Uke 8 - konsept

Gruppe 14

Kristin - kristlgj@ifi.uio.no

Menti

Plan for gruppetimen

- Forskjell på instansvariabler, lokale variabler og parametere
- Forskjellen på variabel, referanse, objekt (Minnegaten-presentasjon)
  - Når endrer vi variabel og når endrer vi objekt?
- Kort om immutable objects, None, og self
- Datastrukturtegning

Menti

Læringsmål uke 8

- Forstå (mer av) hva som skjer bak kulissene når vi oppretter og bruker objekter
- Kjenne til forskjellen på å endre en referansevariabel og å endre objektet den refererer til
- Kunne skrive programmer med samlinger av (referanser til) objekter
- Kunne forstå og tegne datastruktur-tegninger
- Kunne sette seg inn i og skrive mindre programmer med flere klasser, og med objekter som refererer til andre objekter

Hva ble gjennomgått på forelesningen?

Tenk selv, så diskuter

- Hva lærte du, hva skjønte du ikke?
- Se gjerne over slides fra forelesningen om dere ikke husker!
- Er det noe dere lurer på?

Menti

Forskjell på instansvariabler, lokale variabler og parametere i en metode

- **Instansvariabler** - kan aksesseres i hele klassen (på tvers av metoder)
- **Lokale variabler** - kan bare aksesseres i egen metode, oppstår og dør hver gang metoden kalles (finnes kun mens metoden utføres)
- **Parametere** - oppfører seg som lokale variabler men initialiseres med argumentene i metodekallet (får verdi "utenifra")

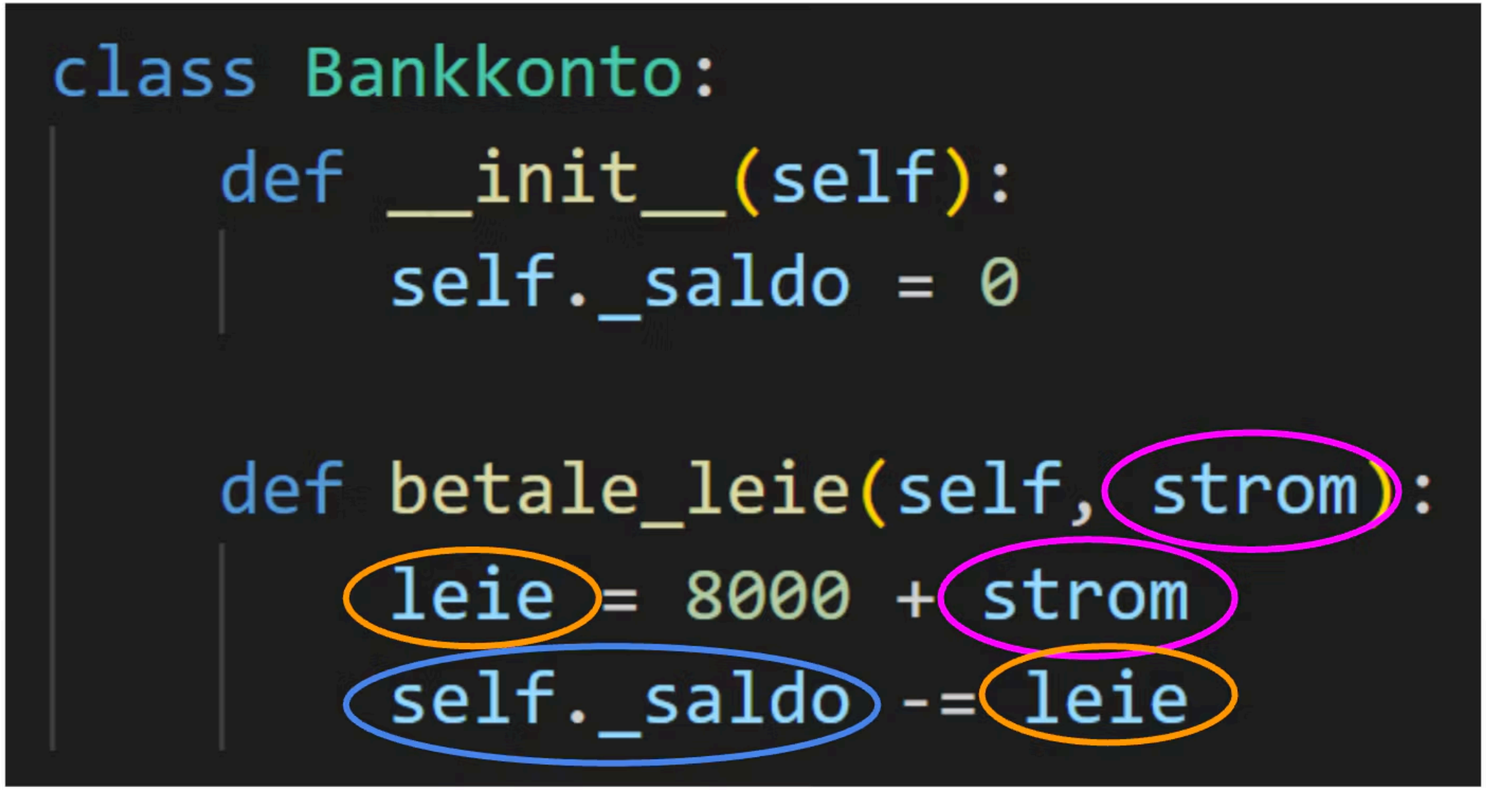
Menti

```
class Bankkonto:
    def __init__(self):
        self._saldo = 0

    def betale_leie(self, strom):
        leie = 8000 + strom
        self._saldo -= leie
```

