Mentimeter

Uke 7 - konsept

Gruppe 14

Kristin - kristlgj@ifi.uio.no

Mentimeter



Mentimeter

Plan for gruppetimen

- Mentispørsmål om fagord fra forelesning
- Refleksjon av forelesning
- Recap av pensum med oppgaver
  - klasser og objekter
    - instansvariabler
    - konstruktør
    - metoder
    - laste inn klasse

Mentimeter

Læringsmål denne uken

- Kjenne til motivasjon og bakgrunn for objektorientert programmering
- **Programmering: Kunne definere en klasse, opprette objekter og kalle metoder på objektene**
- Kunne bruke og forklare begrepene klasse, objekt, klassedefinisjon, konstruktør, metode og instansvariabel
- Forstå sentrale begreper som grensesnitt og innkapsling
- Kjenne til utviklingsprosessen for en klasse gjennom design, implementasjon og testing

Mentimeter

Hva beskriver best objektorientering?

0 ✓

Organisering av et program i klasser og objekter

0 ×

Å organisere all kode i funksjoner, der hver funksjon har sitt ansvarsområde

0 ×

Å unngå variabler

0 ×

Å unngå det globale skopet ved å skrive all kode i et hovedprogram

Mentimeter

Hva er en klasse?

0 ✓	0 ✗	0 ✗	0 ✗
En oppskrift for objekter	Det samme som et objekt	En samling av variabler	En tekststreng

Mentimeter

Hva er et objekt?

0 ✓ En konkret instans av en klasse

0 ✗ Det samme som en klasse

0 ✗ En oppskrift for å modellere virkeligheten

0 ✗ En pakke med moduler

Mentimeter

Hva brukes konstruktøren __init__ til?

0 ✓ Å initialisere et nytt objekt

0 × Å avslutte programmet

0 × Å skrive ut dokumentasjon

0 × Å importere en modul

Hva er en instansvariabel?

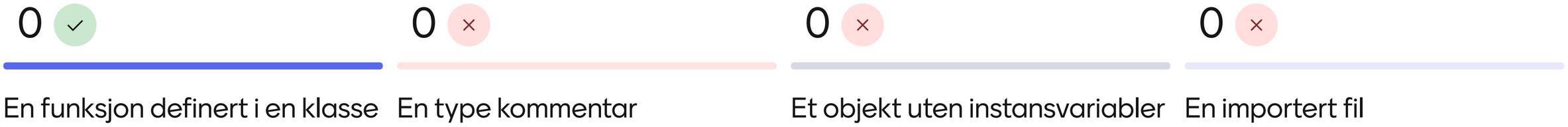
0 ✓
En variabel som tilhører et bestemt objekt

0 ×
En variabel som aldri endrer verdi

0 ×
En variabel som tilhører en bestemt klasse

0 ×
En variabel som bare brukes i løkker

Hva er en metode?



Mentimeter

Hva er et grensesnitt?

0 ✓
Måten annen kode aksesserer et objekt på

0 ×
En oversikt over alle metoder og instansvariabler et objekt har

0 ×
En kopi av et objekt

0 ×
Et annet ord for terminal der man kjører programmer

Hva betyr innkapsling?

0

Konstruksjon av et objekt der instansvariabler er utilgjengelige utenfor klassen

0

Å dele all data til det globale skopet

0

At du kan bruke metoder som du ikke vet hvordan er skrevet

0

Å skrive kode uten klasser

Mentimeter

Hva innebærer modellering i programmering?

0 ✓
Å lage en forenklet modell av et system

0 ×
Å skrive all kode som objekter

0 ×
Å bytte programmeringsspråk til Modula

0 ×
Å måle hvor raskt et program kjører for å finne forbedringer

Mentimeter

Hva ble gjennomgått på forelesningen? Tenk selv i 2 min, så diskuter

- Hva lærte du, hva skjønte du ikke?
- Se gjerne over slides fra forelesningen om dere ikke husker!

