IN1010 uke 12

Gruppe 9

Agenda

- Litt repetisjon av tråder/monitor
- Vente på at tråder skal bli ferdige
  - .join()
  - Barrierer
    - → CountDownLatch
    - → CyclicBarrier
- ☕ Pause 11:00-11:15
- Jobbe med oppgaver/trix/oblig!

Repetisjon av tråder

Sekvensiell vs parallell

Sekvensiell

- Ting gjøres i rekkefølge (etter hverandre)



Parallell

- Ting gjøres samtidig



Tråder i Java: Eksempel

```java
class PizzeriaParallell {
    Run | Debug
    public static void main(String[] args) {
        // Starter tråder
        for (int i = 0; i < 5; i++) {
            Thread nyTraad = new Thread(new PizzaBaker());
            nyTraad.start();
        }
```

```java
class PizzaBaker implements Runnable {
    @Override
    public void run() {
        // Oppgaven
        for (int i = 1; i <= 5; i++) {
            System.out.println("- Minutt " + i + " av baking...");
        }
        System.out.println(x:"Bakeren bakte en pizza.");
    }
}
```

Tråder og monitor

Kritiske regioner

- Som vi så kan det oppstå problemer når tråder endrer på felles data
  → disse områdene = kritiske regioner

→ Må beskytte kritiske regioner med lås
  → bare en tråd av gangen kan aksessere

Monitor

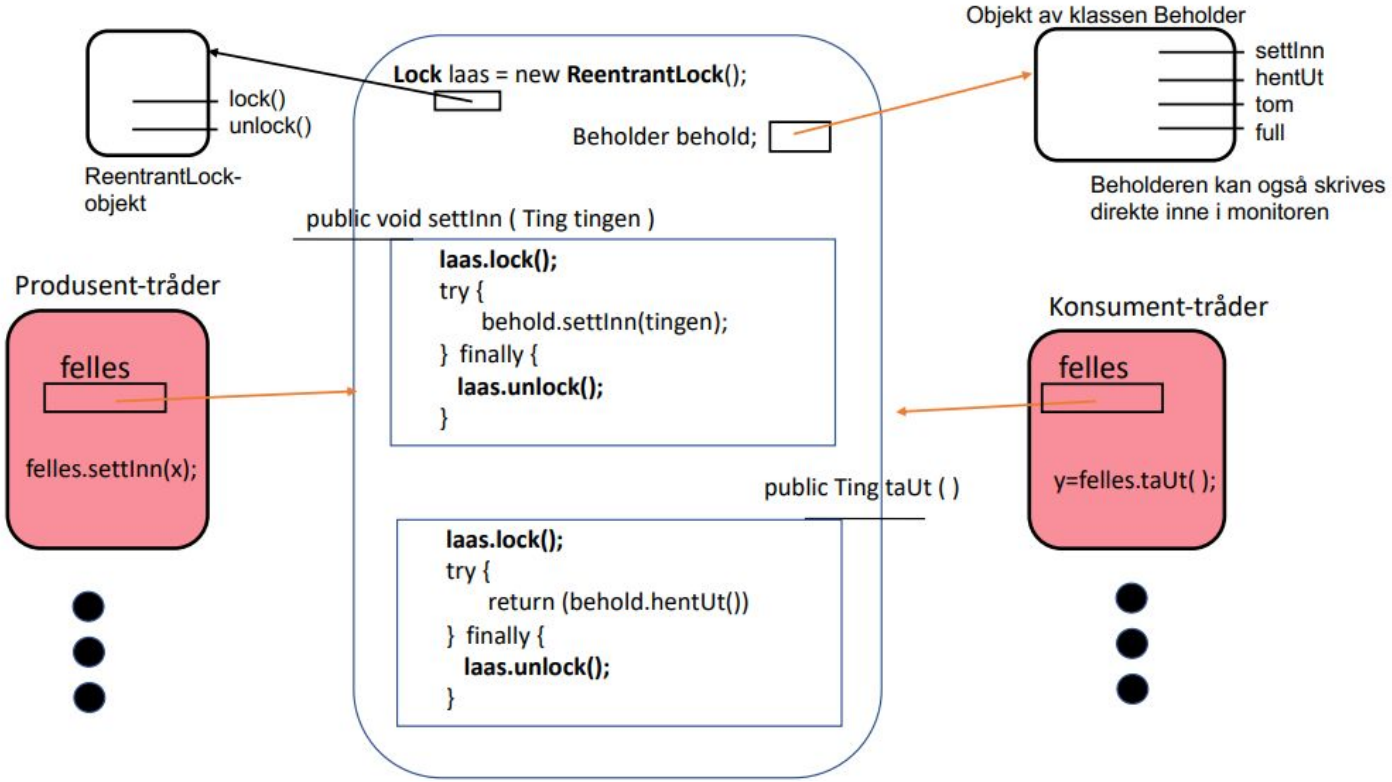
- Lager Monitor-klasse for å beskytte den delte dataen
- Har en klasse som vil bruke med tråder → lager en Monitor-klasse “rundt” den

