UKE 7 SEMINAR

KLASSER OG OBJEKTER

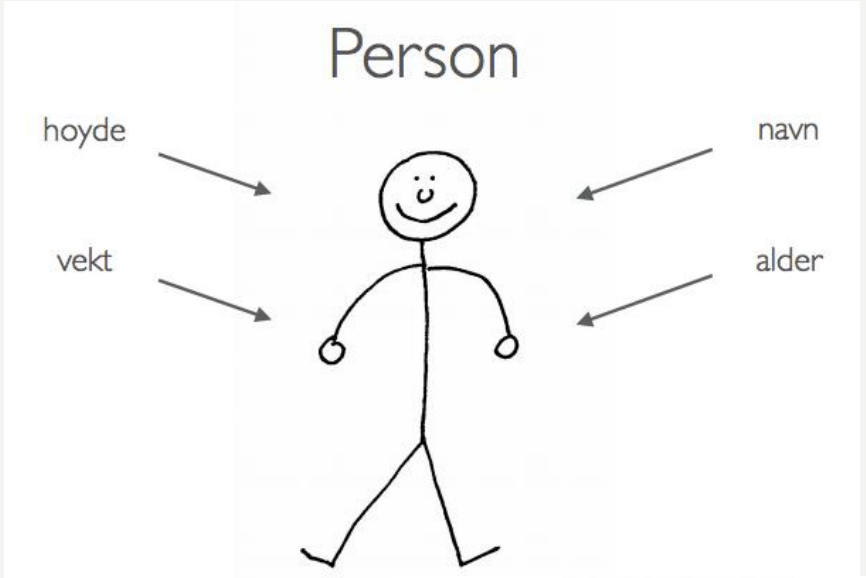
OBJEKTORIENTERT PROGRAMMERING

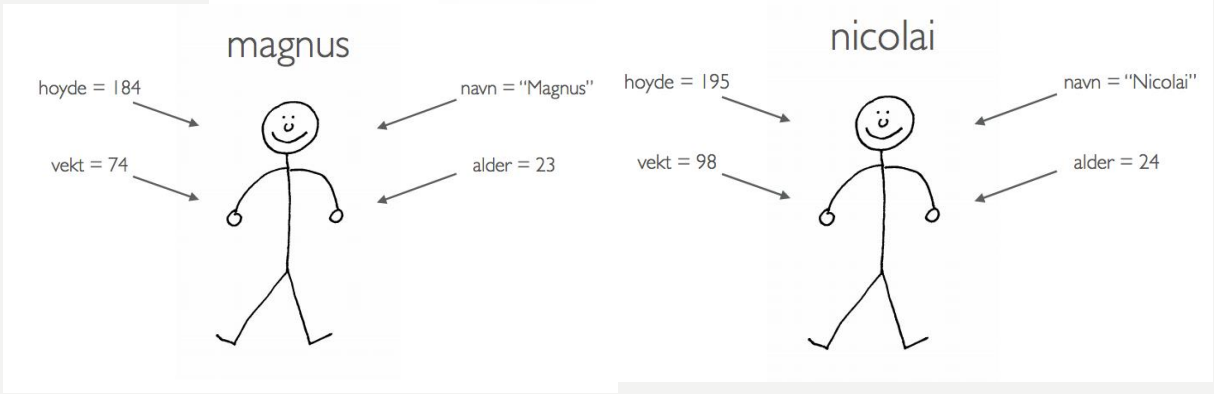
- Handler om å modellere verden
  - Vi kan lage en modell av noe som finnes i verden
  - Denne modellen har egenskaper og handlinger den kan gjøre

- Eks:
  - En person har egenskaper: navn, alder, høyde osv.
  - En person har handlinger: si navnet sitt, bli eldre, vokse høyere osv.

KLASSER OG OBJEKTER

- En klasse er en mal
- Vi kan lage objekter av en klasse
  - Eks: Person er en klasse.
    Klassen har egenskaper og handlinger som er felles for alle personer
  - Objektene er Magnus og Nicolai.
    De er instanser av klassen Person.
    De er ulike objekter av samme klasse.





INTERFACE/GRENSESNITT

Metode:
en prosedyre eller funksjon som hører til en klasse

- Alle klasser har et public interface – et grensesnitt
- Grensesnittet inneholder metodene i klassen

- Eks:
  - Vi har et objekt «bil», og den har en metode kjoer(km).
  - Vi vet ikke hvordan metoden er definert, men vi skjønner hva den gjør.
    Den tar inn hvor mange km bilen kjører, og kanskje oppdaterer kilometerstanden og drivstoffet.
  - Vi vet ikke hvordan det gjøres, vi vet bare at det skjer når vi kaller metoden.
    - Dette er **innkapsling**!