Hva i betal_leie() er instansvariabel, lokal variabel, og parameter?

Menti



instansvariabel lokal variabel parameter

Menti

Variabel, referanse, objekt

se egen presentasjon (Minnegaten_IN1000)

Hva er forskjellen?

Variabler inneholder referanser til objekter

Variabel: merkelapp vi setter på en verdi

Referanse: hvor verdien ligger i minnet

Objekt: selve verdien

Menti

Når endrer vi variabel og når endrer vi objekt?

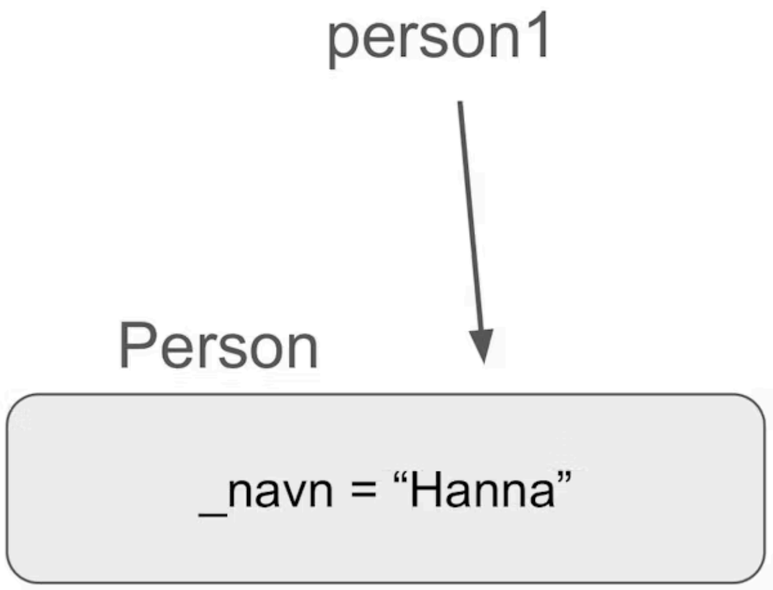
```
person1 = Person("Hanna")
person2 = Person("Erik")

person1.endre_navn("Hanne") # Endrer objektet
person1 = person2           # Endrer variabelen
```