Altså:

- Monitor er en beholder for den delte dataen (kritiske regioner)
- All låsing og opplåsing skjer her

→ Trådene kaller på metoder i monitoren for å endre på dataen på en sikker måte

Eksempel


lock()
unlock()
ReentrantLock-objekt
Lock laas = new ReentrantLock();
Beholder behold;
Objekt av klassen Beholder
settInn
hentUt
tom
full
Beholderen kan også skrives direkte inne i monitoren
public void settInn (Ting tingen)
laas.lock();
try {
behold.settInn(tingen);
} finally {
laas.unlock();
}
Produsent-tråder
felles
felles.settInn(x);
Konsument-tråder
felles
y=felles.taUt( );
public Ting taUt ( )
laas.lock();
try {
return (behold.hentUt())
} finally {
laas.unlock();
}

Vente på at tråder skal bli ferdige

Vente på tråder



Eksempel med pizzeria:

- Kunde bestilt 3 pizzaer

→ Må vente på at alle 3 blir ferdige før man kan levere dem

Vente på tråder

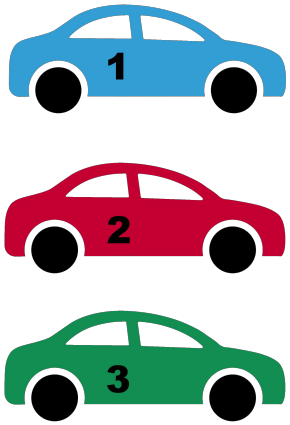
- 2 måter å gjøre det på:
  - med .join()
  - med en barriere

Main-tråden





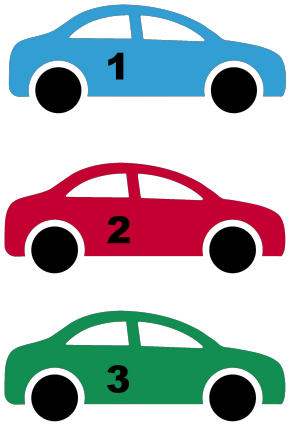
Bil-tråder



Join

- traad.join() → venter på at traad skal avslutte før går videre

Bil-tråder





Main-tråden



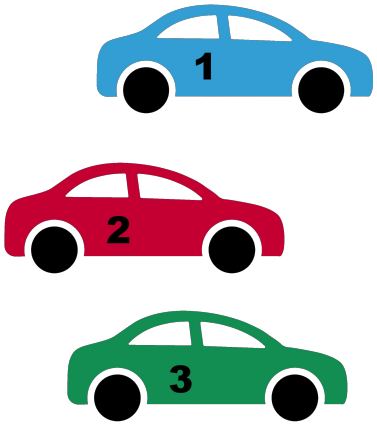
```
bil1.start();
bil2.start();
bil3.start();

bil1.join();
bil2.join();
bil3.join();
System.out.println("Race ferdig!");
```