Tenk gjerne igjennom:

- *Hva er forskjellen på et objekt og en klasse, egentlig? Kan dere finne en metafor fra virkeligheten som beskriver dette?*
- *Hittil har vi brukt variabler, samlinger, løkker, og funksjoner for å løse oppgaver. Hvorfor tror dere man fant opp objektorientering? Hvilket problem løser dette?*

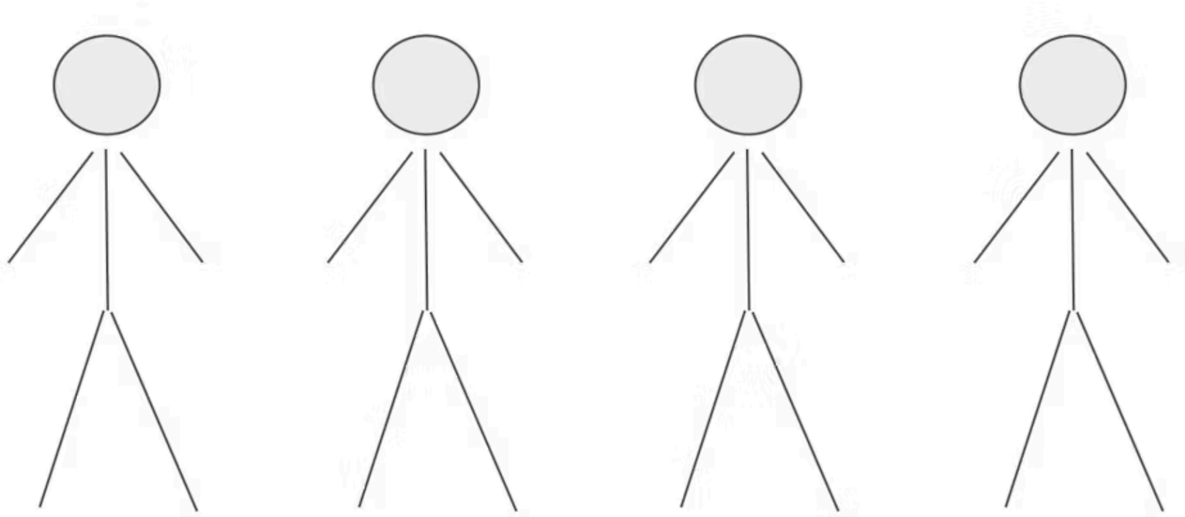
Mentimeter

Objekter og klasser - hva er det?

Objektorientert programmering handler om å “modellere” verden.

- Objekter representerer ting - har egenskaper (*instansvariabler*) og oppførsel (*metoder*)
- En klasse beskriver et sett med objekter med lik oppførsel (mal for objektet)

Klasse er en mal for noe
Objekter er *instanser* av klasser



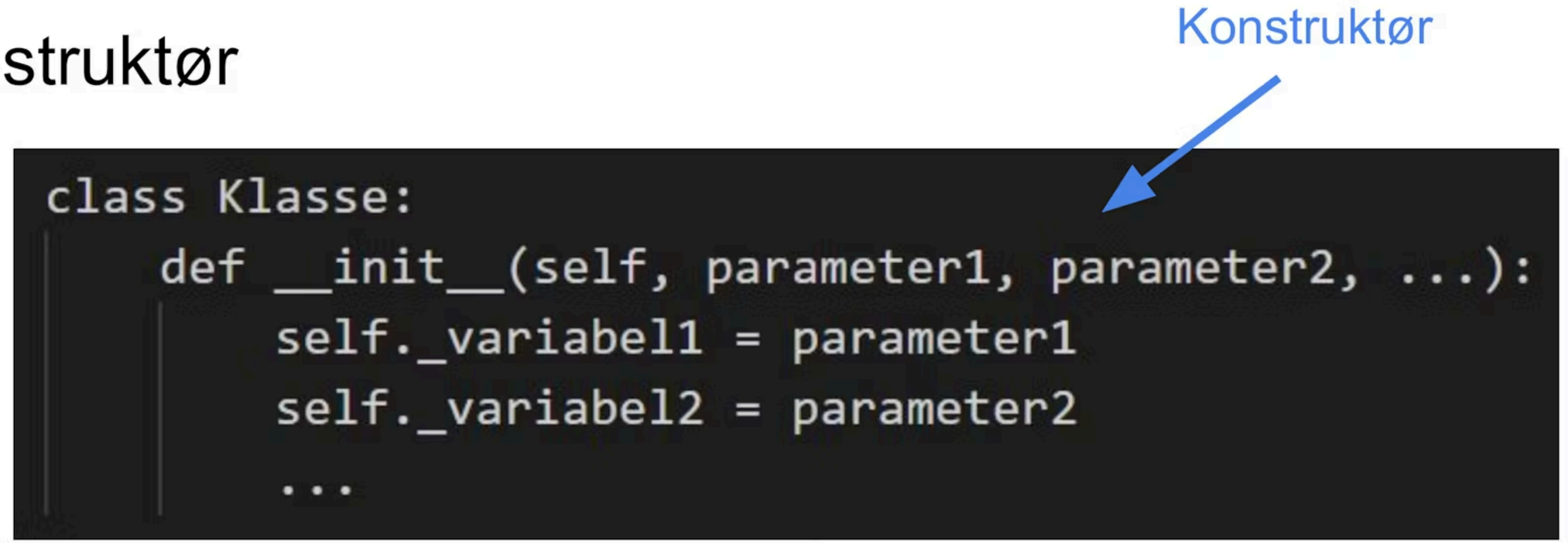
Mentimeter

Instansvariabler

- instansvariabler er egenskapene til en klasse
- variabler som hører til klasse
- de vil få egne verdier for hver *instans* av klassen (hvert objekt)

