IN1010 våren 2026

Faglærerne i IN1010
in1010@ifi.uio.no

Sensorveiledning til eksamen 29. mai 2026

Generelt

I utgangspunktet får man full uttelling for det enkelte delspørsmålet om

- koden er riktig,
- den gjør det oppgaven ber om og
- er oversiktlig og lettlest

Effektivitet er ikke viktig i dette emnet.

Rene skrivefeil ignoreres vanligvis, men feil som kan tyde på manglende forståelse (for eksempel kode som ikke passer inn eller svarer på oppgaven) skal gi trekk.

Dette ignoreres:

- mangler enkelte «`;`» , «`{`» eller «`}`» om koden ellers gir mening
- småfeil i navn

Studentene har fått beskjed om at **`import`**-spesifikasjoner, kommentarer eller tilgangsmodifikatorer (**`public`**/**`private`**/**`protected`**) ikke er viktig på eksamen.

Feil i kode kopiert mellom ulike deloppgaver trekkes kun én gang.

Se også «Generelt om oppgaven» i oppgavesettet.

Poengene/vektene nevnt i denne veiledningen er veiledende og endelig karakter settes etter en helhetlig vurdering.

Oppgave 1 — Klassediagram (5 %)

Her skal studentene vise at de forstår klassehierarkier med klasser og subklasser, og at de kan uttrykke dette i et klassediagram tegnet slik det er vist i forelesningene og annet undervisningsmateriale.

Dette må de ha med:

- alle de nevnte klassene `Familie`, `Taxi`, `Minitaxi`, `Standardtaxi` og `StandardtaxiMedBarnesete` samt interface-et `HarBarnesete`
- viser korrekt hva som er subklasse av hva (dvs at det er en tydelig pil i riktig retning)
- viser at `Familie` ikke henger sammen med de andre klassene
- viser sammenhengen mellom klassen `StandardtaxiMedBarnesete` og interface-et `HarBarnesete`
- angir `Taxi` som abstrakt (teller 1 poeng)

Oppgave 2 — Java-deklarasjoner (11 %)

Her skal studentene vise at de kan programmere klasser og subklasser i Java.

- De må ha med alle klassene nevnt i forrige oppgave. Alle instansvariabler nevnt i oppgaveteksten må være med, og de må ha fornuftige navn. (3 poeng)
- Det er viktig at klassestrukturen i Java samsvarer med klassestrukturen i klassediagrammet i forrige oppgave (1½ poeng, overlapper med forrige punkt).
- Initieringen i konstruktørene må være riktig (dvs kalle `super(...)` for parametre til superklassen) (3 poeng).
- Klassen som implementerer `HarBarnesete` (og bare den) må ha en instansvariabel med riktig antall barneseter (3 poeng). Husk at denne variabelen også skal initieres i konstruktøren.
- Instansvariablene bør angis som **`final`** (½ poeng).
- Antall km i `Familie`-klassen bør deklareres som **`int`** (se oppgave 5), men det er ikke noe trekk om den deklareres som **`double`**.

Oppgave 3 — Sjekkemetode (8 %)

I denne oppgaven er det viktig å få med seg dette:

- Testen kan enten skrives som én litt komplisert metode i `Taxi`-klassen eller som en virtuell metode som redefineres i noen subklasser. Begge varianter er OK.
- Objekter av klasser som implementerer `HarBarnesete` må teste om drosjen har nok barneseter; de andre må sjekke at familen ikke har noen barn med seg.

Oppgave 4 — Lesing fra fil (12 %)

Her skal studentene først og fremst vise at de kan åpne en fil og lese og dekode data i den. Dessuten skal de beherske enkel bruk av `HashMap`.

- Filhåndtering (6 poeng): En `Scanner` er den enkleste måten å lese en tekstfil på, men studentene kan gjerne bruke noe annet så lenge koden er riktig.
- Håndtering av heltall ved fillesing og lagring/uthenting i `HashMap` (innboksing) (2 poeng)
- Metoden `hentTakst` (2 poeng):
  - Oppslag i `HashMap`.
  - Sjekk om nøkkel finnes, ellers returneres 0.
- Kall på `hentTakst` (2 poeng): Her må studentene huske på:
  - Kallene må legges inn i de riktige klassene.
  - Ved kallene må man legge inn håndtering av `FileNotFoundException`.

Oppgave 5 — Beregne pris (9 %)

Denne oppgaven kan enten løses med bare én metode i `Taxi`-klassen eller med virtuelle metoder i subklassene; begge varianter er helt OK så lenge barnesetene håndteres riktig.

Oppgave 6 — En taxikø (10 %)

Denne oppgaven skal vise om studentene kan håndtere en `ArrayList` og om de kan løpe gjennom en liste.

Oppgave 7 — Rekursjon (12 %)

Her sjekkes om studentene behersker rekursjon.

- En ikke-rekursiv løsning gir bare 2 poeng, selv om den skulle være korrekt.
- Et sluttest må være med og gir opp til 4 poeng.
- Et rekursivt kall det også være, og det teller opp til 4 poeng.
- De resterende 4 poeng gis ut fra hvor riktig koden er.

Oppgave 8 — Fjerning fra liste (9 %)

Her skal det sjekkes om studentene behersker dobbeltlinkete lister.

- Korrekt behandling av både `forrige`- og `neste`-pekere teller 4 poeng.
- Om første taxi fjernes, må `førsteTaxi` endres (1 poeng).
- For øvrig gir riktig håndtering av første og siste taxi i køen 4 poeng.
- Intet trekk for å glemme å sette `forrige`- og `neste`-pekerne i det fjernete objektet til **`null`**.

Oppgave 9 — GUI (11 %)

Her skal studentene vise at de kan opprette et GUI-vindu med tekst, felter for å skrive tekst og til sist en trykknapp.

- Brukerinteraksjonen med **Bestill**-knappen (dvs `ActionListener`-klassen og `actionPerformed`-metoden) er viktigst og teller 6 poeng.
- Innholdet i `actionPerformed`-metoden bør være ganske rett frem og teller 2 poeng.
- Det skal gis inntil 3 poeng for å få riktig struktur på GUI-oppsettet med:
  - et vindu (en `JFrame`)
  - to tekster (`JLabel`)
  - to felt for å skrive tekst (`JTextField`)
  - en trykknapp (`JButton`)

  Mindre feil i bruk av, eller navn på, klasser og metoder i Swing trekkes maksimalt 1 poeng. Det er ikke et krav at det brukes `GridLayout`.

Oppgave 10 — Tråder (13 %)

I denne oppgaven skal studentene få mulighet til å demonstrere at de behersker trådprogrammering. Den inneholder også litt listeprogrammering.

- Låsing og åpning av metodene `settInnSistIKøen` og `hentBilligsteTaxi` er viktig og teller 4 poeng.
- Låsing av andre metoder som kalles av de to, gir 1 poeng trekk.
- Riktig venting med `Condition`-objekt teller 4 poeng.
- Riktig håndtering av unntak teller 1 poeng.
- Korrekt innsetting sist i listen teller 4 poeng, men det gis 1 poeng trekk ved manglende test på om listen er tom.