Repetisjonsgruppetime

I dag

- Lenkelister
- Rekursjon
- Tråder
- Eksamenstips

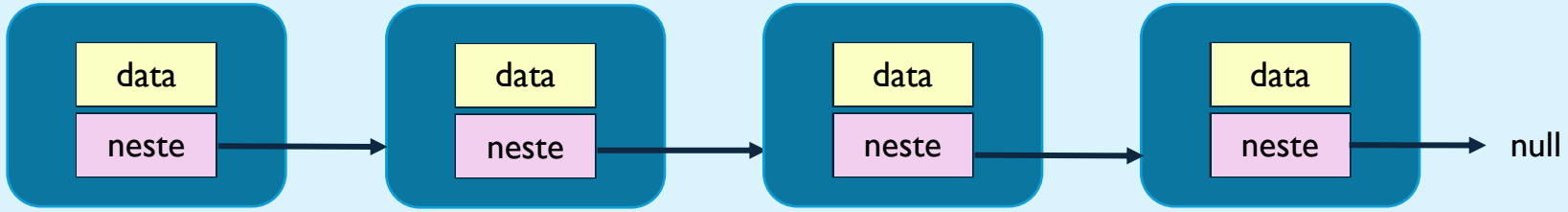
Lenkelister

Lenkelister

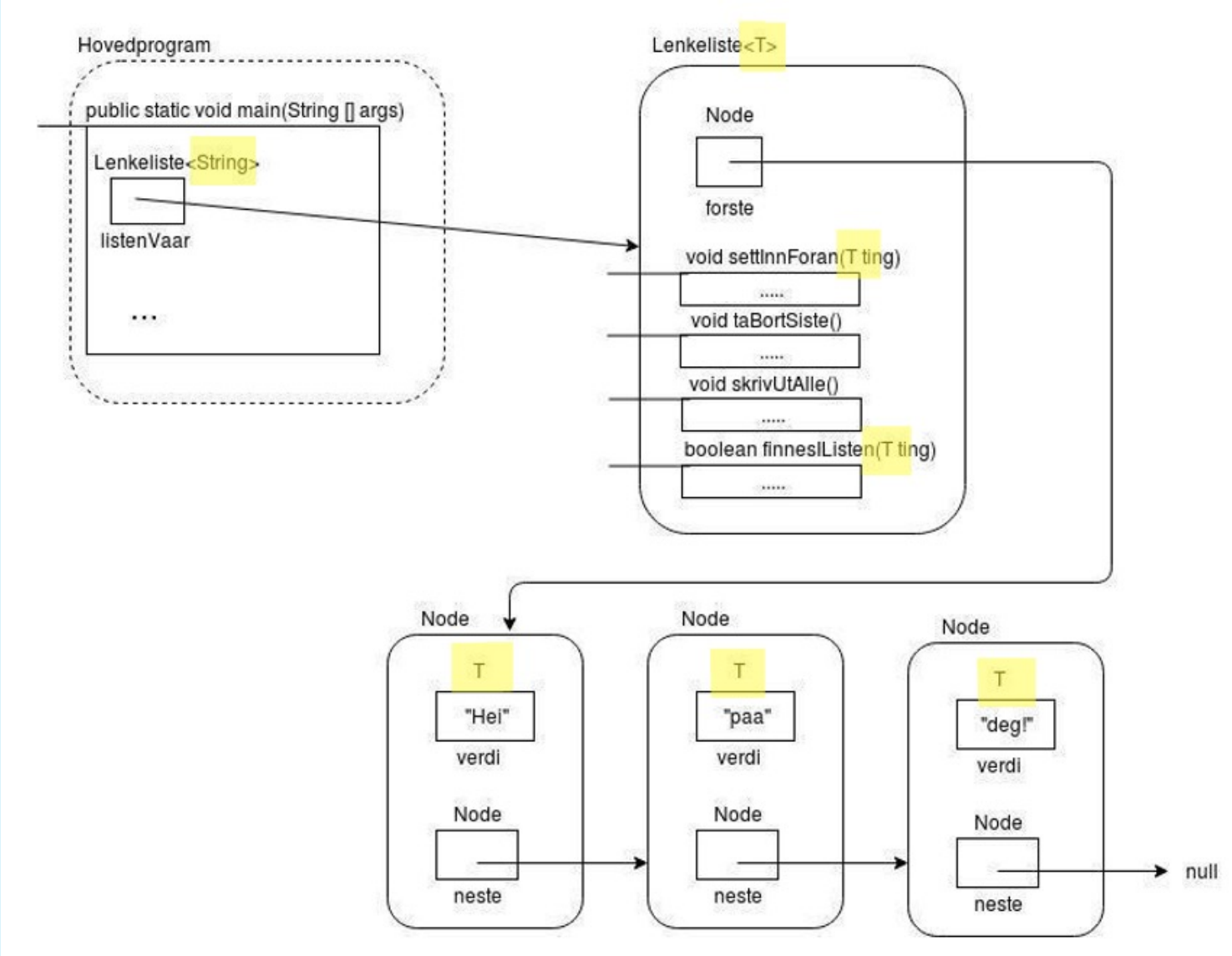
- Noder som inneholder data og er koblet sammen i en lenkestruktur.
- Lenkelisten vil håndtere innsetting, sletting og andre operasjoner.
- Referanse til første (evt. siste) node ligger inni lenkelisten.
- Gjennom referansen får vi tilgang til nodene og dermed data lagret i nodene våre.

