

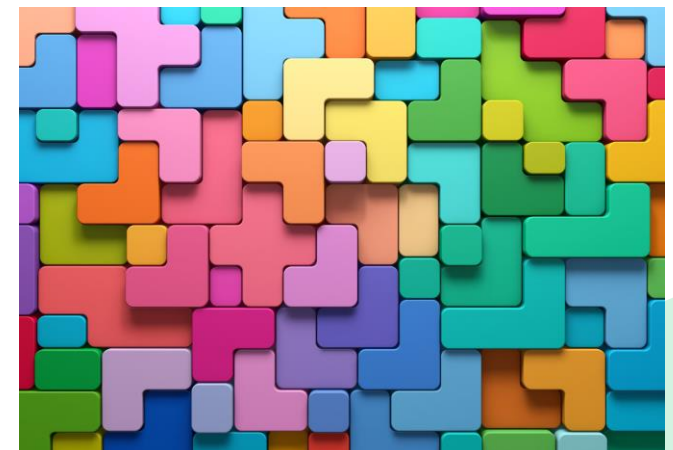
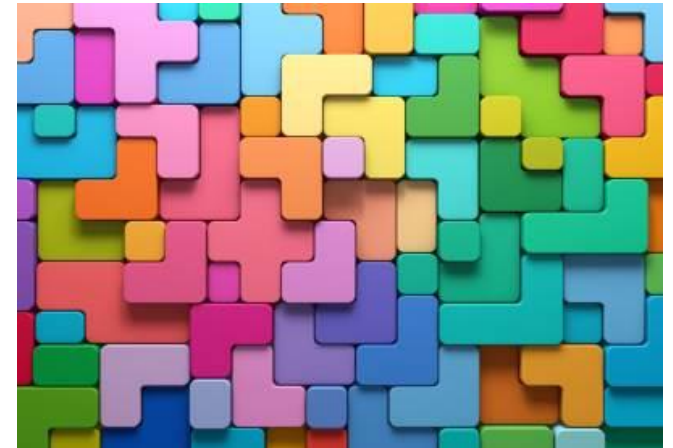
OBJEKTORIENTERT DESIGN

OG

**FUNKSJONER SOM
OBJEKTER I PYTHON**

FORELESNING 8

FREDAG 13/9



(bilder generert av bing image creator)

○ Læremål: Avansert bruk av Python

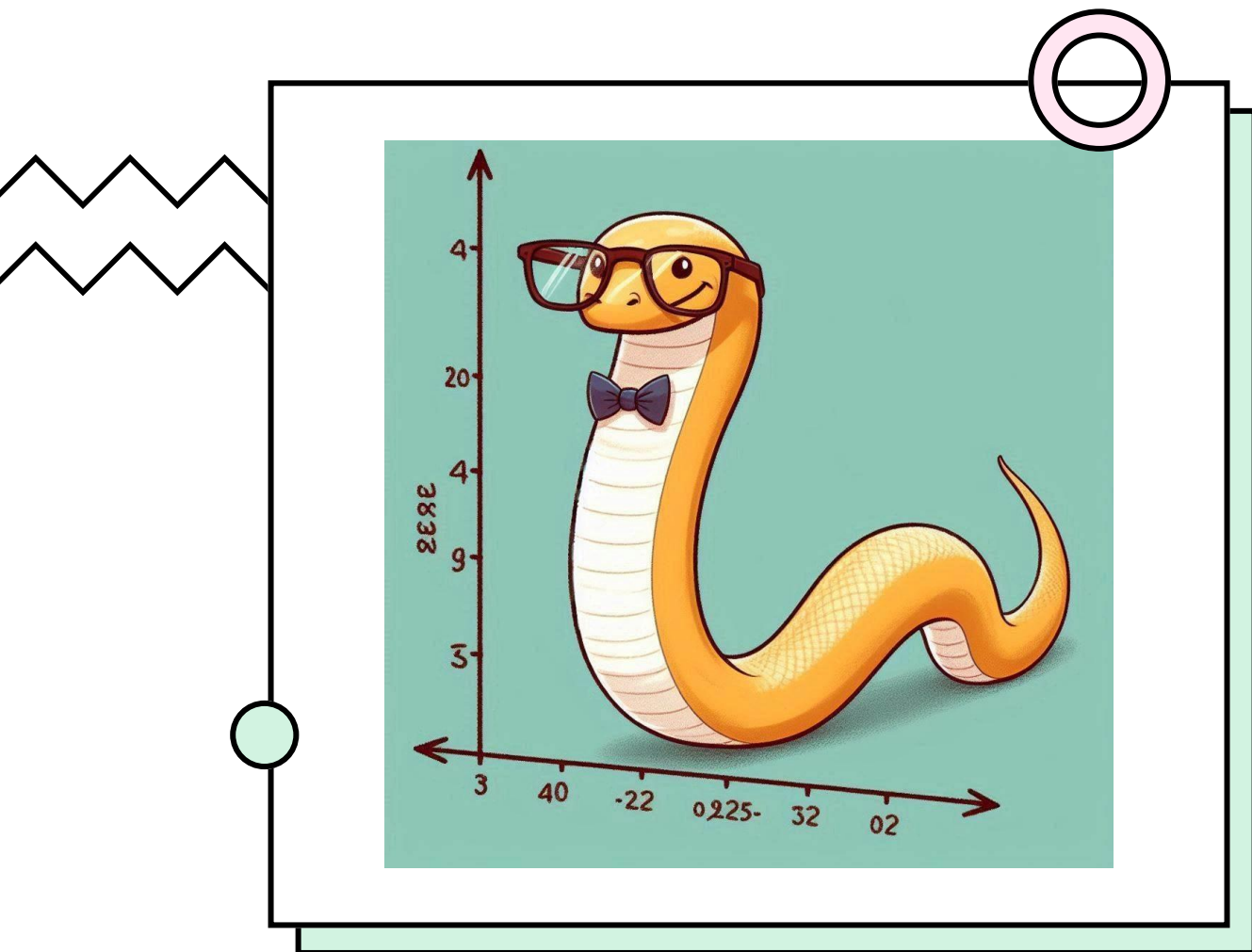
- Gjøre programmene lettere å lese og finne fram i
- *Objektorientert programmering* er en måte å tenke på når vi skriver programmer som åpner nye muligheter
- Lar oss lage spesialtilfeller fra mer generelle tilfeller (arv)
- (+ mer de kommende ukene)



