

IN1010 Objektorientert programmering – eksamen våren 2023

Oppgavetekst fra UiOs eksamen våren 2023, IN1010 (overskriften er eksamenssett.no sin; utsnittene under er klippet uendret ut av UiOs publiserte dokument)

Eksamen IN1010, våren 2023

I denne oppgaven skal du skrive deler av et lite program som behandler tog med vogner og lokomotiv.

Les gjennom hele oppgaven før du begynner på besvarelsen. Tegning av klassehierarkiet skal leveres i oppgave 1, samt alle andre svar (det vil si all programkode) skrives i svarfeltet for oppgave 2-5. Angi gjerne hvilken oppgave du svarer på som kommentarer i programkoden.

Du kan bruke norske bokstaver i denne besvarelsen.

Oppgave 1. Klassehierarki for skinnegående materiell (5 poeng)

Alle vogner og annet materiell som kan kjøre på en skinnegang, representeres av klassen **Skinnegående** som har følgende metoder:

hentId returnerer en String som identifiserer dette skinnegående objektet.

hentSporvidde returnerer et heltall som angir hvor brede spor det kan kjøre på.

Sporvidden og identifikatoren skal være parametere til klassens konstruktør. Skinnegående objekter skal kunne hektes sammen i en dobbeltlenket liste i en senere oppgave, så du skal også ha med instansvariabler i klassen som gjør dette mulig.

I dette eksamenssettet er alt skinnegående materiell enten av klassen **Lokomotiv** eller klassen **Vogn**. Alle Vogn-objekter er enten av klassen **Godsvogn** eller klassen **Passasjervogn**.

Skinnegående materiell kan implementere et interface **Motordrevet** med følgende grensesnitt:

fossilt som returnerer true dersom fossilt brennstoff (bensin, diesel, kull, ved) benyttes, ellers false.

trekkraft som returnerer antall kW (kilowatt) som et heltall.

Alle vogner har en lengde angitt som heltall i centimeter. Godsvogner har en maks tillatt lastevikt, angitt som et desimaltall i kilogram, og passasjervogner har et heltall som representerer maks antall passasjerer det er plass til i vognen. Disse instansvariablene skal være parametere til konstruktørene til klassene.

⇒ Tegn et klassehierarki for situasjonen beskrevet over.

Oppgave 2. Programmere klasser og interface (15 poeng)

I denne oppgaven skal du programmere klasser og interface som beskrevet i oppgave 1.

⇒ Programmer klassene **Skinnegående**, **Lokomotiv**, **Vogn**, **Godsvogn**, **Passasjervogn** og interface-et **Motordrevet**.

Oppgave 3. Klassen Tog (40 poeng)

I oppgave 3 skal du skrive en klasse **Tog** med variabler og metoder. Et tog er en dobbeltlenket liste av Skinnegående-objekter og du skal bruke instansvariablene i disse objektene til å lage denne listen. I denne oppgaven kan du aksessere instansvariabler i Skinnegående-klassen og dens subclasser.

3a) 1 poeng

- ⇒ Skriv konstruktør og instansvariabler for klassen Tog med pekere til første og siste Skinnegående-objekt i listen.

3b) 4 poeng

- ⇒ Skriv metoden **leggTil** i klassen Tog som legger til et Skinnegående objekt bakerst i listen. Parameter til metoden er en referanse til objektet som skal legges til.

3c) 7 poeng

- ⇒ Skriv metoden **taUt** i klassen Tog som tar ut et element i listen. Parameter til metoden er en referanse til objektet som skal tas ut. Du kan anta at objektet som tas ut, ligger i toget. Metoden skal returnere en referanse til det objektet som tas ut.

3d) 7 poeng

- ⇒ Skriv metoden **finnOgTaUt** i klassen Tog som finner et element i listen og tar det ut. Parameter til metoden er en String som er id-en til det Skinnegående-objektet som skal tas ut (du kan anta at id-en er unik). Metoden skal returnere en referanse til det objektet som tas ut.

3e) 7 poeng

- ⇒ Skriv metoden **leggTilForan** i klassen Tog som har to parametere av typen Skinnegående. Den første parameteren er en referanse til et objekt du kan anta allerede ligger i toget, den andre er en referanse til et objekt som skal settes inn i listen foran objektet som refereres av den første parameteren.

3f) 7 poeng

- ⇒ Skriv en iterator over alle Skinnegående-objektene i et Tog.

3g) 7 poeng

- ⇒ Skriv metoden **Passasjervogn[] hentPassasjervogner ()** i klassen Tog som returnerer en array med referanser til alle Passasjervogn-objekter i toget.
Hint: Tell antall passasjervogner før du oppretter arrayen.

