

IN1000
Forelesning 10

Objektorientert design

Oblig 8 = Prøveeksamen

- Som oblig
  - Ignorér “Skriv svar her:”
  - Lever oppgavene som filer (.txt og .py)
  - Tips: Lag lovlige (skrevne, trykte) hjelpemidler til eksamen underveis!
  - Du får ikke brukt elektroniske hjepemidler på eksamen
- Som prøveeksamen i Inspera (i mai)
  - En mer realistisk versjon (mer slik det blir i eksamenslokalet)
  - Her kan du prøve å løse oppgavene på tid (4-5 timer)
  - OBS: Oppgavene kan gå noe raskere siden du tross alt har løst dem en gang før
  - Du får ikke kjørt / testet koden din i Inspera! (Vi tar hensyn til dette når vi retter eksamen)

Litt ekstra motivasjon: Spilldesign

Who This Is For

This documentation is meant to be an introduction to game development and MonoGame. Readers should have a foundational understanding of C# and be comfortable with concepts such as classes and objects.

https://learn.microsoft.com/en-us/dotnet/csharp/tour-of-csharp/tips-for-python-developers

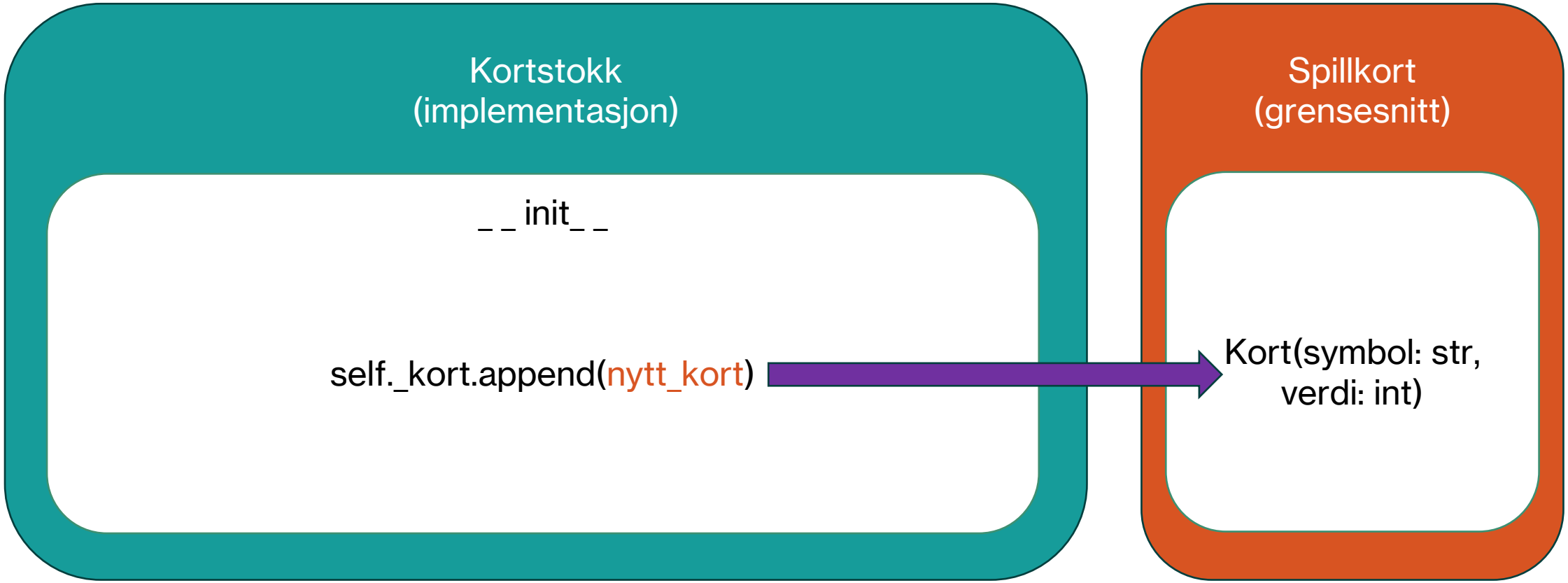
To måter å jobbe med Python på

- “Pythonsk”
  - Omfavner at objekters typer kan være hva som helst – spesielt for akkurat Python
  - Hvis det funker, så funker det (prøve og feile)
  - Hvis det mot formodning ikke funker, så fang opp og håndtér alle mulige feilmeldinger (“bedre å få tilgivelse enn tillatelse”)
- Vår måte
  - Bygger et grunnlag for å gå videre til mer klassiske programmeringsspråk
  - Vi er mer opptatt av hva typer til objekter er (for å skape en bevissthet rundt det)
  - Sjekk heller først at ting har riktig type slik at vi kan stole på at vi vet hva de er (“bedre å få tillatelse enn tilgivelse”)
- Det beste er å beherske begge deler (men den første er lurt å være klar over)

Hvor langt er vi kommet?

- Oblig 6
  - Fokus på å kunne *bruke* objekter riktig
  - Grensesnitt
- Oblig 7
  - Fokus på *hvordan* objekter gjør det de gjør
  - Implementasjon
  - Sammenkobling av *flere* hjemmelagde objekter (klasser)
- Oblig 8 ← **vi er her**
  - Prøveeksamen (uten tidspress)
  - Bli kjent med eksamensformatet og spesielt den store oppgave 4
  - Fylle inn “skjelettet” fra de siste forelesningene med mer kjøtt på beinet

Se en klasse innenfra og en annen utenfra – samtidig



Livekoding: Forsøk 1



Vi trenger først en fil fra en tidligere forelesning!

Vi kan bruke ordbøker og lister til alle data

- Men da må vi kjenne innholdet og strukturen godt
- Vi kan også bruke funksjoner til å gjøre ting med disse
- Men da må vi også vite hvordan disse funksjonene skal brukes
- Hvordan ser dette ut hvis vi prøver å definere objekter som kan gjøre noe selv?
- Et **Land**-objekt som er en samling med fylker
- **Fylke**-objekter som er samlinger med kommuner
- **Kommune**-objekter som inneholder data om kommunene

Livekoding: Forsøk 2



Hvor mye kan vi kapsle inn i klassen?

- Med offentlige instansvariabler må brukeren av klassene fortsatt vite mye om hvordan dataene er strukturert (med nøkler og verdier)
- Det er fortsatt mye som må gjøres manuelt – objektene gjør ikke så mye selv
- Samtidig gir dette fleksibilitet – vi er ikke låst til en “riktig” måte å bruke objektene på
- Hva om vi prøver å innkapsle så mye som mulig og la objektene gjøre mest mulig på egen hånd?
- Objekter som overlater mye til brukere (på godt og vondt)
  → Objekter som gjør mest mulig selv

Livekoding: Forsøk 3



menti.com

4349 1742

Kan også brukes om du ser opptak (siste spørsmål)