Uke 13:
Prøveeksamen og eksamensinfo

Siri Moe Jensen

Geir Kjetil Sandve

Dagens forelesning

- Gjennomgang av (deler av) prøveeksamen
- Spørsmål fra prøveeksamen
- Informasjon og tips til eksamen

- Frivillig deltakelse i Master-undersøkelse
  - Ikke endel av pensum/ undervisningsmaterialet i emnet
  - Pensum- og eksamensrelevant
  - Siri går gjennom oppgavene til slutt
  - Det gjøres ikke opptak av dette

Hva skal vurderes, og hvordan?

Etter å ha tatt IN1000

A. forstår du prinsippene for objektorientert programmering og kan benytte disse til å skrive enklere objektorienterte programmer

B. kan du programmere i programmeringsspråket Python og kan bruke dette til å løse mindre problemer ved hjelp av valg, løkker, funksjoner, lister, klasser og objekter

C. kan du skrive oversiktlige og lesbare programmer

D. er du i stand til å sette deg inn i andres programmer, finne eventuelle feil i dem og modifisere dem

Hoveddeler av eksamen som må bestås

- Del 1 og 2 samlet: Forstå, utvide, finne og rette feil i programmer ("tracing") (D)
- Del 3: Procedural programmering i Python (B)
  - Riktig valg og bruk av Python mekanismer for løsing av enklere problemer
  - ..og en mer krevende algoritme
- Del 4: Objektorientert programmering og datastrukturer (A)
  - Grunnleggende teori, f eks koble konsepter som klasse, objekt, instansmetode, konstruktør til kode-eksempler.
  - Programmering med flere klasser og referanser mellom objekter
- Vurderes på tvers av deloppgaver med programmering (Del 3 og 4)
  - Algoritmer - har vist at kandidaten kan komme frem til og implementere en fremgangsmåte for å løse et gitt problem (A, B)
  - Oversiktlige og lesbare programmer vurderes hovedsaklig i obligene (C)

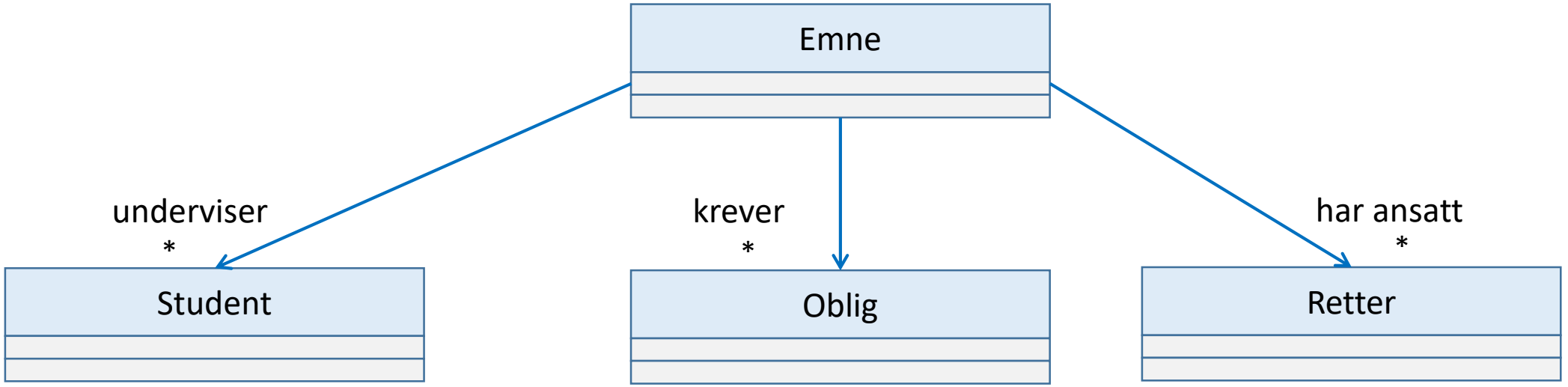
Prøveeksamen oppgave 4

- 4 klasser: Emne, Student, Oblig og Retter
- Flere av klassene håndterer id for objekter av andre klasser
  .. men kun Emne-klassen er oppgitt å referere til objekter av disse klassene

(ikke med på årets prøveeksamen)

Les gjennom <...>. Tegn deretter et UML klassediagram som viser alle klasser du er bedt om å implementere (kun klassenavn) og relasjonene mellom dem. For hver relasjon skal du ta med antall og navn (hva relasjonen representerer).