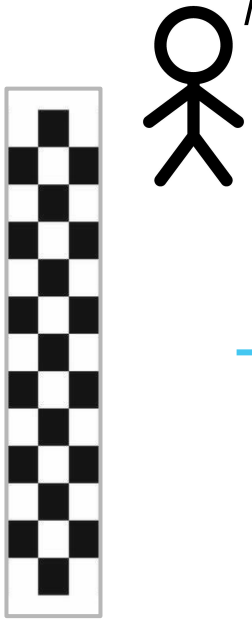
Join

- traad.join() → venter på at traad skal avslutte før går videre

Bil-tråder



Main-tråden



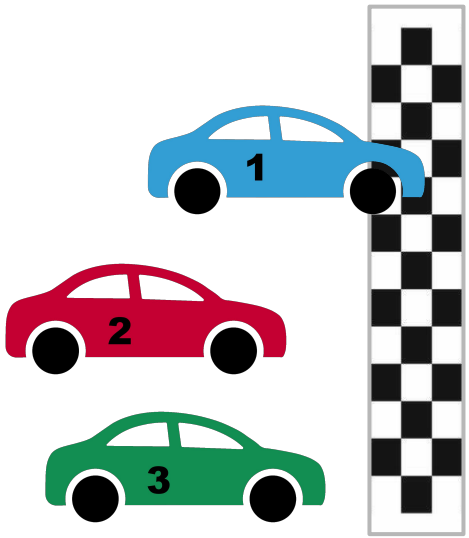
```
bil1.start();
bil2.start();
bil3.start();

bil1.join();
bil2.join();
bil3.join();
System.out.println("Race ferdig!");
```

Join

- traad.join() → venter på at traad skal avslutte før går videre

Bil-tråder



Main-tråden

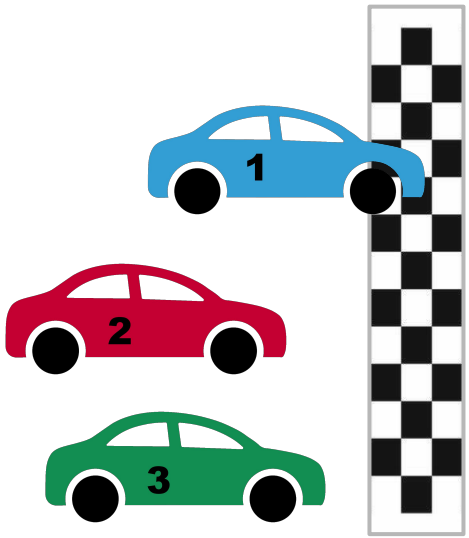
```
bil1.start();
bil2.start();
bil3.start();

bil1.join();
bil2.join();
bil3.join();
System.out.println("Race ferdig!");
```

Join

- traad.join() → venter på at traad skal avslutte før går videre

Bil-tråder



Main-tråden

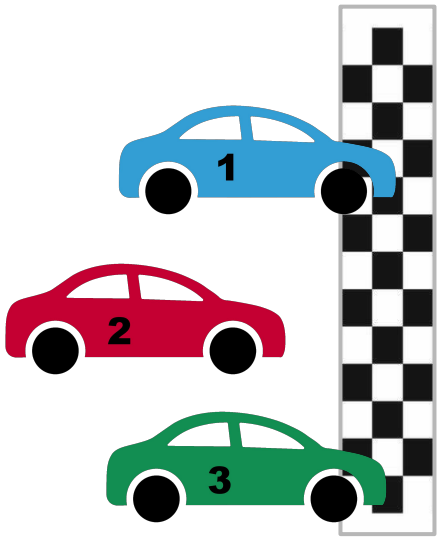
```
bil1.start();
bil2.start();
bil3.start();

bil1.join();
bil2.join();
bil3.join();
System.out.println("Race ferdig!");
```