Menti

```python
person1 = Person("Hanna")
person2 = Person("Erik")

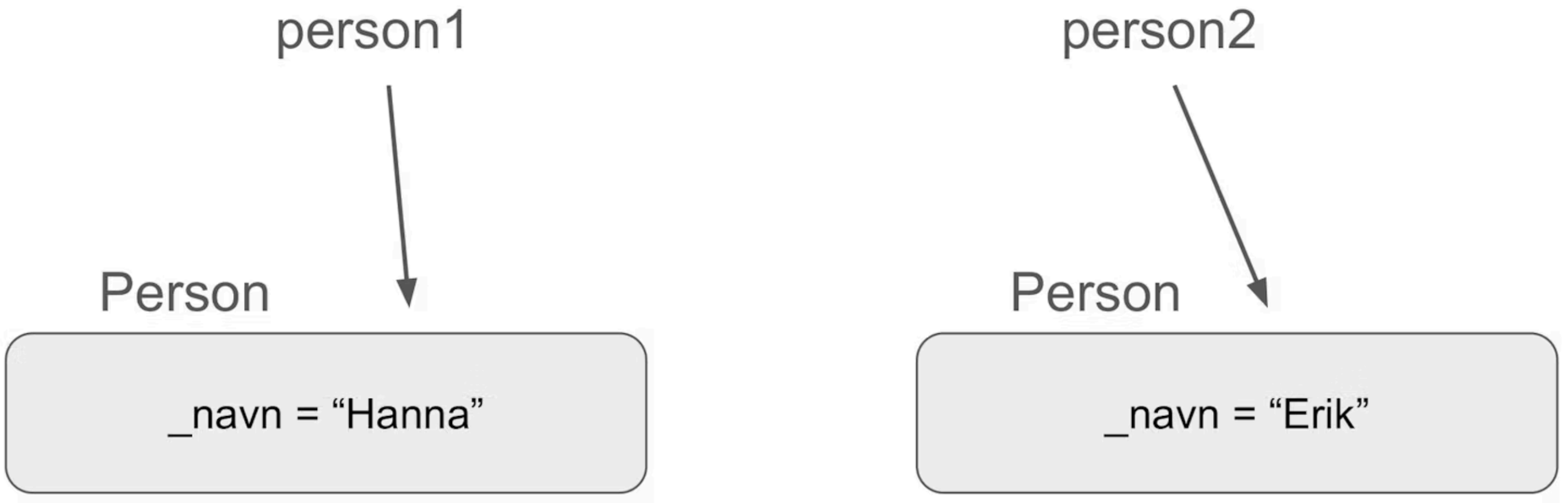
person1.endre_navn("Hanne") # Endrer objektet
person1 = person2           # Endrer variabelen
```



Menti

```python
person1 = Person("Hanna")
person2 = Person("Erik")

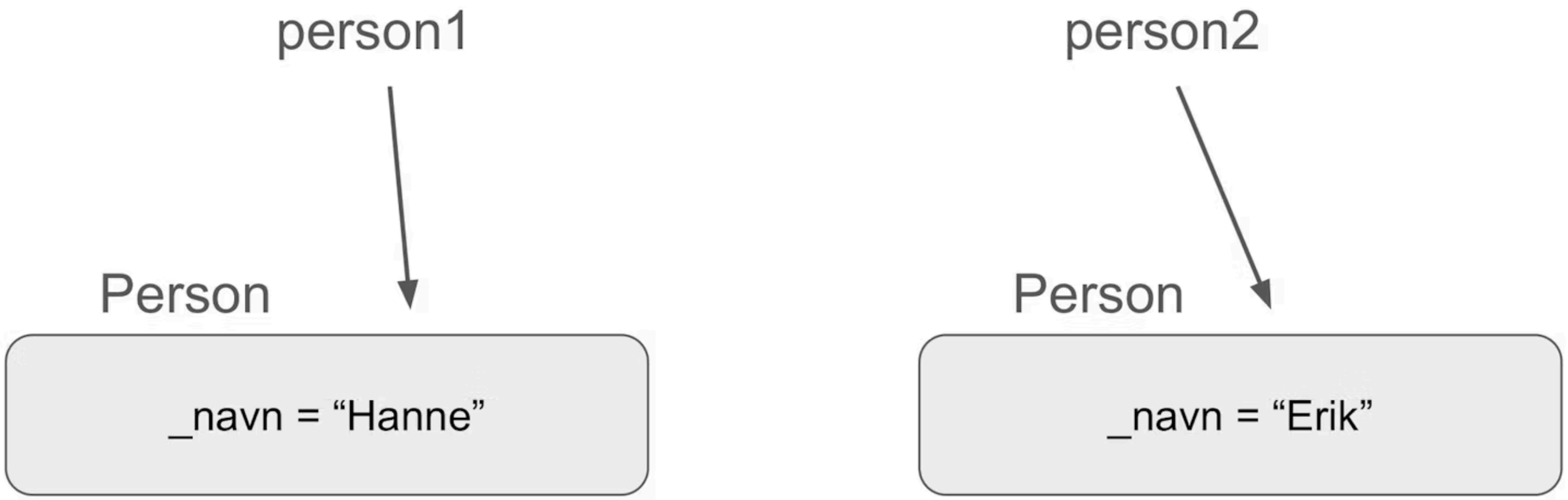
person1.endre_navn("Hanne") # Endrer objektet
person1 = person2           # Endrer variabelen
```



