Intro og tips til oblig 8

Fagutvalgets IN1000-seminar

3. november 2021

Siri Moe Jensen

I dag…

- Hva handler oppgaven om ("domenet")
- Hva er aggregering?
- Noen tips og fallgruver

De åpenbare

- Les oppgaven
  - When in doubt, les den om igjen
  - if still in doubt, spør på Mattermost eller i gruppetime
- Skriv din egen løsning
- Test, test og test (små biter og helhet)
- Vær sikker på at du forstår hva som skjer i hvert skritt
  - teori og/ eller test: pythontutor eller print-setninger (fjernes)

Domene:
Tungregning eller High Performance Computing (HPC)

HPC systems

Each of the HPC facilities consists of a **compute resource** (a number of **compute nodes** each with a number of **processors** and internal shared-memory, plus an interconnect that connects the nodes), <...>

<...>.UiOs lokale tungregneanlegg Fox – der forskere får tilgang til UiOs kraftigste regneklynge. Fox har om lag 3000 kjerner fordelt på 24 computenoder, hver med 128 kjerner og 512 gigabyte minne og en lynrask NVMe interndisk på flere terabytes. <....>

Hva med "racks"?

Domene:
Tungregning eller High Performance Computing (HPC)

Når du har bruk for MYE datakraft!!

- Data science, prosessering av store mengder forskningsdata (bioinformatikk, astronomi)
- Google, Facebook, Amazon...
- Militære formål, helsestatistikk, meterologi, ...

- UiO og Uninett (og resten av verden) tilbyr stadig kraftigere og mer avanserte tjenester for dette
- Enorm vekst i etterspørsel, og svært rask utvikling i arkitekturer og teknologi

=> Forvirrende landskap og terminologi!

- Heldigvis beskriver obligteksten en veldig forenklet modell som kan implementeres ganske direkte

Racks under montering av regneklyngen Saga - arvtageren til Abel som tilbyr tungregnings-tjenester.

