Eksamen 2025 – Objektorientert Programming

Rapport

Innkapsling: I dette prosjektet har jeg tilegnet meg gode kunnskaper fra dette emne. Innkapsling har gitt en god forståelse ved å gjøre alle instansvariabler private i klassene mine. Tilgang til disse variablene skjer gjennom *gettere* som jeg har brukt. Dette gir full kontroll over hvordan data blir lest og endret. Det sikrer at objektets tilstand ikke blir manipulert direkte utenfor klassen, noe som gir bedre kontroll og mindre feil. Jeg har også laget egne klasser med spesifikke ansvarsområder, som *DataInsert* og *VehicleFileReader.* Jeg bruker begge klassene for å kapsle inn logikk og følge prinsippet om lav kobling. Dette har gjort koden lett å forstå ved å teste og vedlikeholde.

Arv og polymorfi: Jeg har brukt arv ved å lage en abstrakt superklasse *Vehicle*. Denne klassen inneholder felles felter og metoder for alle kjøretøy. Klassene som *FossilCar, ElectricCar, Motorcycle* arver fra Vehicle. og legger til sine unike egenskaper. Polymorfi kommer til uttrykk når kjøretøyene håndteres i lister av typen List<Vehicle> i klassen *MenuProgram*, og i metoder som sjekker objekttypen instanceof i *DatabaseInsert*. På denne måten kan koden håndtere ulike kjøretøytyper på en fleksibel måte. Jeg har også overskrevet *toString()* i subklassene, slik at hvert av kjøretøyene kan presentere seg selv på en måte som passer sin type.

Refleksjon: En utfordring jeg møtte på var å få registrert *RegistrationNumber og ChassisNumber* korrekt i databasen. Jeg forsøkte å endre på parameterne i *insertFossilCar* metoden i klassen, *DatabaseInsert,* men jeg fikk det ikke til å fungere som planlagt. Dette er noe jeg ønsker å forbedre, men samtidig er jeg fornøyd med hvordan jeg har strukturert prosjektet med egne pakker for modellklasser, databasehåndtering og filinnlesing. Jeg er også fornøyd med hvordan jeg har brukt arv og polymorfi i prosjektet, blant annet ved å bruke instanceof i databasen for å skille mellom ulike kjøretyper. Dette viser at jeg forstår viktige prinsipper og logikken i objektorientert programmering.