Oppgave 4. Sporvidde (20 poeng)

Kun vogner og lokomotiv med samme sporvidde kan brukes sammen i et tog. Dersom sporvidden ikke er den samme for alle vognene og lokomotivene, så skal et unntak ("Exception") av typen **FeilSporvidde** kastes. I oppgave 4 skal du skrive kode for ulike måter å kontrollere dette på.

4a) 1 poeng

- ⇒ Skriv unntaket FeilSporvidde.

4b) 7 poeng

Vi ønsker å kunne sjekke at alle objektene i et tog har samme sporvidde etter at de er lagt inn. Skriv en iterativ (ikke rekursiv før i 4d) metode **sjekkSporvidde** som går gjennom hele toget og sjekker at alle vognene har samme sporvidde. Hvis programmet oppdager et objekt med en annen sporvidde, skal unntaket du laget i oppgave 4a kastes.

⇒ Skriv metoden sjekkSporvidde i klassen Tog.

4c) 4 poeng

Bruk unntaket du laget i oppgave 4a) til å lage en **leggTilSikker**-metode som setter inn et nytt objekt bakerst i listen (slik som i oppgave 3b) og som kaster et unntak når en vogn eller et lokomotiv med en annen sporvidde enn de som allerede er i toget, blir forsøkt lagt til.

⇒ Skriv metoden leggTilSikker i klassen Tog

4d) 8 poeng

⇒ Skriv en rekursiv løsning for å sjekke at alle objektene i et tog har samme sporvidde, inkludert eventuelle endringer eller tillegg du vil gjøre i svar på tidligere oppgaver.

Oppgave 5. Tråder (20 poeng)

I oppgave 5 skal du skrive én trådklasse som heter **Leter** og én som heter **Resultat**. Begge disse skal implementere Runnable. Du skal også skrive en klasse **Monitor** som skal være en monitor.

(Du skal altså ikke skrive noe fullstendig program som starter opp noen tråder – du skal bare skrive disse tre klassene med metoder og variabler.)

5 a) 5 poeng

Leter-trådene skal lete gjennom hvert sitt tog. De skal lete etter alle skinnegående objekter der Id-en begynner med en bestemt tekststreng. Hver gang en Leter-tråd har funnet et skinnegående objekt som passer, skal den legge en referanse til dette inn i monitoren ved å kalle metoden **leggTil** i monitoren. Når en Leter-tråd har lett gjennom hele sitt tog, skal den si fra til monitoren om dette ved å kalle metoden **ferdigLeting** i monitoren og så terminere.

Som parameter til Leter-klassens konstruktør skal du ha en referanse til toget tråden skal lete i, en referanse til monitoren og en referanse til String-objektet det skal letes etter.

Hint: Metoden `startsWith(String s)` i String-klassen returnerer true hvis String-objektet den kalles på, begynner med s.

⇒ Skriv trådklassen Leter

5 b) 3 poeng

Det skal bare være én instans av tråden Resultat. Resultat-tråden skal hente Skinnegående-objekter fra monitoren ved å kalle metoden **hentNeste** i monitoren og skrive ut Id-en i terminalvinduet ved hjelp av `System.out.println`. Hvis hentNeste returnerer null er det ikke flere Skinnegående- objekter igjen å skrive ut, og tråden kan terminere. Resultat-tråden skal kunne startes opp samtidig med Leter-trådene og således kunne utføres i parallell med Leter-trådene.

Resultat-klassens konstruktør skal bare ha en referanse til monitoren som parameter

⇒ Skriv trådklassen Resultat

5 c) 12 poeng

Monitoren skal inneholde de tre metodene `leggTil`, `ferdigLeting` og `hentNeste` som beskrevet over. Som parameter til monitorens konstruktør skal du bare ha hvor mange `Leter-tråder` som finnes.

⇒ Skriv `Monitor`-klassen

Kilde: Oppgaveteksten er gjengitt ordrett fra Universitetet i Oslo, Institutt for informatikk: «Eksamen IN1010, våren 2023 – Oppgavetekst med sensorveiledning» ([eksamen2023-med-sensorveiledning.pdf](#)), publisert på [uio.no](#) under IN1010 v23 – Eksamensinfo 2023. UiO har ikke publisert oppgavesettet separat; dette arket inneholder bare oppgaveteksten, klippet ut av dokumentet uten endringer. Sensorveiledningens tekst er utelatt. Avsnittet «(Du skal altså ikke skrive noe fullstendig program ...)» står nederst på side 6 i kilden og er her plassert etter innledningen til oppgave 5. Satt opp av [eksamenssett.no](#), som ikke er tilknyttet UiO.