Regneklyngen Saga

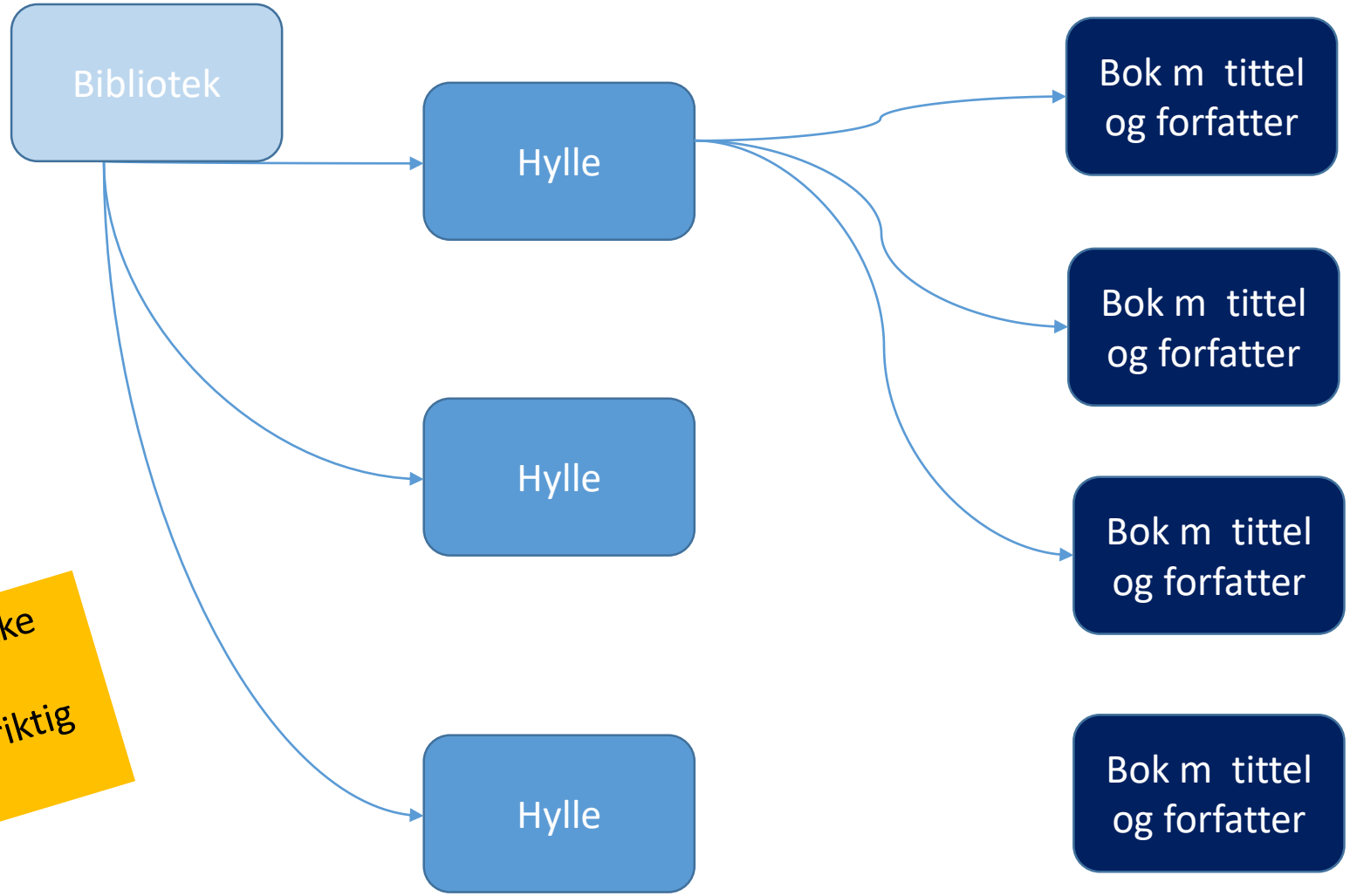
Sentrale begreper i oppgaven

- **Datasenter**: "Noen" som tilbyr tungregnings-tjenester, typisk ved hjelp av en eller flere regneklynger
- **Regneklynge**: Et system sammensatt av mange datamaskiner (**noder**) som kan samarbeide tett om en oppgave og ~fremstå som én maskin
- **Rack** (skap): Nodene som skal samarbeide plasseres i ett eller flere rack ("møblering")
- **Node**: En av maskinene i en regneklynge
- **Prosessor**: En node kan ha en eller flere (2) prosessorer som utfører instruksjoner – og et minne (hukommelse) med en begrenset størrelse