Ulike typer lenkelister

- Kø/FIFO – first in first out
- Stack/stabel – first in last out
- Prioritetskø
  - Her er det relevant med interfacet Comparable
- Enkellenket liste
- Dobbellenket liste



Lenkelister kan være generiske



Interfacet Comparable

- Sammenligner objekter på en ønsket måte.
- Praktisk i prioritetskøer
  - Klassen Hund som prioriteres ut i fra alderen.
- Comparable er generisk: har klasseparameter der vi sender med typen som skal sammenlignes

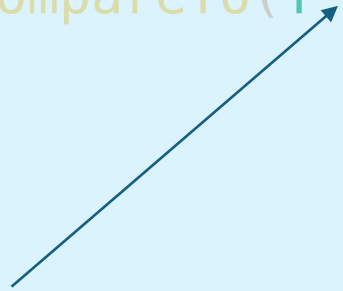
Comparable sin compareTo

compareTo-metoden må implementeres i klasser som implementerer Comparable

```
public interface Comparable<T>
{
    public int compareTo(T o);
}
```

- Leksikografisk (alfabetisk) bokstavsammenligning

- Returnerer et positivt heltall når dette objektet er større enn det som sendes inn.
- Returnerer et negativt heltall når dette objektet er mindre enn det som sendes inn.
- Returnerer 0 når objektene vi sammenligner er like.



Objektet vi sammenligner med er det som sendes inn i compareTo()

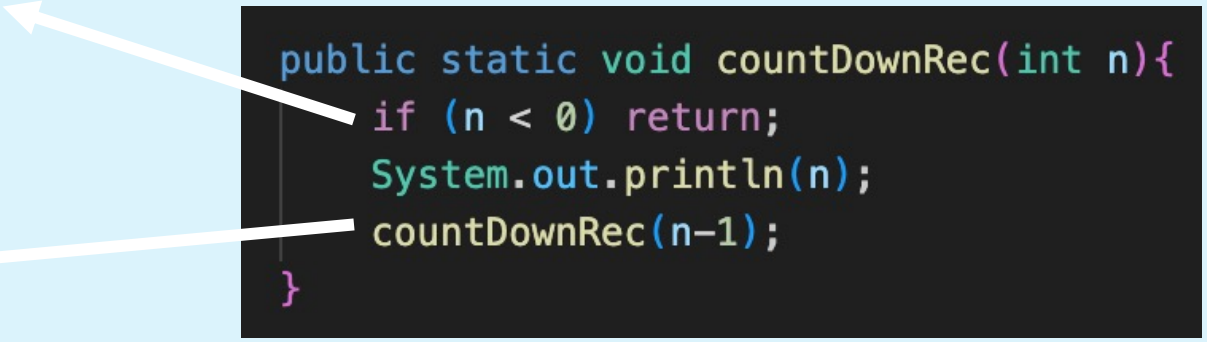
Interfacet Iterable

- For å iterere over elementer i beholdere med en for each-løkke
- hashNext() – sjekker om det er et til element å iterere over.
- next() – går til neste element.
- hasNext-metoden og next-metoden kan tilpasses nøyaktig slik vi ønsker å ha dem