Menti

```python
person1 = Person("Hanna")
person2 = Person("Erik")

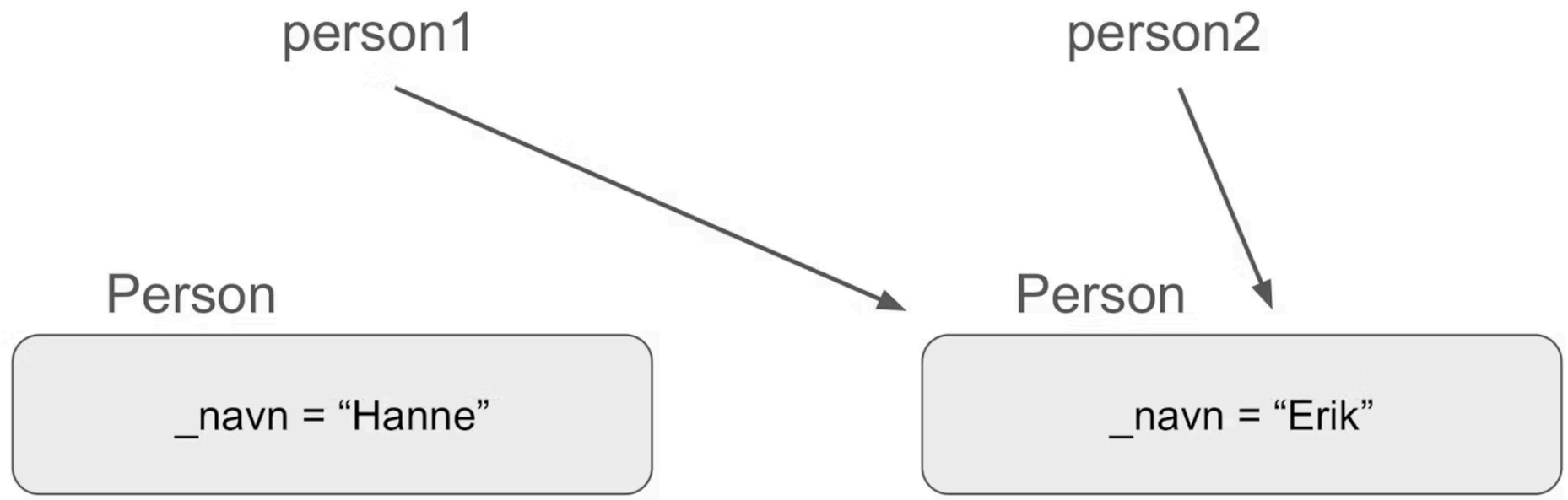
person1.endre_navn("Hanne") # Endrer objektet
person1 = person2           # Endrer variabelen
```



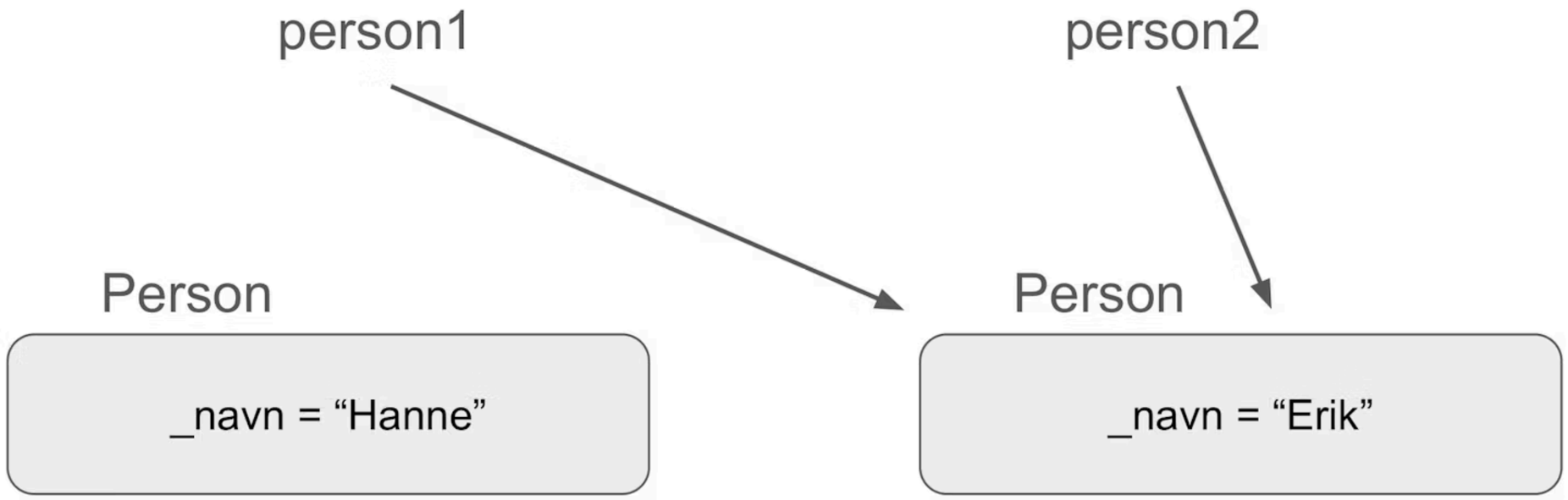
Menti

```
person1 = Person("Hanna")
person2 = Person("Erik")

person1.endre_navn("Hanne") # Endrer objektet
person1 = person2           # Endrer variabelen
```



