IN1000

Uke 7

Plan for dagen

- Hvis dere vil ha utsettelse: studieinfo@ifi.uio.no
- OOP - objektorientert programmering
- Klasser og objekter
- Innkapsling og grensesnitt

Klasser og objekter

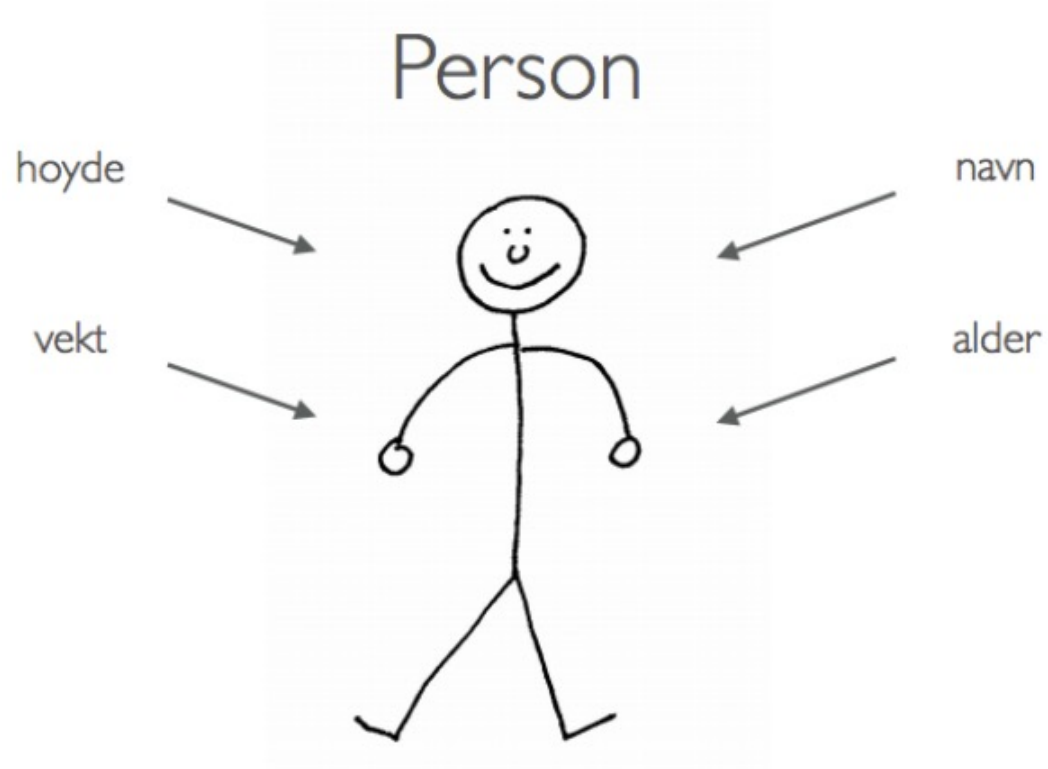
- Handler om å «modellere» verden

- Tenk på en klasse som en gruppe, og et objekt som et medlem av denne gruppen

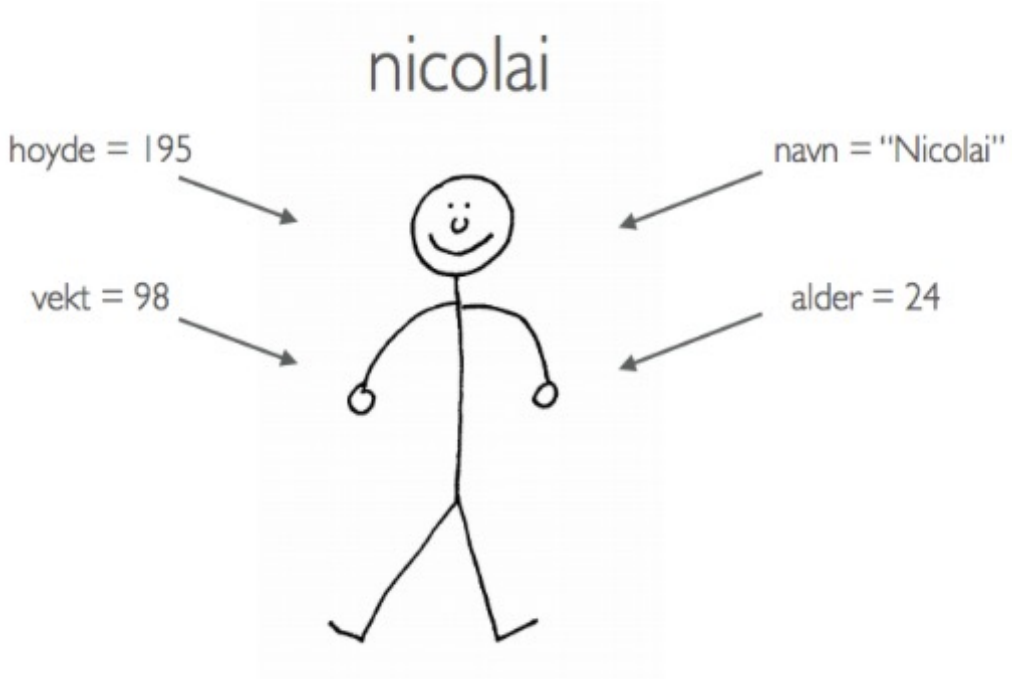