Rekursjon

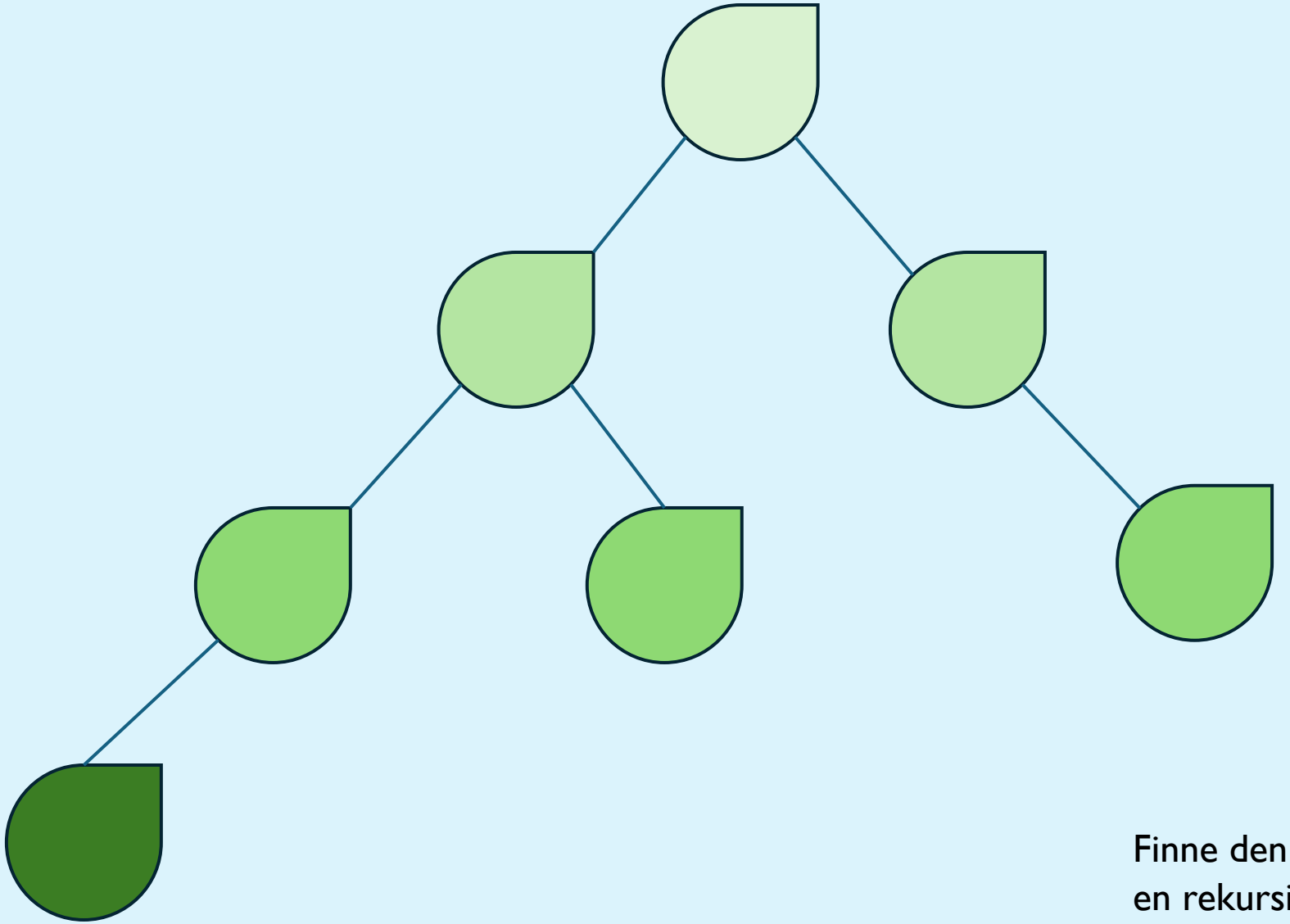
Hva inneholder en rekursiv metode?

