that extends thread.

1. Override the run method.

Create an object of that class and call start()(not run).

Problem 1: Producer-Consumer with Bounded Buffer

Concepts: Thread Communication, wait(), notify(), Synchronization

Problem:

Implement a bounded buffer where multiple producer threads produce data and multiple consumer threads consume it. Ensure thread-safe communication using wait() and notify().

Challenges:

Avoid deadlock and race conditions.

Limit buffer capacity.

Problem 1: Producer-Consumer with Bounded

Imagine this: A Donut Shop!

**Implementing the runnable interface:**

This is the preferred way in real projects.

1. Create a class that implements runnable.
2. Override the run method.
3. Pass the object to a thread constructor.
4. Call start.

**Using lambda expression(JAVA 8+):**

This is the quick and elegant way if the logic is short.

**What is bounded buffer?**

A bounded buffer is a fixed size data structure like a box, queue or shelf that is shared between one or more producer thread(that add items to the buffer).

One or more consumer thread(that removes item from the buffer)

**Problem statement:**

Problem Statement

**> Create a thread that acts like a timer, printing the current seconds count every 1 second, up to 10 seconds.**

**Objective**

**Learn how to:**

**Create a thread**

**Use sleep() to pause thread execution**

**Run a task repeatedly at timed intervals.**

**Steps:**

**Step**

**Description**

**Create a class that extends Thread or implements Runnable**

**Inside run(), use a loop to print second count from 1 to 10**

**After each print, pause the thread for 1 second using Thread.sleep(1000)**

**In main(), start the thread and observe the timer-like behavior**

**Problem statement:**

**Implement a class Timer that simulates a simple timer using Java multithreading.**

**The constructor initializes the total duration n (in seconds).**

**The method startTimer() launches a separate thread that:**

**Prints "Seconds passed: X" every second for X = 1, 2, ... n**

**After reaching n, prints "Timer completed!"**

**Your implementation must use a separate Thread and Thread.sleep(1000) to simulate the delay. Do not use built-in schedulers (like java.util.Timer or ScheduledExecutorService).**

**Add start, pause,resume and stop.**

**startTimer,pauseTimer, resumeTimer, stopTImer.**