IN1000

UKE 8 - GRUPPE 1

MIAU@IFI.UIO.NO

Plan for dagen

- None
- Self
- Referanser
- Datastrukturtegning
- Oppgave

None

- None forteller oss at en variabel ikke holder på en verdi/et objekt akkurat nå
- I noen tilfeller kan det var nyttig å sette en variabel som None til å begynne med
- Det kan også være hjelpsomt å bruke None i f.eks. en if-sjekk

None

```
molly = Hund("Molly")
bella = None

liste = [molly, bella]

for hund in liste:
    if hund is not None:
        hund.bjeff()
```

Self

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

    def ta_ut(self, beløp):
        self._saldo -= beløp
        self._vis_saldo()

    def _vis_saldo(self):
        print(self._saldo)

bank = Bankkonto(1000)
bank.ta_ut(500)
```

Oppgave

Skriv en klasse ‘Sirkel’ i filen sirkel.py. En sirkel har en radius. Opprett en konstruktør som oppretter radius (sendt inn som parameter).

Lag så følgende metoder:
1. En som returnerer diameteren til sirkelen (radius * 2)
2. En som returnerer omkretsen til sirkelen (diameter * pi)
3. En som returnerer arealet til sirkelen (radius^2 * pi) for å få eksponent skriver man ** , f.eks. er 2^3 = 2 ** 3

Opprett så en test-fil der du lager to sirkler med ulik radius. Skriv ut den ene sirkelens omkrets og areal, og den andre sirkelens diameter.

Løsningsforslag

```python
class Sirkel:
    def __init__(self, r):
        self._radius = r

    def diameter(self):
        return self._radius * 2

    def omkrets(self):
        return self.diameter() * math.pi

    def areal(self):
        return (self._radius ** 2) * math.pi
```

Referanser

- Vi vet at en variabel er noe som holder på en verdi

```
string = 'hei'
tall = 1
float = 3.14
boolean = True
```

- Men når det er snakk om objekter og klasser, sier vi at en referanse peker på objektet