Join

- traad.join() → venter på at traad skal avslutte før går videre

Bil-tråder



Main-tråden

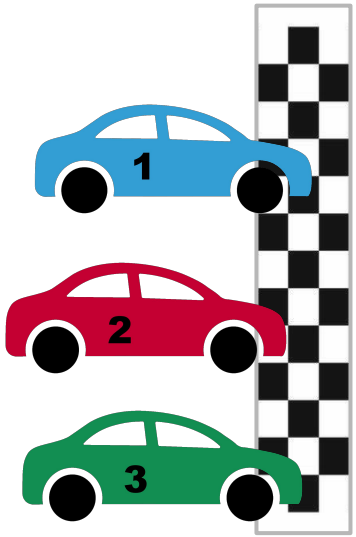
```
bil1.start();
bil2.start();
bil3.start();

bil1.join();
bil2.join();
bil3.join();
System.out.println("Race ferdig!");
```

Join

- traad.join() → venter på at traad skal avslutte før går videre

Bil-tråder



Main-tråden

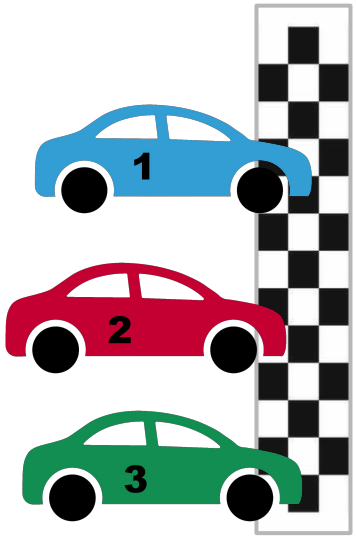
```
bil1.start();
bil2.start();
bil3.start();

bil1.join();
bil2.join();
bil3.join();
System.out.println("Race ferdig!");
```

Join

- traad.join() → venter på at traad skal avslutte før går videre

Bil-tråder



Main-tråden

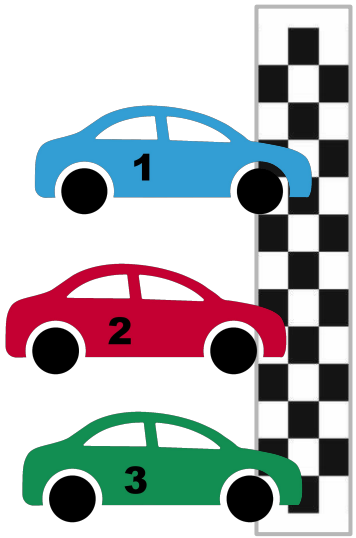
```
bil1.start();
bil2.start();
bil3.start();

bil1.join();
bil2.join();
bil3.join();
System.out.println("Race ferdig!");
```