Merk: vi får en kopi av *referansen*, ikke *objektet*! person1 og person2 peker på *samme* objekt, ikke to kopier. Hvis vi endrer objektet person1 refererer til vil endringen også gjelde person2 (ettersom det er samme objekt, bare to variabler til det samme)



Menti

```
def endre(a, b):
    a = a + [4]
    b.append(4)


x = [1, 2, 3]
y = [1, 2, 3]
endre(x, y)
print(x, y)
```

Hva printes?

Oppgave 1

1. Hva refererer henholdsvis variablene *a*, *b* og *c* til i hovedprogrammet?
2. Hvorfor bruker vi ikke *self* noe sted i hovedprogrammet?

```
class A:
    def __init__(self, a):
        self._a = a

class B:
    def __init__(self, b):
        self._b = b

class C:
    def __init__(self, c):
        self._c = c

def hovedprogram():
    c = A("B")
    a = B("C")
    b = C("A")
    b = a
    a = c
    c = b

hovedprogram()
```

Menti

- **immutable objects** er objekter man ikke kan endre på (f.eks. String)
- **None** er en verdi som sier at variabelen ikke holder på noe objekt for øyeblikket
- **self** brukes for å peke på det nåværende objektet (objektet vi currently ser på). Vi trenger ikke å sende det inn som argument når vi kaller på metoder fordi det automatisk blir sendt inn når vi kaller metoden på gjeldende objekt

Objekter kan ha instansvariabler med referanse til andre objekter!

Oppgave 2

1. Hva er det *self* refererer til i init-metoden (konstruktøren) og i metodene hent_navn og endre_studieprogram?
2. Hva slags variabler er *navn*, *studieprogram* og *nytt_program*?
3. Hvordan skiller variablene i 2. seg fra variablene *_navn og _studieprogram*?
4. Hvorfor trenger vi *self* i metodene *hent_navn* og *endre_studieprogram*?
5. Hvorfor trenger vi ikke å inkludere *self* som argument når vi kaller metodene hent_navn og endre_studieprogram på et Student-objekt?

```python
class Student:
    def __init__(self, navn, studieprogram):
        self._navn = navn
        self._studieprogram = studieprogram

    def hent_navn(self):
        return self._navn

    def endre_studieprogram(self, nytt_program):
        self._studieprogram = nytt_program
```

```python
class Matbutikk:
    def __init__(self):
        self._varer = []
```

a) Utvid klassen `Matbutikk`, slik at den får en metode for å hente lista `varer`.

b) Skrive en metode for å legge til et `Vare`-objekt i lista `varer`.

c) Trenger vi flere parametere enn `self` i konstruktøren over? Begrunn svaret.

d) Skriv klassen `Vare`, med instansvariabler `vareid` og `varenavn`. init-metoden (konstruktøren) tar inn `vareid` og `varenavn`. `Vare` skal ha to metoder; én for å hente `vareid` og én for å hente `varenavn`.

e) Skriv et hovedprogram med en `Matbutikk`, og opprett og legg til tre `Vare`-objekter. Skriv ut `vareid` sammen med `varenavn` for de tre varene på tre forskjellige linjer. Dere blir nødt til å kalle relevante metoder underveis.

