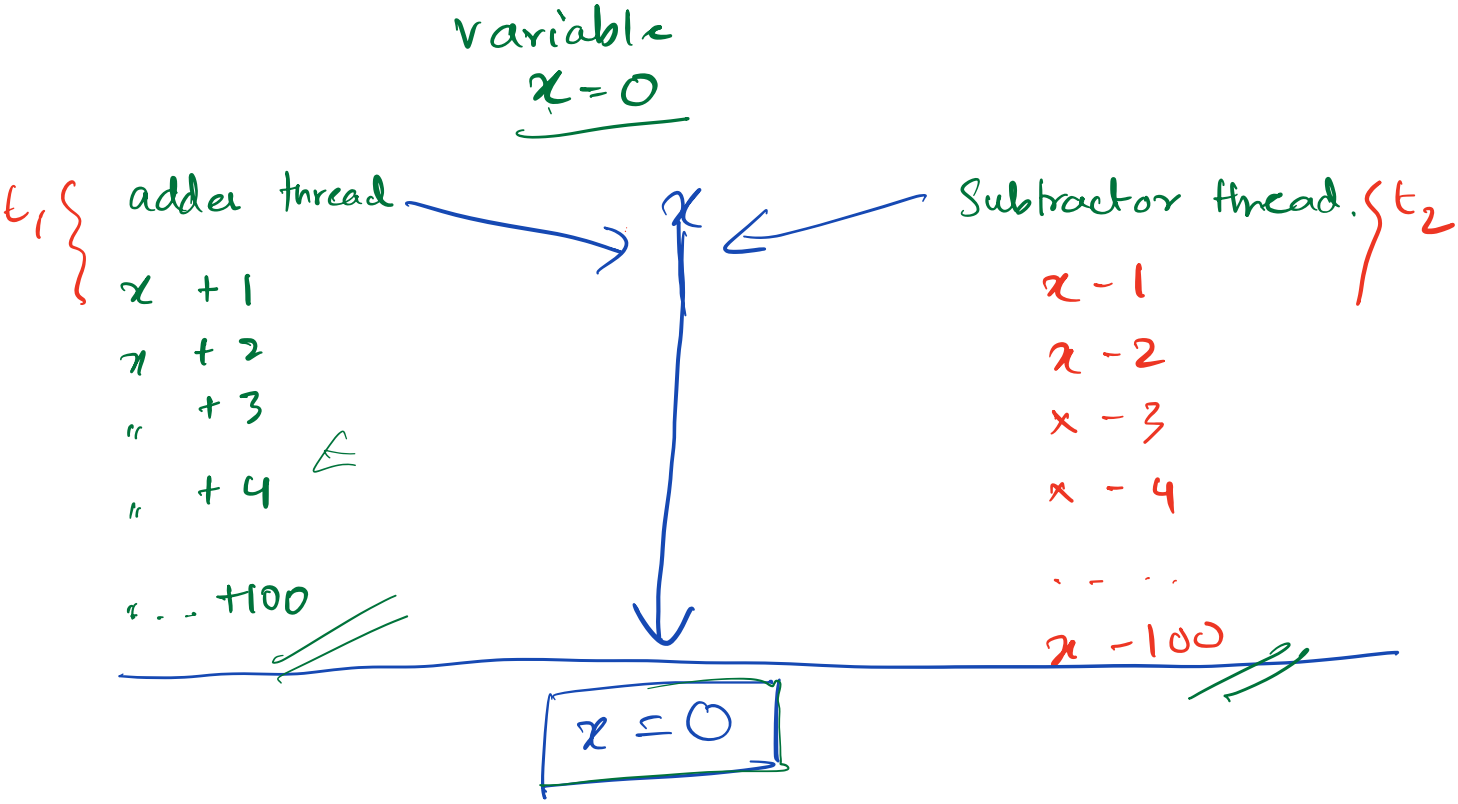
Agenda:

1. Multithreaded Merge Sort — Coding.
2. Adder & Subtractor Problem — Data Synchronisation issue
3. When does data Synchronisation happen?
4. Properties of a good solution to this issue.
5. Mutex.