Join

- traad.join() → venter på at traad skal avslutte før går videre

Bil-tråder



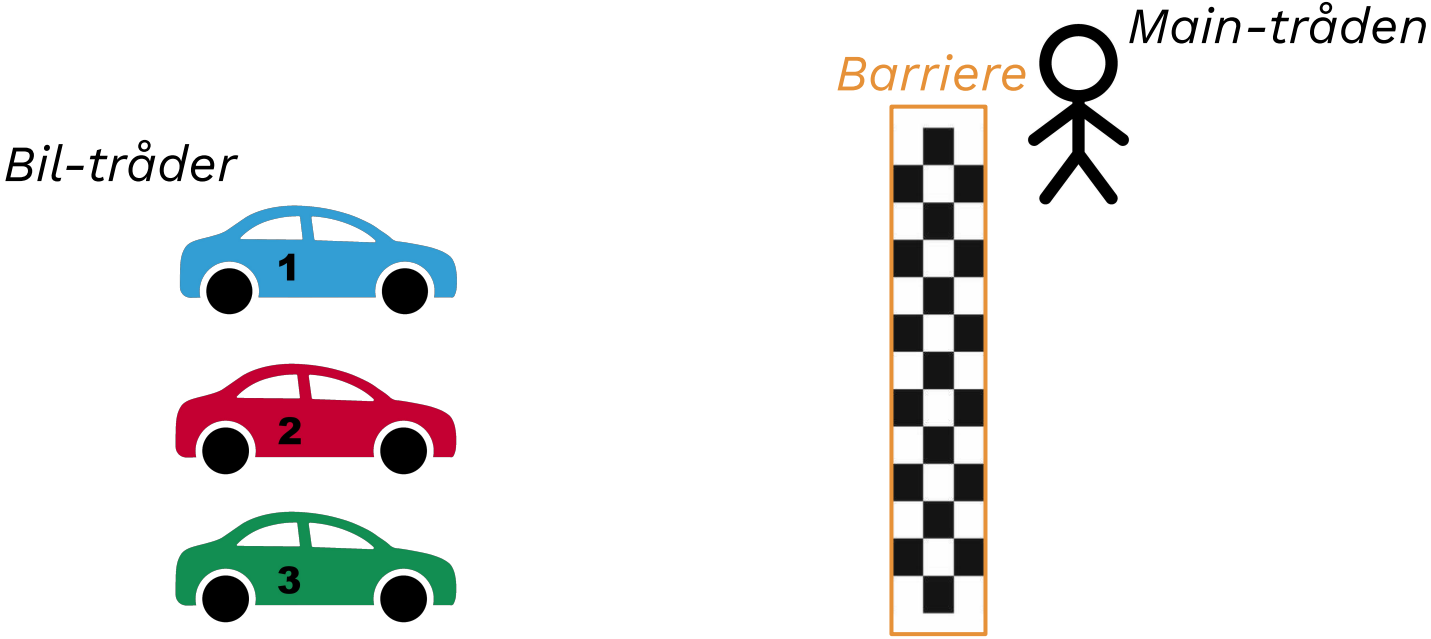
Main-tråden

```
bil1.start();
bil2.start();
bil3.start();

bil1.join();
bil2.join();
bil3.join();
System.out.println("Race ferdig!");
```

Barriere: CountDownLatch

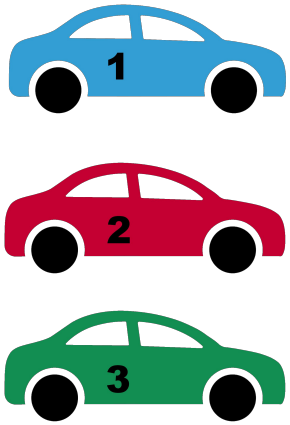
- Når en tråd er ferdig teller den ned med .countDown()
- Kan vente på at alle tellet ned med .await()



Barriere: CountDownLatch

countdown:
3

Bil-tråder



Barriere



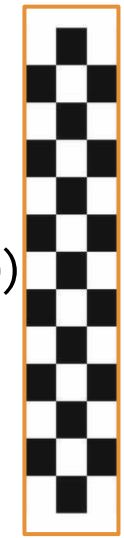
Main-tråden

```
CountDownLatch barriere=
new CountDownLatch(3);
```

Barriere: CountDownLatch

countdown:
3

Barriere



Main-tråden



```
CountDownLatch barriere=
new CountDownLatch(3);
```

Bil-tråder



=new Thread(new BilTraad(barriere))