```
referanse = Sirkel(10)
```

Referanser

- Variabler er som små bokser som inneholder en verdi, mens referanser inneholder adressen til objektet.
- Variabler kan inneholde ting som strings, ints, floats og booleans (immutable)

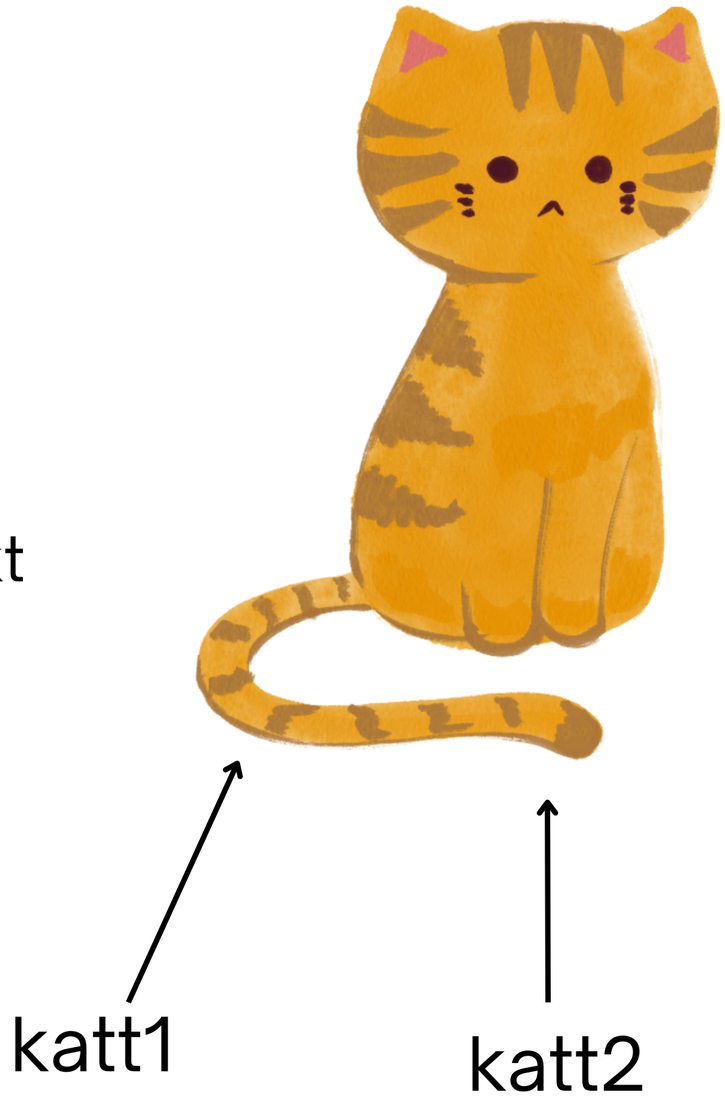
- Referanser peker på ting som lister, ordbøker, objekter/ klasser og mengder (mutable)

Referanser

- Vi bruker referanser for å få tak i objekter
- En referanse er noe (f.eks. et navn) som peker på/henviser til et objekt i minnet
  - Den *er* ikke objektet, men en *peker* til objektet

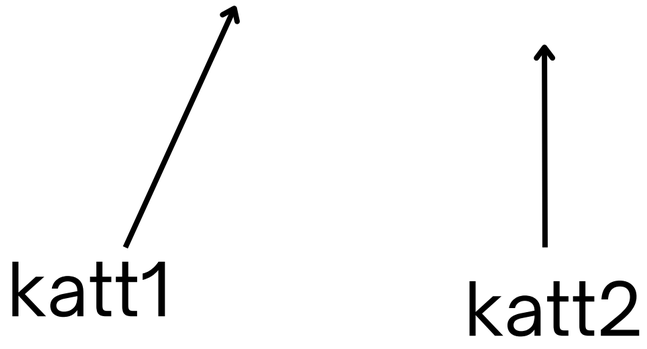
Referanser

- Vi bruker referanser for å få tak i objekter
- En referanse er noe (f.eks. et navn) som peker på/henviser til et objekt i minnet
  - Den *er* ikke objektet, men en *peker* til objektet
  - Flere referanser kan til referere til samme objekt
    - katt1 = Katt()
    - katt2 = katt1
  - Om man endrer på det referte objektet (vha. en metode) med et av navnene, endres det for alle
    - katt2.endre_farge(“svart”)



Referanser

- Vi bruker referanser for å få tak i objekter
- En referanse er noe (f.eks. et navn) som peker på/henviser til et objekt i minnet
  - Den *er* ikke objektet, men en *peker* til objektet
  - Flere referanser kan til referere til samme objekt
    - katt1 = Katt()
    - katt2 = katt1
  - Om man endrer på det referte objektet (vha. en metode) med et av navnene, endres det for alle
    - katt2.endre_farge(“svart”)



Eksempel med datastruktur

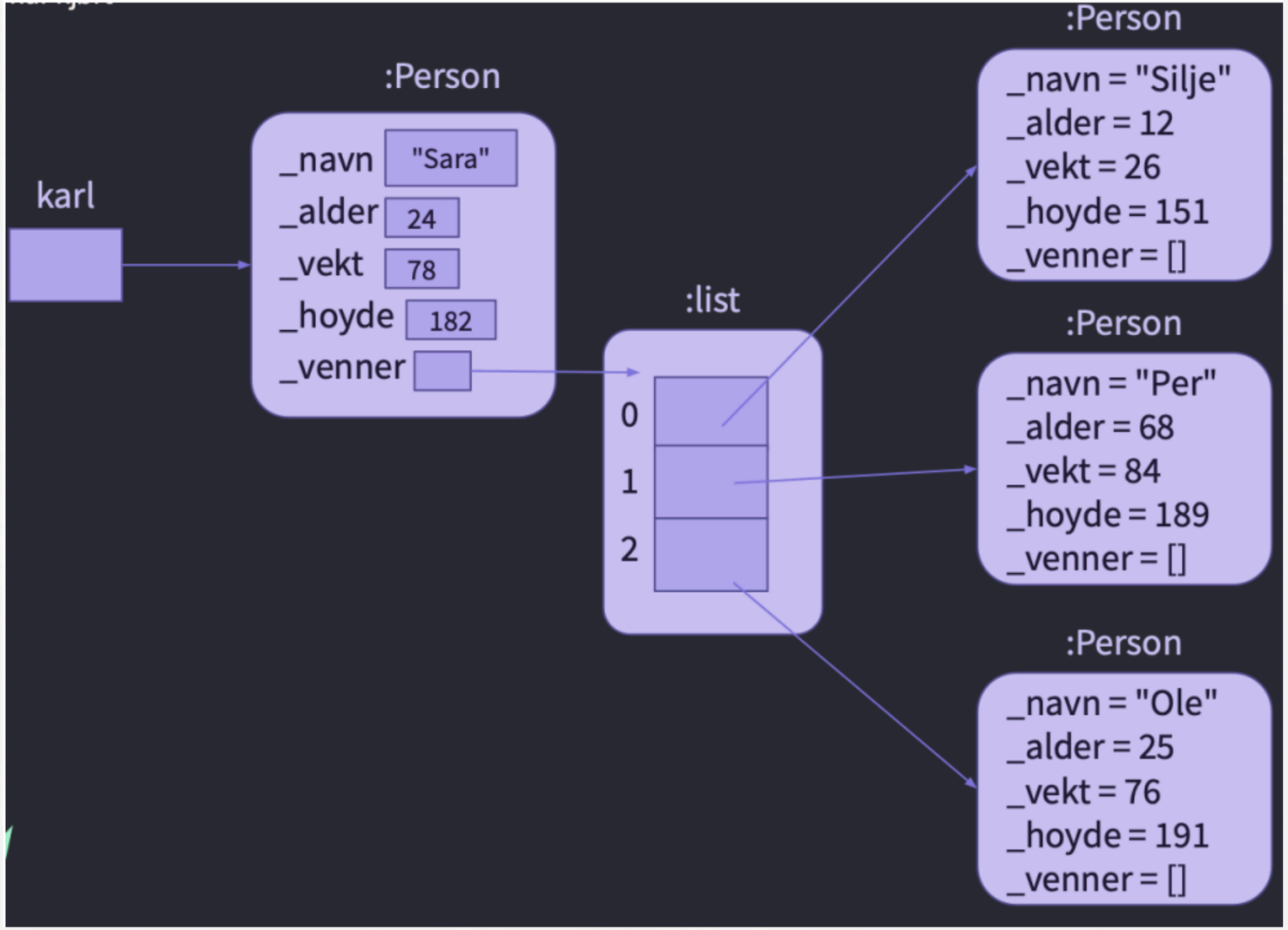
```
class Person:
    def __init__(self, navn, alder, vekt, høyde, venner):
        self._navn = navn
        self._alder = alder
        self._vekt = vekt
        self._høyde = høyde
        self._venner = venner