Vi kan f eks *splitte* sammensatt innhold og legge det i ulike klasser

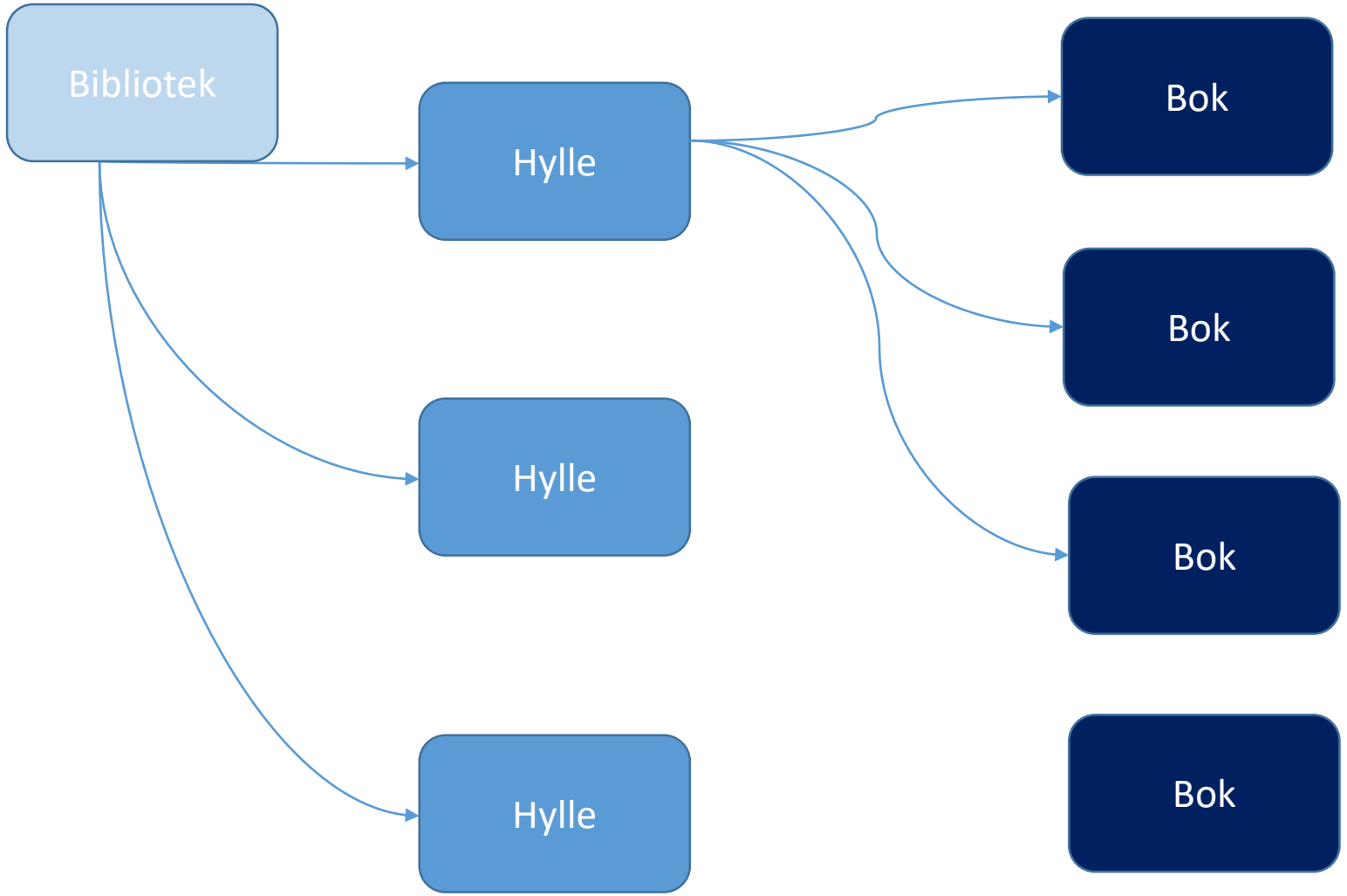
Bibliotek med hyller der hver hylle inneholder en eller flere bøker og hver bok har en tittel og en forfatter