Oppgave 3

Menti

Datastruktur-tegning

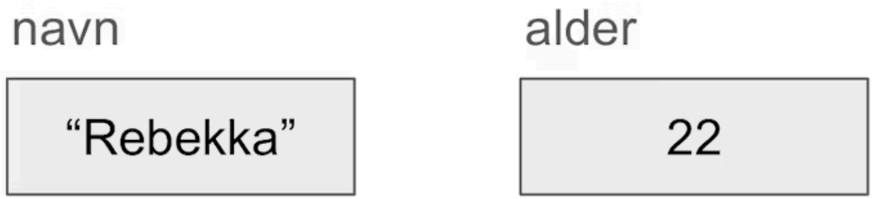
- **Variabler** - rektangel med verdi inni og variabelnavn over
- **Objekter** - rektangel med runde kanter, med instansvariabler inni og klassen (typen) over med ":" foran
- **Instansvariabler** - inni objektet, enten boks/rektangel eller <_instansvariabel> = <verdi>
- **Referanser** - pil fra variabel til objekt

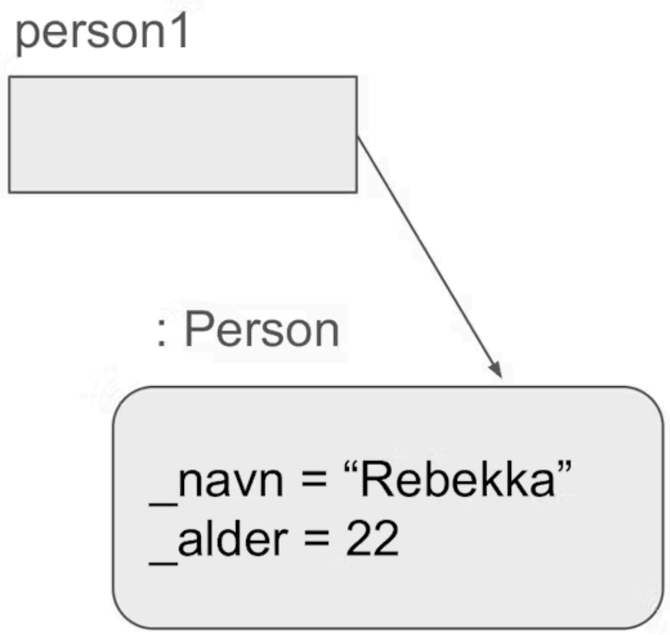
- **Variabler** - rektangel med verdi inni og variabelnavn over
- **Objekter** - rektangel med runde kanter, med instansvariabler inni og klassen (typen) over med ":" foran
- **Instansvariabler** - inni objektet, enten boks/rektangel eller <_instansvariabel> = <verdi>
- **Referanser** - pil fra variabel til objekt

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

```
navn = "Rebekka"
alder = 22

person1 = Person(navn, alder)
```





Oppgave 4

Gitt hovedprogrammet ved siden av, tegn datastrukturen slik den ser ut etter linjen forelder.få_barn(barn). En datastrukturtegning viser opprettede objekter og sammenhengen mellom dem. Tips: Hvordan datastrukturer tegnes i IN1000 ligger bakerst i ukas forelesningsslides, også fra i fjor. Sammenlign deres tegning med Python Tutors.

```
class Forelder:
    def __init__(self):
        self._barn = None

    def få_barn(self, barn):
        self._barn = barn

class Barn:
    def __init__(self, fødselsdato, fødselsvekt):
        self._fødselsdato = fødselsdato
        self._fødselsvekt = fødselsvekt

def hovedprogram():
    forelder = Forelder()
    barn = Barn("01012024", 3500)
    forelder.få_barn(barn)

hovedprogram()
```

Oppgave 5

1. Hva vil skrives ut i programmet? Kom opp med et svar og begrunn hvorfor før dere kjører programmet.
2. Endre programmet, slik at linjen dobbel_latte = KaffeLatte(2) står under linjen bestillinger = [] i stedet for inni for-løkka. Hva skrives ut nå? Kom opp med et svar og begrunn hvorfor før dere kjører programmet.
3. Hvor mange objekter av klassen KaffeLatte finnes i hovedprogrammet i a), og hvor mange finnes det i hovedprogrammet i b)? Begrunn hvorfor.

```
class KaffeLatte:
    def __init__(self, ant_espressoshots):
        self._antall_espressoshots = ant_espressoshots

    def hent_antall_espressoshots(self):
        return self._antall_espressoshots

def hovedprogram():
    bestillinger = []
    for i in range(5):
        dobbel_latte = KaffeLatte(2)
        bestillinger.append(dobbel_latte)

    for bestilling in bestillinger:
        print(bestilling)

hovedprogram()
```

Menti

Skriv ned en ting som ble klarere og en ting du fortsatt sliter med:) Og tilbakemeldinger på gruppetimene!