Mentimeter

Konstruktør



Konstruktør - bestemmer hva som skal være med ved opprettelsen av et objekt. Parametre til konstruktøren angir hvilke *instansvariabler* vi kan velge verdi for når vi oppretter et nytt objekt.

Mentimeter

Eksempel:

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

Når vi oppretter en *instans* av klassen (et objekt) sender vi med argumentene slik:

```
person1 = Person("Isabell", ...)
```

Mentimeter

Oppgave 1

Med utgangspunkt i tegningene lagd av personene på tavla lag en klasse Person, med de gitte instansvariablene: navn, alder, høyde og favorittfarge.

Mentimeter

Metoder

- I tillegg til egenskaper (*instansvariabler*) vil vi at klassen også skal tilby tjenester (via *metoder*)
- Metoder er som funksjoner/prosedyrer, bare at de tilhører en klasse (og har derfor tilgang til alle instansvariablene til klassen)

```
class Klasse:
    ...

    def metode_navn(self, parameter1, ...):
        # Kode inni metoden
```

Mentimeter

Metoder

- I tillegg til egenskaper (*instansvariabler*) vil vi at klassen også skal tilby tjenester (via *metoder*)
- Metoder er som funksjoner/prosedyrer, bare at de tilhører en klasse (og har derfor tilgang til alle instansvariablene til klassen)
- Husk self som parameter!

```
class Klasse:
    ...

    def metode_navn(self, parameter1, ...):
        # Kode inni metoden
```

Mentimeter

Eksempel:

```python
class Person:
    def __init__(self, navn, favorittfarge):
        self._navn = navn
        self._favorittfarge = favorittfarge

    def ny_favorittfarge(self, ny_farge):
        self._favorittfarge = ny_farge
```

```python
person1 = Person("Isabell", "grønn")
person1.ny_favorittfarge("blå")
```

Mentimeter

Metoder, grensesnitt og innkapsling

- *public interface (offentlig grensesnitt)* - alle metodene man kan bruke
  - grensesnittet til en klasse er alt som kan gjøres utenfor skopet til klassen - altså alt man har tilgang til “utenifra”
- *innkapsling* - vet hva en metode gjør, trenger ikke vite hvordan
  - kan altså bruke metoder uten å vite implementasjonen
  - instansvariabler og metoder som ikke skal være i grensesnittet får navn som begynner med ‘_’ (for å vise at de ikke skal aksesseres utenifra)

Dere har allerede brukt metoder!

- list.append(), string.lower() etc.

Mentimeter

Oppgave 2

Vi skal nå utvide Person-klassen vår 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 jeg er <alder> år gammel”.
3. feire_bursdag(self) →
    a. Øk alderen til personen med 1.
    b. Skriv ut en bursdagshilsen med den nye alderen.
    c. Hvis personen fremdeles er under 18 år, skal også høyden økes med 2.

Mentimeter

Laste inn klasse

- i IN1000 vil vi at alle klasser skal skrives i egen fil
- vi vil altså ha en fil til hver av klassene våre og en fil til hovedprogrammet

- det gjør at vi må laste inn klassen til hovedfilen (hovedprogrammet)
- lettest hvis man har alle filene i samme mappe!

from <filnavn> import <Klasse>

```
from person import Person
```

<- hvis vi har Person-klassen i person.py

Mentimeter

Oppgave 3

Gitt klassen Person som vi har laget tidligere. I en egen fil:

1. Opprett 3 instanser (objekter) av klassen med navn/alder/høyde/favorittfarge du velger selv.
2. Skriv deretter ut en hilsen fra hver person deres ved hjelp av skriv_ut_hilsen-metoden.
3. Feire 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.

Mentimeter

Oppgave 4

Skriv en klasse **Dyr**. Et dyr har følgende egenskaper:

- Art
- Kjønn
- Vekt

Lag en konstruktør (init-metode).

Lag en metode som skriver ut informasjonen om dyret.

Opprett tre objekter av klassen Dyr i en annen fil, for å representere tre forskjellige dyr. Skriv ut informasjon om alle dyrene.

Mentimeter

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