IN1000 forenkling/ avgrensning:
-implementasjons-nær modell
-relasjoner viser kun klasser med instansvariabler som kan referere til objekter av andre klasser

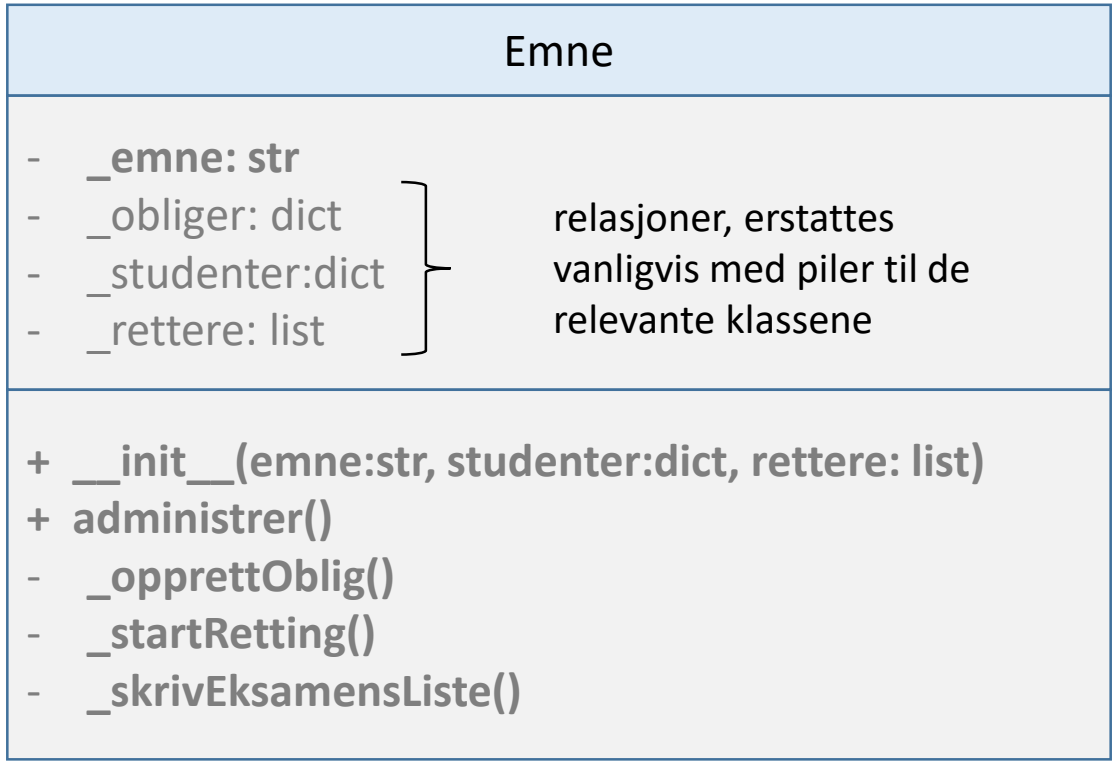


(ikke med på årets prøveeksamen)