Adder & Subtractor Problem :



Define the task:

- Adder class
- Subtractor class

```
class Adder implements Callable<Void> {

}
```

↳ To wait for the adder operation to get completed.

- conclusion - Expected output was alway x = 0 but that was not the case.

Why this happened?

v.val += i

1. Read the existing val
2. perform addition
3. update the value.

t1

t2

v.val -= i

1. Read the existing val
2. perform subtraction.
3. update the value.

t1 Operations

t = 0

t + 1 = 1

val = t

Val

0

1

-1

t2 operation

t = 0

t - 1 = -1

val = t

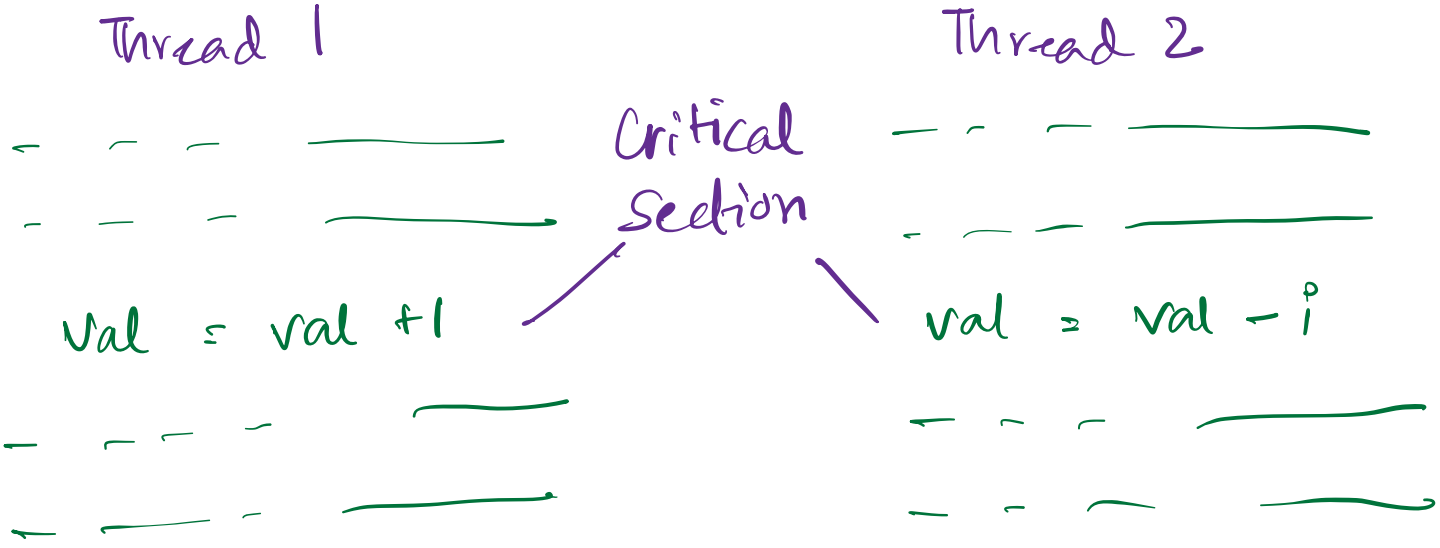
When does synchronisation Problem happen?

- When more than one thread accesses the same data at the same time, it can lead to inconsistent results.

1. Critical Section

It's a piece of code that is going to work on shared data, where potential synchronisation issues come.

Ex:



Note: It's not always possible to avoid critical sections.

2. Race Condition

When more than one threads are entering the critical section of same variable at the same time.

3. Preemptiveness:

When one thread switches to another thread before completing the critical section.

Note: Can't stop preemptiveness