Barriere: CountDownLatch

countdown:
3

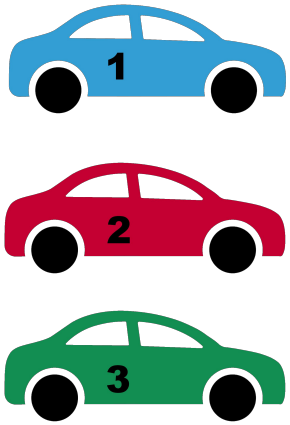
Main-tråden

Barriere



```
CountDownLatch barriere=
new CountDownLatch(3);
```

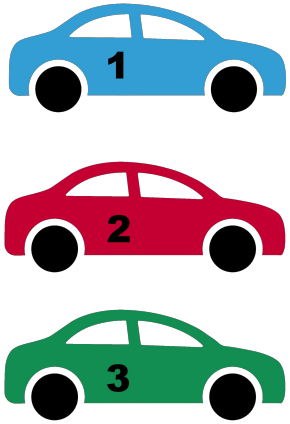
Bil-tråder



Barriere: CountDownLatch

countdown:
3

Bil-tråder



Barriere



Main-tråden

```
CountDownLatch barriere=
new CountDownLatch(3);

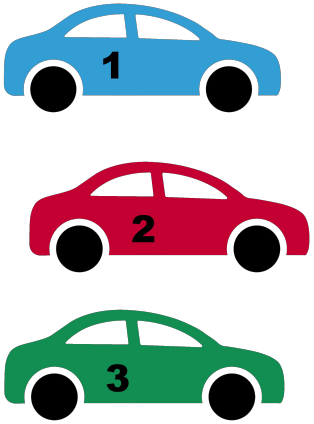
bil1.start();
bil2.start();
bil3.start();

barriere.await();
System.out.println("Race
ferdig!");
```

Barriere: CountDownLatch

countdown:
3

Bil-tråder



Barriere



Main-tråden

```
CountDownLatch barriere=
new CountDownLatch(3);

bil1.start();
bil2.start();
bil3.start();

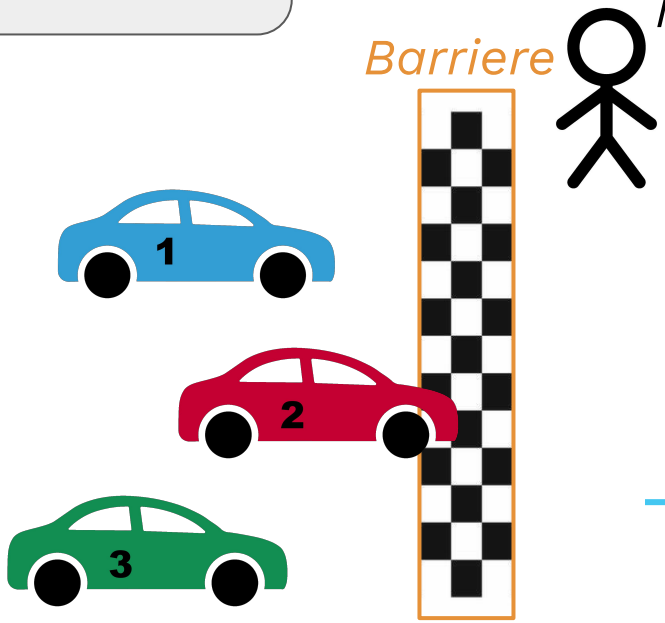
barriere.await();
System.out.println("Race
ferdig!");
```

Barriere: CountDownLatch

countdown:
3

Bil-tråder

Barriere



Main-tråden

```
CountDownLatch barriere=
new CountDownLatch(3)

bil1.start()
bil2.start()
bil3.start()

barriere.await()
System.out.println("Race
ferdig!")
```