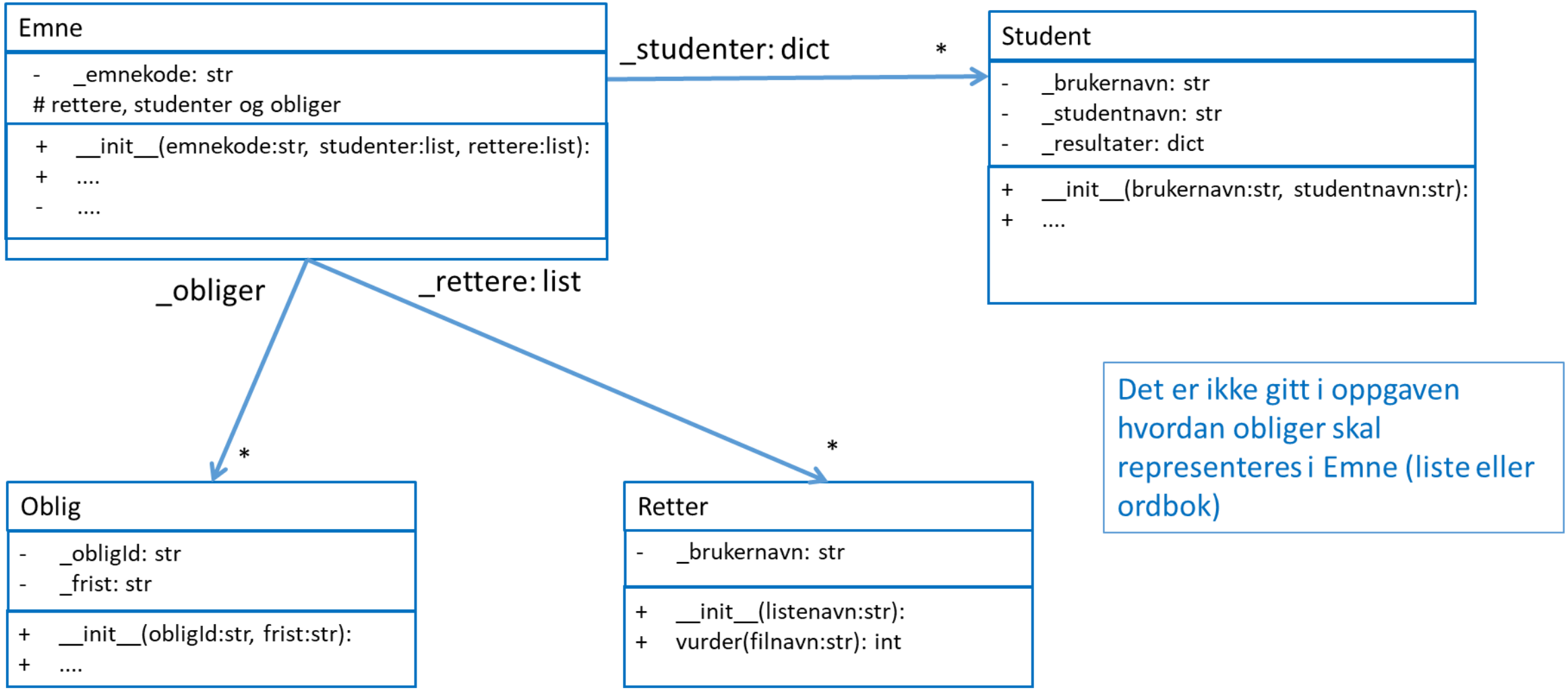
Lag deretter en ny tegning, med et klassediagram som kun viser klassen Emne.

Her skal du ta med alle instansvariabler og metoder i Emne. Angi type for instansvariabler, parametere og returverdier, og om instansvariabler og metoder er public (tilgjengelige i klassens grensesnitt) eller ikke.

Det meste av dette er gitt i oppgaveteksten, noe vil du kanskje bestemme underveis i oppgave-løsingen.