- **Basistilfelle** (“base case”)
  - Rekursjonen stopper, kaller ikke seg selv igjen → «stop case».
  - Kallene «nøstes opp» fra siste til første kall.
- **Rekursjonssteg** (“recursive step”)
  - Endrer noe slik at man til slutt når basistilfellet.
  - I dette tilfelle er det ett rekursivt kall, men en det kan være flere recursive kall (feks Fibonacci).



```
public static void countDownRec(int n){
    if (n < 0) return;
    System.out.println(n);
    countDownRec(n-1);
}
```

Eksamenstips: tegn rekursjonen



Finne den dypeste noden i et tre med en rekursiv metode

(lik rekursjonsoppgave eksamen 2019)

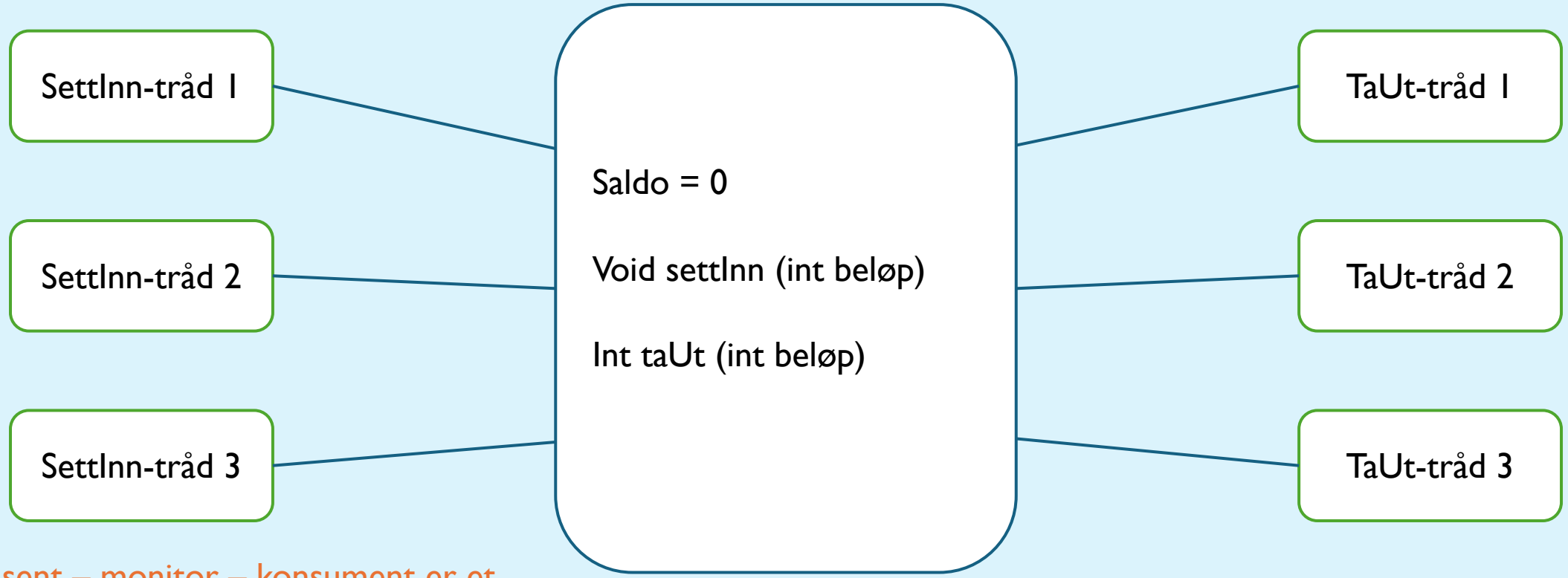
Tråder

Typisk: produsent og konsument

Produsent-tråder som setter inn penger på kontoen (monitoren)

Monitor – bankkonto som inneholder en saldo og metode for å sette inn og ta ut penger

Konsument-tråder som tar ut penger fra kontoen



Produsent – monitor – konsument er et typisk mønster på eksamensoppgaver!