Klasser og objekter

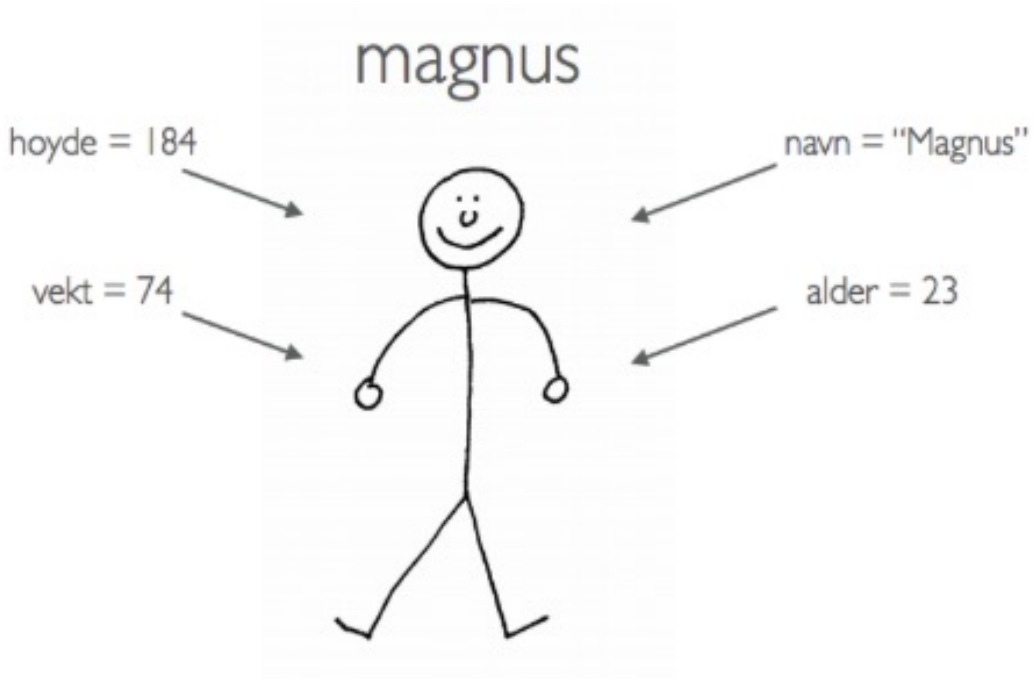
- Klassen person:

- Egenskaper alle objekter av en klasse har (instansvariabler)



Eksempler på objekter av klassen Person





Syntaks (generell)

```
class Klasse:
    def __init__(self, parameter1, parameter2):
        self._egenskap1 = parameter1
        self._egenskap2 = parameter2
```