○ Støttelitteratur

- Ikke pensum (pensum = forelesninger + prosjekter)
- Det som står om IN1910 er fra tidligere semestre og gjelder ikke (nødvendigvis) for i år!
- Men for de som vil ha et annet perspektiv på det vi gjennomgår – eller bare liker å lese tekst – så kan det være nyttig med litt [støttelitteratur](#)





LIVEKODING:
DATAKLASSER
OG MER OM
FUNKSJONER



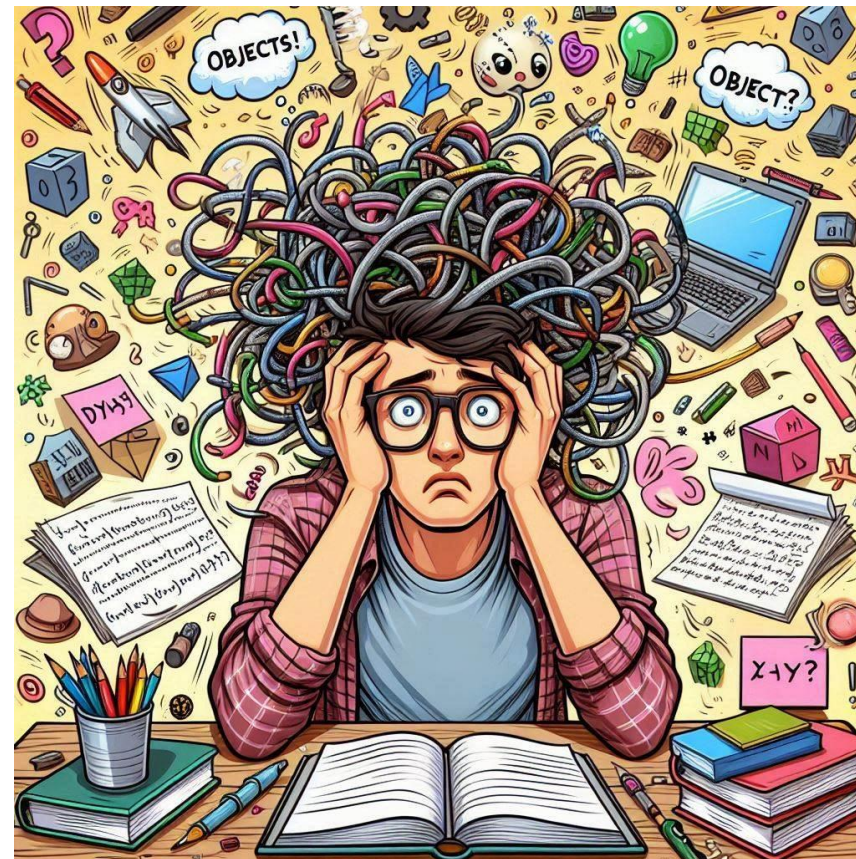
○ Dataobjekter – en måte å jobbe på:

- Prøv først med å arve fra NamedTuple
 - Hvis det ikke funker (f.eks. hvis noe må endres)...
- Prøv en dataclass
 - Hvis det heller ikke funker
(f.eks. hvis `__init__` må gjøre mer enn å bare sette attributter)...
- Bruk en vanlig klasse
 - (fordel: ingen import nødvendig)
- Dette er ikke en regel, bare et forslag!



Nesten *alt* i Python er objekter!

- Det tomme objektet **None** er et objekt (av klassen **NoneType**)
- Funksjoner er objekter (av klassen **function** eller **builtin_function_or_method**)
- Objekters metoder er objekter (av klassen **method** eller **builtin_function_or_method**)
- Selv *klasser* er objekter (av klassen **type**)



○ Etter forelesningen

- [Vurderingskriteriene til prosjekt 1](#) er nå publisert
- Sjekk også oppgavetekst for eventuelle endringer
- Får du ikke kontakt i løpet av et par dager, eller opplever andre hindringer i prosjektarbeidet, bruk [dette skjemaet](#)