Vi har brukt lock og condition:

Låser

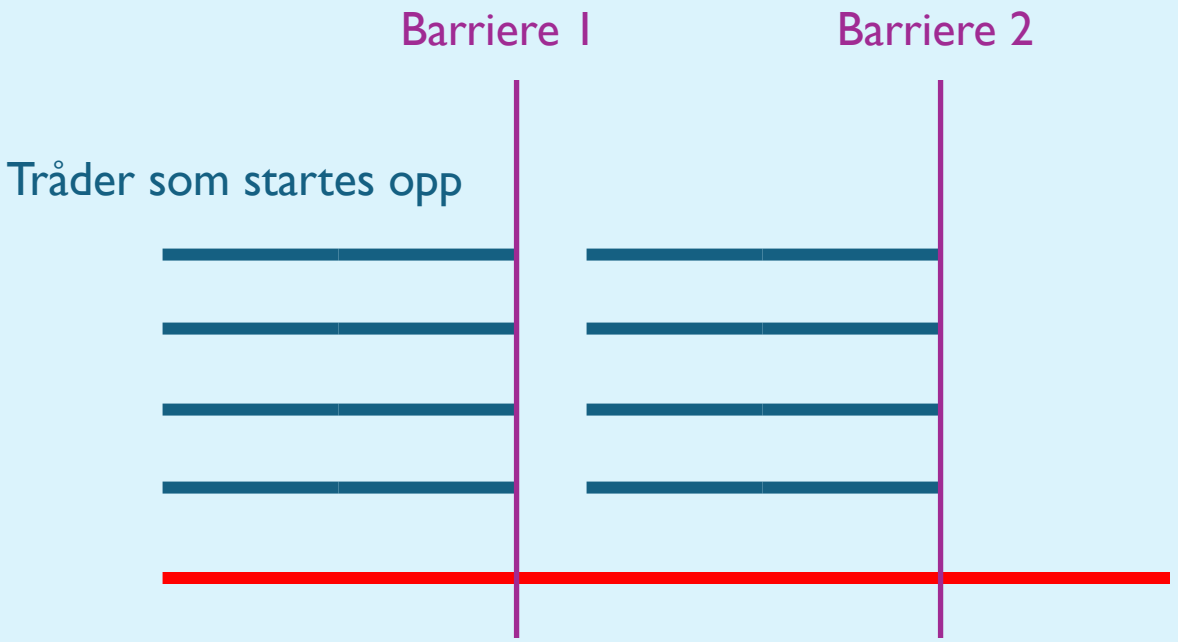
- Kun én tråd kan aksessere delt data om gangen.
- Kritisk region der kun en tråd kan gjøre endringer om gangen.
- Andre tråder må vente i kø.

Conditions

- Passiv venting - tråden venter til den får beskjed/signal.
  - I motsetning til aktiv venting der tråden kontinuerlig sjekker om det er klart.

Barrierer

- Barrierer gjør at tråder møtes ved ett bestemt punkt i programmet.
- Tråder ikke går videre før alle har nådd barrieren.
- Kan kontrollere rekkefølgen av operasjoner på en trygg måte



Maintråd med barrierer – maintråden fortsetter ikke videre før trådene har kommet til barriere.

Tips til eksamen

- Løs tidligere eksamensoppgaver mange ganger
  - Lærer deg «mønster» i eksamensoppgavene
  - Gjør det kanskje lettere å tolke oppgavene/vite hvordan du skal gå frem for å løse oppgaver under eksamen.
- Noter ting du ofte glemmer eller feil du ofte gjør underveis
  - Bruk disse som en «checkliste» på eksamen
- Vit omtrent hvor mye tid du har per oppgave (ut i fra poeng)
- Husk at du ikke kan kompilere kode på eksamen, og du får ikke så mye hjelp om feil – skru av hjelpen fra VSCode☺

Flere tips til eksamen

- Tenk på edgecases
  - Nullsjekker, gyldig input osv.
  - Lenkelister: hvis man tar ut siste eller første element i listen, tom liste
- GUI-oppgaver krever en del kode som ser veldig lik ut og som du ikke behøver å huske – skriv ut dette.
- Løs lenkelisteoppgaver, og lær deg/forstå det godt selv om du kan ta med hjelpemidler
- Tråder: Det er som regel noe som passer i rollen som produsent og konsument. Se etter dette mønsteret.
  - Husk å bruke Lock (inkl lock() og unlock()) og Condition (inkl. Await() og signalAll())

Lykke til på eksamen!☺