Status midtveis

- Praktisk om midtveiseksamen
- Oppsummering av pensum hittil
- Midtveisevaluering
- Gjennomgang av prøveeksamen



Midtveiseksamen 2026

- Tid: **20. mars kl 15:00–17:00**
- Sted: **Silurveien 2**
- Hjelpemidler: **Ingen!**
- Sensur: Bokstavkarakterer **A–F**
- Sluttkarakter: Midtveiseksamen teller 25 %
  - Både midtveis- og avsluttende eksamen må bestås
  - De to må bestås samme semester

Det finnes en meget god nettside om eksamen i Silurveien:
https://www.uio.no/studier/eksamen/inspera/gjennomforing/

Oppmøte

Møt opp minst en halvtime før eksamen.

Trafikkproblemer er ingen unnskyldning.

Ikke på listen?

Hvis du møter opp og finner at du ikke står på listen selv om du mener du burde stått der, så snakk med en eksamensvakt. Du har rett til å gå opp til eksamen uansett, så blir saken avgjort i ettertid.

Varighet

Eksamen varer bare to timer.

- Det blir *ikke* tid for faglærerne til å gå «trøsterunde».

Men om du tror det er en feil i oppgaven?

1. Les oppgaveteksten én gang til.
2. Les oppgaveteksten enda en gang, og nøye.
3. Hvis du stadig mener at det er en *viktig* feil i oppgaven, kan du ta kontakt med eksamensvakten.

Automatrettet eksamen

Midtveiseksamen er laget for automatisk retting.

- Oppgavene er av samme type som i prøveeksamen:
  - Én- og flervalgsoppgaver
  - Dytte kodebiter på plass
  - Hva skjer?
  - . . . og lignende

  Altså ingen tradisjonelle programmeringsoppgaver.
- Poeng:
  - Generelt: poeng for rett svar, 0 poeng for galt
  - Noen (markerte!) oppgaver: > 0 for rett svar, < 0 for galt svar, 0 om man ikke svarer, men aldri < 0 på en oppgave totalt.
- Sensuren blir sannsynligvis klar mandagen etter.
- Det er ikke mulig å klage på karakteren man får.

Ved sykdom

Hvis du er syk på eksamensdagen, må du ta kontakt med lege *samme dag* for å få en legeattest. Denne må leveres i et nettskjema innen tre dager.

Det arrangeres utsatt prøve («konteeksamen») i april for de som var syke og har levert legeattest.

Om du stryker

Du kan da melde deg til ny prøve («konteeksamen»).

Om du står men får en dårlig karakter

Trøst deg med at midtveiseksamen bare teller 25 %.

Det aller viktigste å ha med seg

- studentkortet
- UiO-brukernavn og passord

Viktig!

Sjekk på forhånd at de fungerer!

- blyant eller kulepenn

Hovedpunkter i pensum hittil

- Programmeringsspråket Java
  - Klasser og interface-er
    - Subklasser
    - Konstruktører
    - Polymorfi
    - Generiske klasser
  - Unntak
  - Iteratorer
- Modellering
  - Arv og komposisjon
  - Klassediagrammer
  - Datastrukturtegninger
- Beholdere
  - Arrayer
  - Lenkelister i ulike varianter

Sjekkliste for klasser og objekter

Objektorientert programmering dreier seg om å bruke objekter til å modellere/programmere ting eller begreper.

Klasser definerer hvordan et objekt skal representeres.

Subklasser definerer spesialiseringer av klasser.

Interface-er beskriver egenskaper som klasser kan implementere.

Statiske variabler og metoder finnes i «selve klassen».

Referanser peker på objekter; deklarasjonen av referansen angir ønsket subklassenivå.

Typekonvertering kan gi oss et annet «blikk» (dvs subklassenivå) på et objekt.

Polymorfe metoder velger riktig redefinert (`@Override`-et) metode.

Generiske klasser har en typeparameter (f eks `class Kø<T>`).