SKRIV EN KLASSE

- Definer klassen
- Definer konstruktøren og instansvariablene
- Definer metodene

- Instansvariabler skrives med self._ foran
  - De er tilgjengelige overalt i klassen, men ikke utenfor klassen
- Metoder som ikke skal aksesseres utenfra, skrives også med _ foran

```python
# Definer klassen
class Person:

    # Konstruktør
    def __init__(self, navn, alder):
        # Instansvariabler
        self._navn = navn
        self._alder = alder

    def si_hei(self):
        print("Hei")
```

KONSTRUKTØR

- Bestemmer hva som må være med ved opprettelsen av ett objekt
- Heter alltid __init__ med to understreker på hver side av init.

- Eks: Å lage en person uten navn er litt rart
  - Vi kan kalle "settNavn"-metoden hver gang en person opprettes, men mye bedre å bestemme at navn MÅ oppgis når objektet opprettes.
- Navn kan sendes med som parameter til konstruktøren
  - Parametere må lagres i instansvariabler

```
# Konstruktør
def __init__(self, navn, alder):
    # Instansvariabler
    self._navn = navn
    self._alder = alder
```

OPPGAVE 1 I PAR

- Lag en klasse person basert på tegningene fra sliden.

- Klassen skal ha metodene:
  - settNavn(self, navn) -> setter navnet på personen
  - skrivNavn(self) -> skriver ut navnet å personen på dette formatet "Jeg heter <navn>"
  - settAlder(self, alder) -> setter aldren på personen
  - settVekt(self, vekt) -> setter vekten på personen
  - settHoyde(self, hoyde) -> setter høyden på personen
  - skrivUtInfo(self) -> som skriver ut infoen om personen
  - skrivUtHilsen(self) -> som skriver ut en hilsen fra personen, på formen "Hei, jeg heter <navn> og jeg er <alder> aar gammel"
  - hoyereEnn(self, annenPerson) -> tar inn en annen person og returnere True hvis personen er høyere enn den andre personen, False ellers

OPPGAVE 2 I PAR

- Endre oppgave 1 slik at den nå i stedet for å ha metodene settNavn, settAlder, settVekt og setHoyde tar dette inn og setter disse (instans)variablene i konstruktøren istedenfor?

- Hvilke fordeler/ulemper har dette? Diskuter.

OPPRETT OG BRUK OBJEKTER

- Lag en ny fil, f.eks. hovedprogram.py
- Lag en prosedyre som skal holde på koden, f.eks. hovedprogram()
- Lag objektene og sørg for å lagre dem i variabler eller i en samling

```
def hovedprogram():
    person1 = Person("Kari", 20)
    person2 = Person("Ola", 23)
```

EN KLASSE PER FIL

- Det er mulig å skrive flere klasser i en fil, men i IN1000 vil vi helst ha en klasse per fil.
- Filnavn skrives med liten forbokstav og klassen skrives med stor
  - Ha gjerne samme navn på filen og klassen
  - Eks: filnavn person.py - klasse Person

- Lag en egen fil til hovedprogrammet
  - Vi må importere klasser når vi skal bruke dem i en annen fil
  - Syntaks: <from filnavn import Klasse>

```
from person import Person
```

OPPGAVE 3 I PAR

- Gitt klassen person som dere har skrevet, opprett 3 instanser av klassen/objekter med navn/alder/vekt/hoyde du velger selv. Skriv deretter ut en hilsen fra hver person ved hjelp av skrivUtHilsen-metoden. Lagre deretter alle objektene i en liste.

- Utfordring: se om du kan kalle på "skrivUtHilsen()"-metoden i en for-løkke

- Utfordring: lag en funksjon som tar inn en liste med personer, og returnerer den personen som er høyest

OPPGAVE 4 I PAR

- Skriv en klasse Dyr.
- Et dyr har følgende egenskaper:
  - Art
  - Kjønn
  - Vekt

- Lag en konstruktør (init-metode) og metoder for å hente ut data fra hver instansvariabel.

- Opprett tre objekter av klassen Dyr for å representere tre forskjellige dyr.
- Skriv ut informasjon om alle dyrene.

STØRRE OPPGAVE

- Lage en klasse Bygning i filen bygning.py.
- Bygningen skal ha en adresse, en huseier og antall leietakere som bor der nå.
- Det skal være metoder for at leietakere kan flytte inn og flytte ut.
  Huseier skal også kunne endres (ved salg av bygningen).

- Utfordring:
  Det er urealistisk å ha negativt antall leietakere, eller flere leietakere enn antall leiligheter.
  Legg inn sjekker for at dette ikke skal gå.