Oppgave!

```
class Klasse:
    def __init__(self, parameter1, parameter2):
        self._egenskap1 = parameter1
        self._egenskap2 = parameter2
```

- Opprett en fil person.py, og lag en klasse Person med instansvariablene **navn**, **alder**, **høyde** og **vekt**

Syntaks (Person)

```
class Person:
    def __init__(self, hoyde, vekt, navn, alder):
        self._hoyde = hoyde
        self._vekt = vekt
        self._navn = navn
        self._alder = alder
```

Syntaks (Person)

```
class Person:
    def __init__(self, hoyde, vekt, navn, alder):
        self._hoyde = hoyde
        self._vekt = vekt
        self._navn = navn
        self._alder = alder
```

Klassenavn

Instansvariabler

Konstruktør

Metoder

- Er funksjoner eller prosedyrer som er inne i en klasse!

- Generell syntaks:

```
class Klasse:
    # Konstruktor med instansvariabler

    def metode_navn1(self, parameter1, parameter2):
        # gjor noe

    def metode_navn2(self, parameter1, parameter2):
        # gjor noe
```

Oppgave!

```python
class Klasse:
    # Konstruktor med instansvariabler

    def metode_navn1(self, parameter1, parameter2):
        # gjor noe
```

- Vi skal nå utvide Person-klassen med følgende metoder:

1. endre_navn(self, nytt_navn):

   # endre navnet til nytt_navn

2. skriv_ut_hilsen(self):

   # skriv ut en hilsen fra personen på formen «Hei, jeg heter <navn> og er <alder> år gammel.»

3. feire_bursdag(self):

   # Øk alderen til personen med 1.

   # Skriv ut en bursdagshilsen med den nye alderen («Hurra, <navn> blir <år> år!»).

   # Hvis personen fremdeles er under 18 år, skal høyde økes med 2.

Løsningsforslag

```python
class Person:
    def __init__(self, hoyde, vekt, navn, alder):
        self._hoyde = hoyde
        self._vekt = vekt
        self._navn = navn
        self._alder = alder

    def endre_navn(self, nytt_navn):
        self._navn = nytt_navn

    def skriv_ut_hilsen(self):
        print("Hei, jeg heter", self._navn, "og jeg er", self._alder, "år gammel")

    def feire_bursdag(self):
        self._alder += 1

        print("Hipp hurra!", self._navn, "blir", self._alder, "år!")

        if self._alder < 18:
            self._hoyde += 2
```

Begreper: innkapsling og grensensitt

- Alle klasser har et public (tilgjengelig utenfra) interface, et **grensesnitt**, altså metodene som er tilgjengelige for hvert objekt

- **Innkapsling**: vi vet ikke hvordan metoden fungerer innvendig, vi bare vet at den utfører funksjonen
  - Bil-analogi

Hvordan bruke en klasse?

- Last inn klassen i en test-fil, sånn her:

  from <filnavn> import <Klasse>

  f.eks: from person import Person

- Opprett et objekt:

  nikolai = Person(180, 80, «Nikolai», 20)

- Kall på en metode, slik:

  nikolai.endre_navn(«Ole»)

Oppgave!

- Gitt klassen Person som vi har laget:

1. Opprett 3 objekter av klassen med navn/alder/vekt/hoyde
2. Skriv deretter ut en hilsen fra hver person ved hjelp av .skriv_ut_hilsen()
3. Feir bursdagen til én av personene, og skriv ut en ny hilsen fra denne personen
4. Endre navnet til én av personene og skriv ut en ny hilsen fra denne personen

Løsningsforslag

```python
from person import Person

nikolai = Person(180, 80, "Nikolai", 20)
siri = Person(172, 75, "Siri", 50)
geir_kjetil = Person(182, 82, "Geir Kjetil", 38)

nikolai.skriv_ut_hilsen()
siri.skriv_ut_hilsen()
geir_kjetil.skriv_ut_hilsen()

siri.feir_bursdag()
siri.skriv_ut_hilsen()

geir_kjetil.endre_navn("Geir")
geir_kjetil.skriv_ut_hilsen()
```