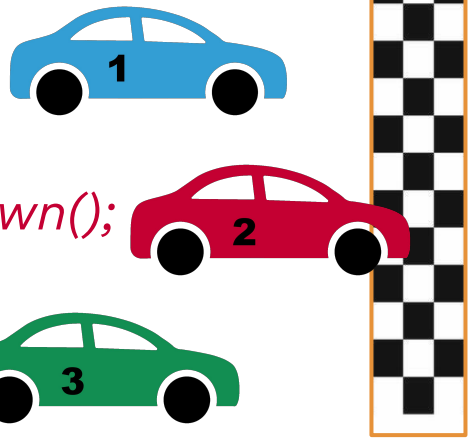
Barriere: CountDownLatch

countdown:
2

Main-tråden

Barriere

Bil-tråder



barriere.countDown();

```
CountDownLatch barriere=
new CountDownLatch(3)

bil1.start()
bil2.start()
bil3.start()

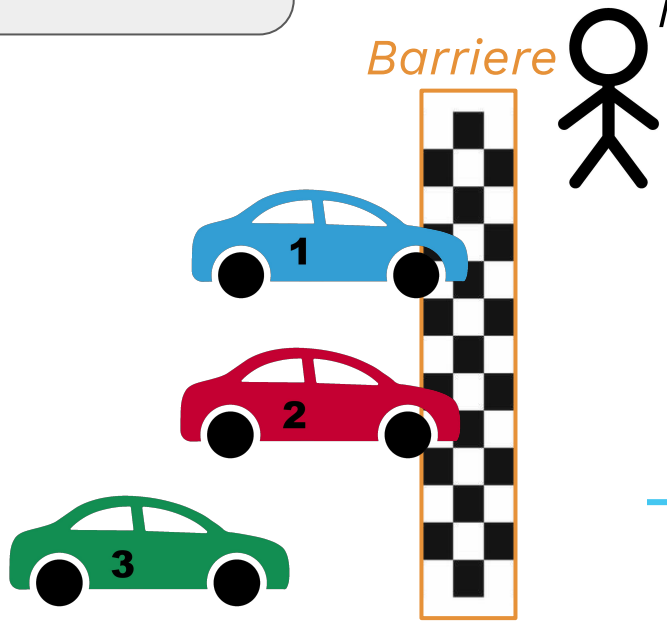
barriere.await()
System.out.println("Race
ferdig!")
```

Barriere: CountDownLatch

countdown:
2

Bil-tråder

Barriere



Main-tråden

```
CountDownLatch barriere=
new CountDownLatch(3)

bil1.start()
bil2.start()
bil3.start()

barriere.await()
System.out.println("Race
ferdig!")
```

Barriere: CountDownLatch

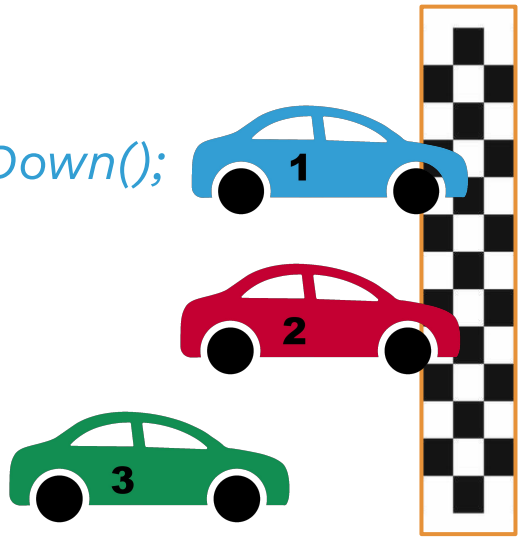
countdown:
1

Barriere

Main-tråden

Bil-tråder

barriere.countDown();




```
CountDownLatch barriere=
new CountDownLatch(3)

bil1.start()
bil2.start()
bil3.start()

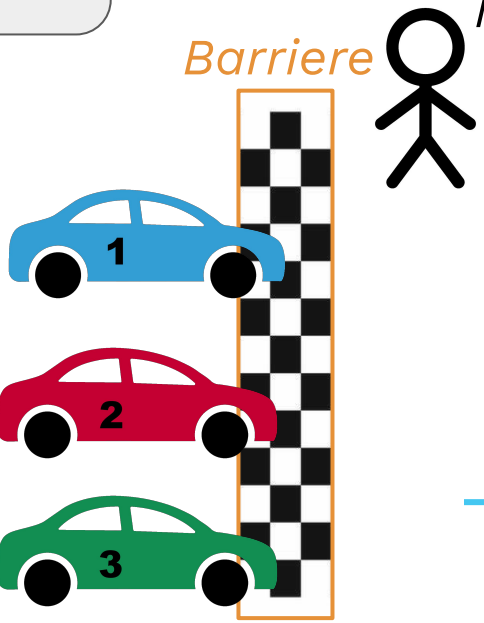
barriere.await()
System.out.println("Race
ferdig!")
```