Bruk av klasser og objekter

Noen ting å huske på med klasser, subklasser og objekter:

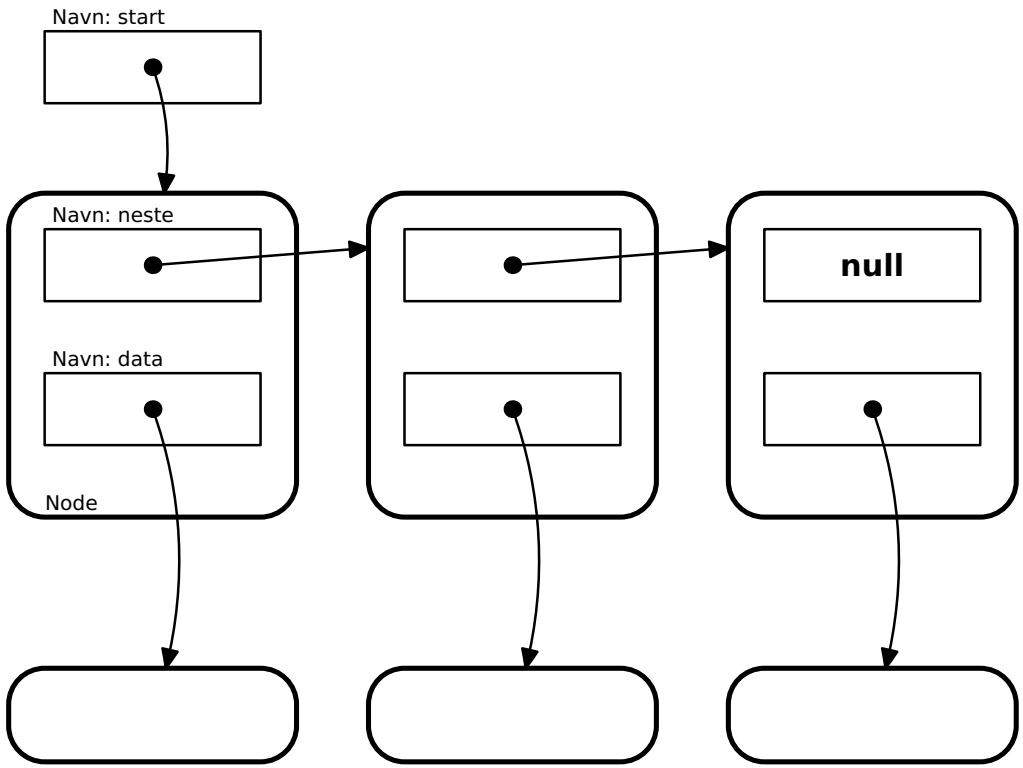
- Et objekt opprettet med **`new Xxx(...)`** er alltid av den klassen. Objekter kan ikke endre klasse.
- **`instanceof`** kan fortelle oss om objektet er av en gitt type (dvs klasse eller interface, inkludert alle dets subklasser eller subinterface).
- Når vi kaller på metoder, kalles alltid (re)definisjonen i selve objektet.

Bruk av referanser

- Alle referanser er deklarert til å være av en gitt type (dvs klasse/interface).
- Referanser kan angi objekter av den gitte typen eller en subklasse/implementasjon. Dette sjekkes av kompilatoren.
- Ved å bruke typekonvertering («casting») kan vi angi at vi ønsker et annet blikk på referansen. Kjøresystemet sjekker om dette er lov.

Lenkelister

En meget fleksibel datastruktur med objekter kjedet sammen med pekere.



Det finnes mange varianter:

Pekere i selve objektet istedenfor å ha egne node-objekter

 Sparer plass  Krever planlegging under programmeringen  Objekter kan ikke være i flere lister

Sluttpeker i tillegg til startpekeren

 Litt mer komplisert kode  Mye raskere for operasjoner på slutten

Dobbeltlinket

 Mer komplisert kode  Mye raskere for endringer som ikke er i starten eller slutten

Egne hode- og haleobjekter

 Tar litt plass  Endringer blir enklere og litt raskere