UiT Norges arktiske universitet

Kort hjemmeeksamen i:	INF-1400 Objektorientert Programmering
Dato:	2022-05-20
Tidspunkt:	09:00-13:15
Kursansvarlig:	John Markus Bjørndalen Rune Bostad Henrik Hillestad Løvold
Antall sider:	3
Support:	Du kan ringe 776 20 880 for support på eksamensdagen. Henrik Hillestad Løvold: 478 10 167
Vekting av spørsmål, eller annen informasjon:	
Viktig informasjon om sitering og plagiering:	1. Dette er en individuell eksamen som skal besvares uten samarbeid med andre. 2. Alle hjelpemidler er tillatt (egne notater, pdfer fra forelesningene, lærebok, internett etc). 3. Alle eksamener som leveres i WISEflow blir automatisk sjekket for plagiat. Det er ikke tillatt å kopiere medstudenter, nettressurser, kilder, eller litteratur uten referanser.

Praktisk informasjon

- Du kan levere pseudokode. Det er ikke behov for å lage en kjørende løsning. Vi er ute etter å se mønster og tenking. Prøver du å lage en kjørende løsning kommer du raskt til å bruke mer tid enn det vi krever.
- Ikke overdriv kompleksiteten i løsningen (f.eks. modellere ting som ikke er beskrevet i oppgaven). Vi forventer ikke en komplett løsning. Det betyr at noen ting kan vøre litt urealistiske. Fokuser på hvordan du skal jobbe med koden og designet på programmet.

Bakgrunn

Et firma som driver med kjøp og salg av kjøretøy, lagrer kjøretøyene i garasjer. Firmaet har behov for å holde oversikt over hvor de enkelte kjøretøyene er plassert.

I denne oppgaven skal du simulere at bilene blir plassert og hentet ut av forskjellige garasjer. I koden du har fått utdelt, kan du se at ulike kjøretøy har forskjellige egenskaper (Personbil og Varebil). Den utdelte koden (precode.py) er skrevet sekvensielt, uten bruk av objektorientering. Det er din oppgave å vurdere koden, og omstrukturere den slik at systemet benytter seg av objektorienterte prinsipper.

Oppgave 1

Lag et klassediagram for de klassene du tenker er naturlige å ta med, inkludert relasjoner mellom klassene. Løsningen skal demonstrere arv og polymorfi (hvis du mener dette gir deg en mindre bra løsning, kan du gjerne kommentere dette og hvordan du ellers ville gjort det som et tillegg).

Oppgave 2

Beskriv hvordan oppførselene kan implementeres og grove trekk for hvordan du endrer den eksisterende koden. Beskriv antagelsene du gjør. Forklar hvordan arv og polymorfi brukes i løsningen.

Oppgave 3

Implementer det du har skissert i oppgave 2 (ikke bruk tiden din på å debugge småfeil).

Oppgave 4

For personbiler ønsker vi å skille mellom fossilbil og elbil. Hvor antall liter drivstoff er viktig for fossilbil mens antall kWh batteriet rommer er viktig for elbil.

1) Vil du plassere disse i klassehierarkiet, og i så fall hvor?
2) Hvilke nye attributter og metoder behøver disse klassene, sammenliknet med Personbil-klassen i hierarkiet ditt?
3) Implementer klassene.