Barriere: CountDownLatch

countdown:
1

Bil-tråder

Barriere



Main-tråden

```
CountDownLatch barriere=
new CountDownLatch(3)

bil1.start()
bil2.start()
bil3.start()

barriere.await()
System.out.println("Race
ferdig!")
```

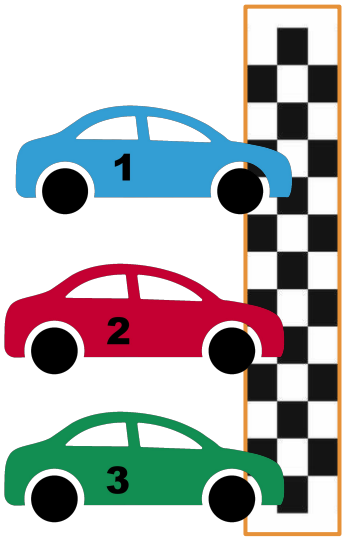
Barriere: CountDownLatch

countdown:
0

Barriere

Main-tråden

Bil-tråder



barriere.countDown();

```
CountDownLatch barriere=
new CountDownLatch(3)

bil1.start()
bil2.start()
bil3.start()

barriere.await()
System.out.println("Race
ferdig!")
```

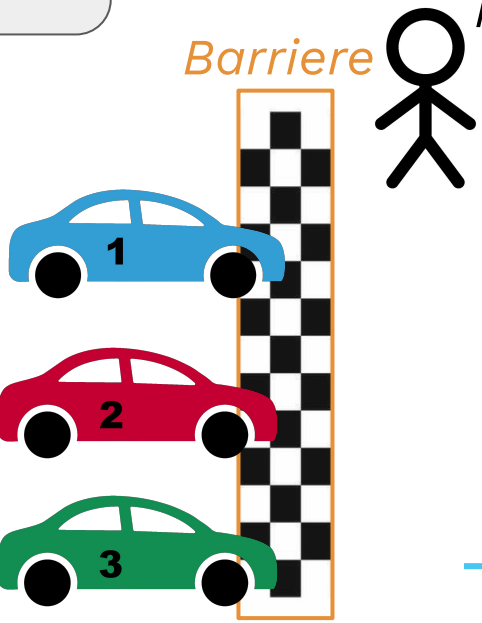
Barriere: CountDownLatch

countdown:
0

Barriere

Main-tråden

Bil-tråder




```
CountDownLatch barriere=
new CountDownLatch(3)

bil1.start()
bil2.start()
bil3.start()

barriere.await()
System.out.println("Race ferdig!")
```

barriere.countDown();

Barriere: CyclicBarrier

- Eksempel: CyclicBarrier(3)
  - Hver tråd kaller på .await() for å vente på de andre
  - Når 3 tråder har kalt .await() slipper alle gjennom

- Forskjell fra CountDownLatch:
  - Samme CyclicBarrier kan brukes flere ganger
  - Når en tråd kaller på .await() kan den ikke fortsette før barrieren åpner
    (med CountDownLatch bare teller tråden ned og kjører videre)

- Holder med å kunne én av dem i IN1010!

Kodeeksempel

CountDownLatch,
monitor

Jobb med oppgaver!

IN1010 Emnesiden → Grupper →
Gruppe 9 → Uke12

Husk å ta pause ☕