silje = Person("Silje", 12, 26, 151, [])
per = Person("Per", 68, 84, 189, [])
ole = Person("Ole", 25, 76, 191, [])

karl = Person("Sara", 24, 78, 182, [silje, per, ole])
```

Eksempel med datastruktur





Oppgave

Lag en klasse blomst. En blomst har en art, en høyde, en instansvariabel som sier om blomster har det bra eller ikke (boolean), og en verdi som forteller hvor lenge siden blomsten ble vannet sist. Konstruktøren setter alle verdiene. Når en blomst blir opprettet vil det være 0 dager siden den ble vannet sist.

En blomst har en metode **hent_status** som returnerer om blomsten har det bra eller ikke.

Metoden **skriv_ut_info** skriver ut all info om blomsten.

Metoden **neste_dag** øker antallet dager siden blomsten ble vannet med 1. Hvis det er mer enn 3 dager siden den ble vannet, vil status til blomsten være at den har det dårlig. Om status på blomsten er at den har det bra, vokser den 1 cm.

Den siste metoden er **vann.** Hvis det er mindre enn 3 dager siden blomsten ble vannet vil statusen til blomsten bli dårlig (overvannet), ellers vil den være bra.

Skriv klassen og et testprogram.

Løsningsforslag

```python
class Blomst:
    def __init__(self, art, høyde):
        self._art = art
        self._høyde = høyde
        self._har_det_bra = True
        self._vannet_sist = 0

    def hent_status(self):
        return self._har_det_bra

    def skriv_ut_info(self):
        print(f'Art: {self._art} \nHøyde: {self._høyde}')

    def neste_dag(self):
        self._vannet_sist += 1
        if self._vannet_sist > 3:
            self._har_det_bra = False

        if self.hent_status():
            self._høyde += 1

    def vann(self):
        if self._vannet_sist < 3:
            self._har_det_bra = False
        else:
            self._har_det_bra = True

        self._vannet_sist = 0
```