Vi kan f eks *splitte* sammensatt innhold og legge det i ulike klasser

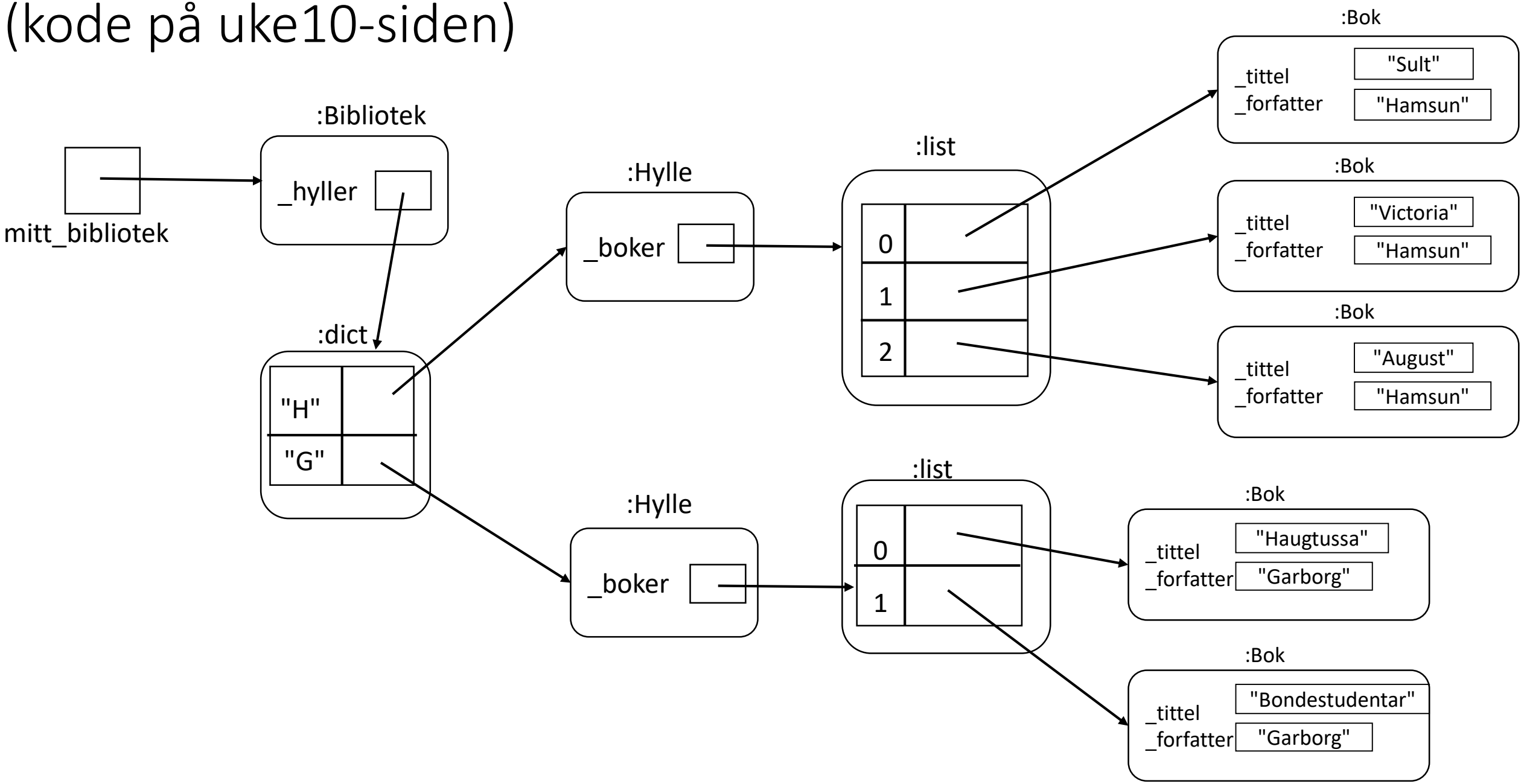


Forenklet tegning - ikke tegn slik i oblig!
Se senere slide for riktig eksempel

Kalles gjerne *aggregering* ("oppsamling") når man betrakter hele strukturen

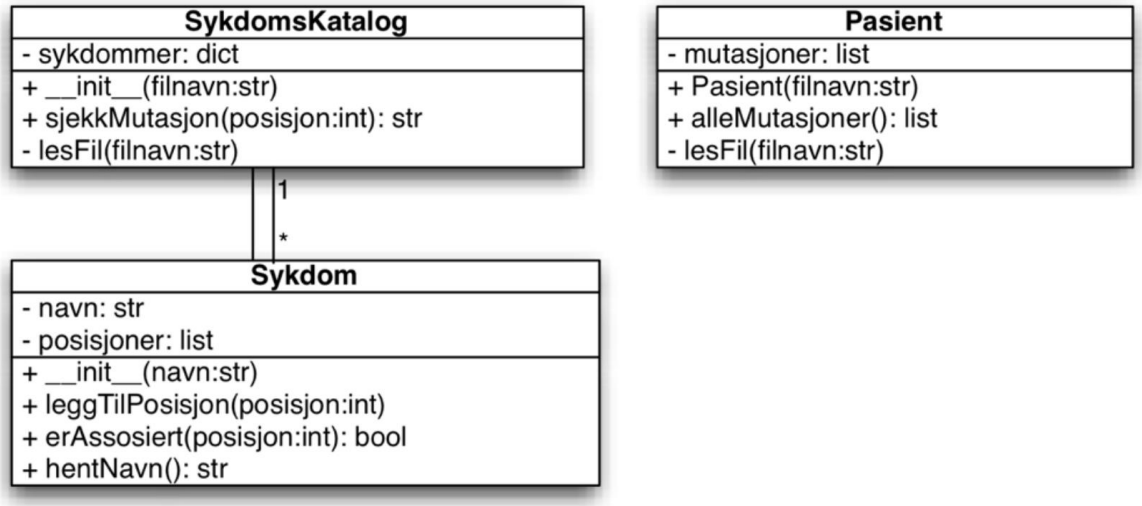


Datastruktur-tegning i IN1000 oblig-format (kode på uke10-siden)



Kan være lurt å tegne klassene i UML

- NB: UML-diagrammene Geir Kjetil viste i dag er klassediagrammer.
- Viser potensialet i klassedefinisjonene – ikke hvordan strukturen ser ut under kjøring (det er ingen objekter i disse tegningene)



Hva trengs for å løse oblig 8?

1. Python konstrukter som gjennomgått på forelesning
2. Ha jobbet med eksempler* på datastrukturer der flere objekter av ulike klasser inngår og samarbeider
    1. Spillelistene fra oblig 7
    2. T-bane og kollektivnettet fra uke 10
    3. Biblioteket fra uke 10 (kode på uke10-siden)
    4. DNA-eksempelet fra uke 11
3. Sette seg inn i modellen som beskrevet i oppgaven

4. Løse oppgaven skritt for skritt ☺ kanskje med nye runder på 1-3

*NB: Det er lite her som kan kopieres uten tilpasning til datastruktur og hva programmet faktisk skal gjøre!

Datasenter-oppgaven

- Klassestrukturen er gitt
- Funksjonaliteten er gitt
- Format på input-filer er gitt
- For de som ønsker det: Grensesnittet til hver klasse er oppgitt i Python
- Fremgangsmåte:
  - Enklest å starte med Node-klassen og bygge videre på den?