(ufullstendig) UML-diagram for del 4 på prøveeksamen

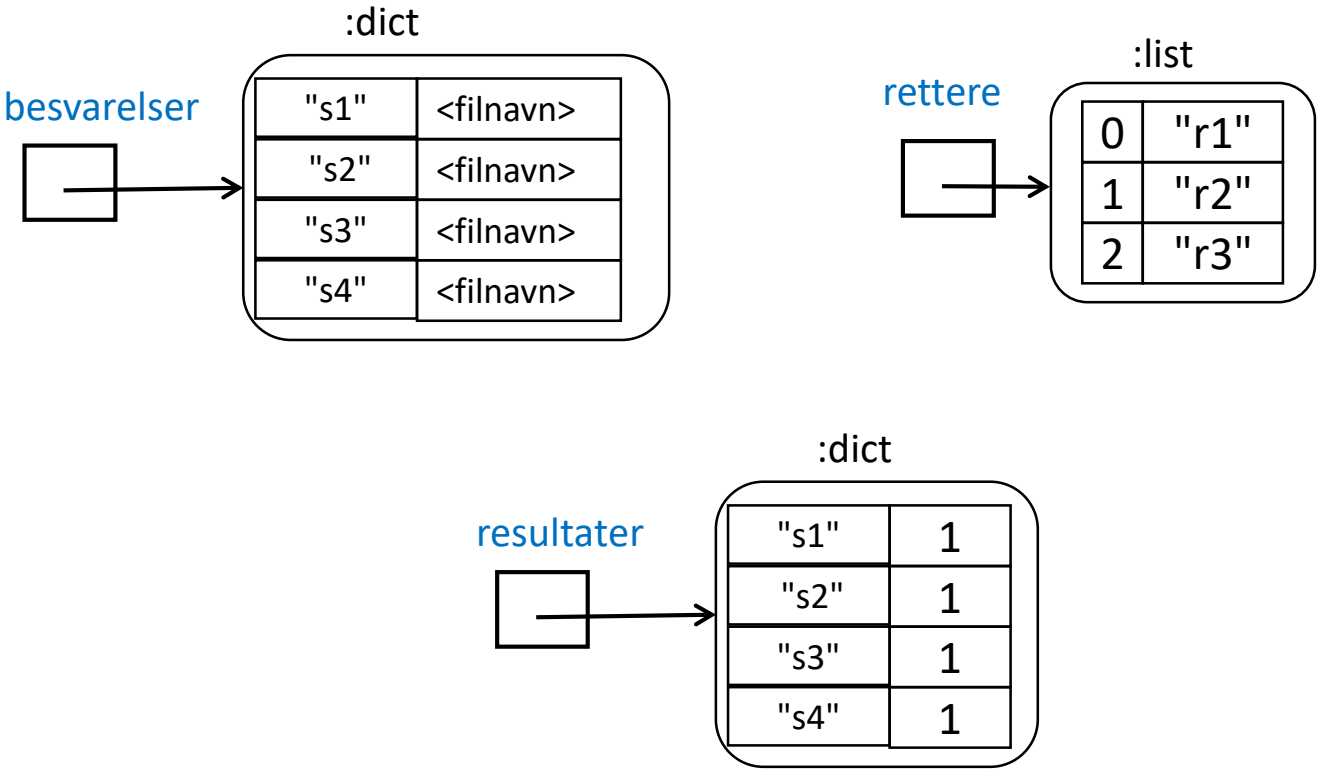


Oppgavene 4f)

fordelRetting *tar som parametere en ordbok med studenters besvarelser og en liste med rettere,*

og lar retterne vurdere én og én besvarelse (bruk metoden ***vurder*** *i klassen* ***Retter****) slik at alle retterne ideelt sett vurderer like mange. Altså skal ingen retter vurdere mer enn 1 besvarelse mer enn de andre.*

*Resultatene av retternes vurderinger samles i en ordbok der nøkkelverdiene er studentenes brukernavn. Metoden markerer obligen som rettet og returnerer ordboken med resultater***.**




```
def fordelRetting(self, besvarelser, rettere):
    resultater = {}
    antR = len(rettere)
    rNr = 0
    for sBruker in besvarelser:
        retter = rettere[rNr]
        res = retter.vurder(besvarelser[sBruker])
        resultater[sBruker] = res
        rNr += 1
        if rNr == antR:
            rNr = 0
        # Alternativt: rNr = (rNr + 1) % antR
    self._rettet = True
    return resultater
```

Oppgave 4g)

Viktig: De metodene du allerede har skrevet løser det meste!

*Skriv følgende non-public metoder i klassen **Emne**:*

***_opprettOblig** genererer et unikt oblignavn,*

```
obligId = "oblig" + str(len(self._obliger)+1)
```

og ber bruker på terminalen om å oppgi frist på formen ååmmdd. Deretter legges en ny oblig til emnet.

***_startRetting** ber om dagens dato og sjekker om noen av obligene i emnet har en utløpt frist uten at besvarelsene er rettet. I så fall henter den alle besvarelser for obligen, fordeler dem på rettere og registrerer resultatet av rettingen i Student-objekter (på riktig oblig) for studenter som har levert.*

<for-løkke gjennom obligene>
<hvis klar for retting:>
<for-løkke gjennom hver student i resultatene:>
<slå opp student-objekt og registrer resultatet>

***_skrivEksamensListe** bygger opp en liste med brukernavn for alle studenter som har fått godkjent alle obliger registrert for emnet, og skriver til sist ut denne på terminal med en passende overskrift.*

Tips til oppgaver med flere klasser

- Sjekk mot oppgavetekst / koden din hva metoder du bruker fra andre klassen skal ha som parametere og hva de leverer ut som returverdi: Hva inneholder de, hvilken datatype?

- Bruk gjerne variabel- og parameternavn som indikerer om
  - innholdet er en liste/ ordbok (flertall) eller en enkelt verdi
  - streng/int/... eller referanse til et objekt

  For eksempel:

```
        for sBruker in self._studenter:
            stud = self._studenter[sBruker]
            if stud.altGodkjent(len(self._obliger)):
                eksamensliste.append(sBruker)
```

IN1000-eksamen 2021 – all info

- Digital hjemme-eksamen i eksamenssystemet Inspera, se semestersiden:
  - Tid og sted, oppmelding etc: https://www.uio.no/studier/emner/matnat/ifi/IN1000/h21/eksamen/index.html
  - Flere IN1000-spesifikke detaljer (oppdateres også under eksamen): https://www.uio.no/studier/emner/matnat/ifi/IN1000/h21/eksamensinformasjon-2021/index.html
- "Alt" om Inspera:
  - https://www.uio.no/studier/eksamen/inspera/
- Les forsiden på eksamenssettet nøye (også prøveeksamen)!
  - Hjelpemidler
  - Problemer/ teknisk hjelp/ "trøsterunde"
  - Fusk

Informasjon (må leses)

Skriftlig eksamen i IN1000
2021 HØST
Varighet: 3.12.2021, 9:00 til 3.12.2021, 13:00.
Det er viktig at du leser denne forsiden nøye før du starter.