  - Husk å bruke eksisterende metoder i de ulike klassene

Typiske spørsmål og problemer

- Navn på klynge => input-filene er oppgitt å ha navn etter regneklyngens navn etterfulgt av ".txt"
  - Generelt, spesielt til eksamen: er du i tvil eller mener det mangler informasjon i oppgaven: Gjør en antakelse, skriv en kommentar om dette i koden, og programmer ut fra den

- Utskrift-eksempel: Merk "eksempel", oppgaveteksten beskriver hva du må ha med – du velger format (og finner riktige verdier)

- Blanding av tab og space i koden lager krøll for Python
  => editor kan settes opp til å endre alle tabs til space (eller omvendt), eller substituere eksplisitt (f eks m/ replace)

Lese fra fil

- Formatet
  - Hver linje *etter den første* oppgir antall noder med et gitt antall prosessorer og minnestr
  - Du må selv opprette nye rack etter behov, og fylle disse
  - Filene inneholder ikke informasjon om hvor mange racks som finnes i datasenteret, bare hvilke noder som er tilgjengelige => vi plasserer disse i racks (som vi "bygger" ved behov)

- Lese fil der første linje avviker fra de andre, en av flere løsninger:
  - fil.readline() lager en liste der hvert element er en linje i filen
  - kan prosessere første linje separat
  - deretter gå gjennom de resterende linjene i listen (hver linje gir info om noder med samme antall prosessorer og minnestørrelse)

- Tips: Anta at racks fylles løpende, og at alle untatt det siste er fulle