Del 1&2 (samlet), del 3 og del 4 må bestås hver for seg. Pass på at du bruker tiden slik at du viser hva du kan på hver av disse.

Generell informasjon:

- Viktige beskjeder under eksamen blir gitt direkte fra faglærer her. Det er derfor viktig at du sjekker denne siden jevnlig og før du kontakter faglærer med spørsmål.
- Besvarelsen din skal reflektere ditt eget, selvstendige arbeid og skal være et resultat av din egen læring og arbeidsinnsats.
- Alle hjelpemidler er tillatt ved skriftlig hjemmeeksamen. Dersom du gjengir tekst fra bøker, forelesninger, nettartikler eller lignende, så må det henvises til disse kildene i besvarelsen for å unngå mistanke om ulovlig tekstlikhet. Dette gjelder også dersom du oversetter tekst fra andre språk.
- Du er selv ansvarlig for å sørge for at eksamensbesvarelsen din ikke er tilgjengelig for andre under eksamenstiden, hverken fysisk eller digitalt.
- Husk at besvarelsen skal være anonym, du skal ikke oppgi hverken ditt eller medstudenters navn.
- Om du vil trekke deg fra eksamen, trykk på hamburgermeny oppe til høyre i Inspera og velg "Jeg vil trekke meg".

Samarbeid under eksamen:
Det er ikke tillatt å samarbeide eller kommunisere med andre under eksamen. Samarbeid og kommunikasjon vil bli betraktet som forsøk på fusk. Det blir gjort plagiatkontroll av alle innleverte eksamener der tekstlikhet/ strukturlikhet mellom besvarelser blir sjekket. Om du bruker notater som er utarbeidet i samarbeid med andre før eksamen, kan dette gi treff i en plagiatsjekk. Slik tekstlikhet kan bli betraktet som lav selvstendighet eller forsøk på fusk. Unngå derfor klipp/lim fra notater laget i samarbeid med andre.

Fusk:
Les om hva som regnes som fusk på UiOs nettsider.

Kontaktinfo:
Brukerstøtte eksamen (tekniske/ praktiske/ administrative problemer)
Faglige spørsmål om oppgaven sendes i epost til faglærere (trykk på lenken eller send til siriamj@ifi.uio.no **og** geirksa@ifi.uio.no). Send gjerne bilde av oppgaven du lurer på.

Hjemmeeksamen med alle hjelpemidler

- Semesterside, Internett, egne obliger og notater, bøker
  - Ingen ekstern (fag-relatert) kommunikasjon tillatt

- Sensur vil gjennomføres som om du ikke kan kjøre programmene dine
  - det kreves ikke kjørbar kode, opplagte skrivefeil gir ikke trekk
  - det er oppgaveteksten som stiller krav til programmet, evt tester er kun eksempler til illustrasjon

Det vil si: Du må teste og vurdere nytten av digitale hjelpemidler ift tidsbruk på forhånd, bruk prøveeksamen (for eksempel å skrive i ekstern editor)

Mange opplever knapt med tid

Lag gode rutiner og velg verktøy mens du jobber med prøveeksamen

- Hvis du bruker ekstern editor: Sjekk at kopiering til Inspera fungerer (innrykk, tegnsett, ...)
- Pass på at editoren din ikke genererer kode og importerer biblioteker

På eksamen

- Poeng sier noe om vektingen av (del)oppgavene. Gå videre om du står fast på noe.
  ... MEN HUSK AT HVER HOVEDDEL MÅ BESTÅS (1&2, 3 og 4)

- Det kreves ikke feilfri kode. Vurder tidsbruk om du tester – en ubetydelig feil kan gi lite trekk, men ta mye tid å debugge (obligene tester dette)

- Vi krever ikke kommentarer på eksamen. Kan brukes for antakelser eller å klargjøre noe i løsningsvalget, men husk at sensor kjenner oppgaven godt

Vurdering

- Bestått/ Ikke bestått eksamensresultat

- Må vise ferdigheter innen alle læringsmål
  (krav om "oversiktlige og lesbare programmer" vil kun ansees ikke oppfylt i helt spesielle tilfeller – f. eks. gjennomgående feil/ manglende innrykk, meningsløse variabelnavn ...)

- Alle vil dessuten få tilsendt en tekstlig, automatisk generert tilbakemelding (uio-epost)