

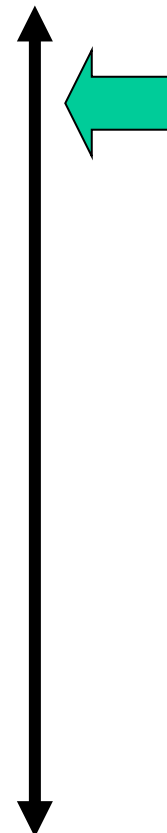
OO Datamodellering

Et dataprogram er jo en “ting” i seg selv, men vil jo vanligvis være en modell av noe annet.

Vi ønsker å lage et program som skal gjøre noe spesifikt. For å implementere dette OO, modellerer vi det ved hjelp av

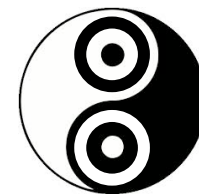
- Objekter
 - Objektene kan være av forskjellig typer
 - Representere konkrete eller abstrakte ting
- Relasjoner/koblinger mellom objektene
 - Objektene kan være løst eller tett koblet sammen på forskjellige måter
 - Koblingene kan også være objekter I seg selv, f.eks. partnerskap, funksjoner mer om dette senere.

data



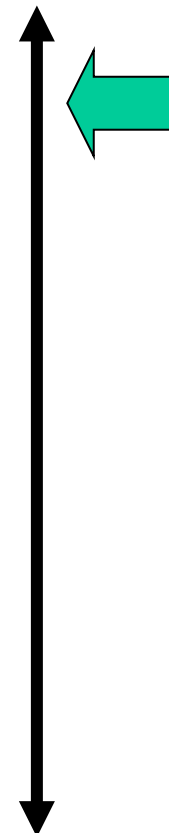
funksjoner

OO Datamodellering



data

- Objekt-orientering er ment å hjelpe oss å lage et system ved at det (i alle fall til en viss grad), svarer til «ting».
- Objektene i systemet kan representere konkrete ting eller abstrakte ting: Person, bok, film, emne, studieprogram,
- Disse er knyttet sammen på forskjellige måter, ofte kalt relasjoner eller assosiasjoner.
- Det er gjerne regler for disse relasjonene
- En vil gjerne ha oversikt over alt dette før en koder, derfor lager en *datamodeller* og tegner *diagrammer* funksjoner





Eksempel: Person

- En *person* har **navn**, **e-post** osv.

klasse

Person

String navn
String e-post

instanser (= objekter)

#1: Person

navn = "Hallvard"
e-post = "hal@idi.ntnu.no"

#2: Person

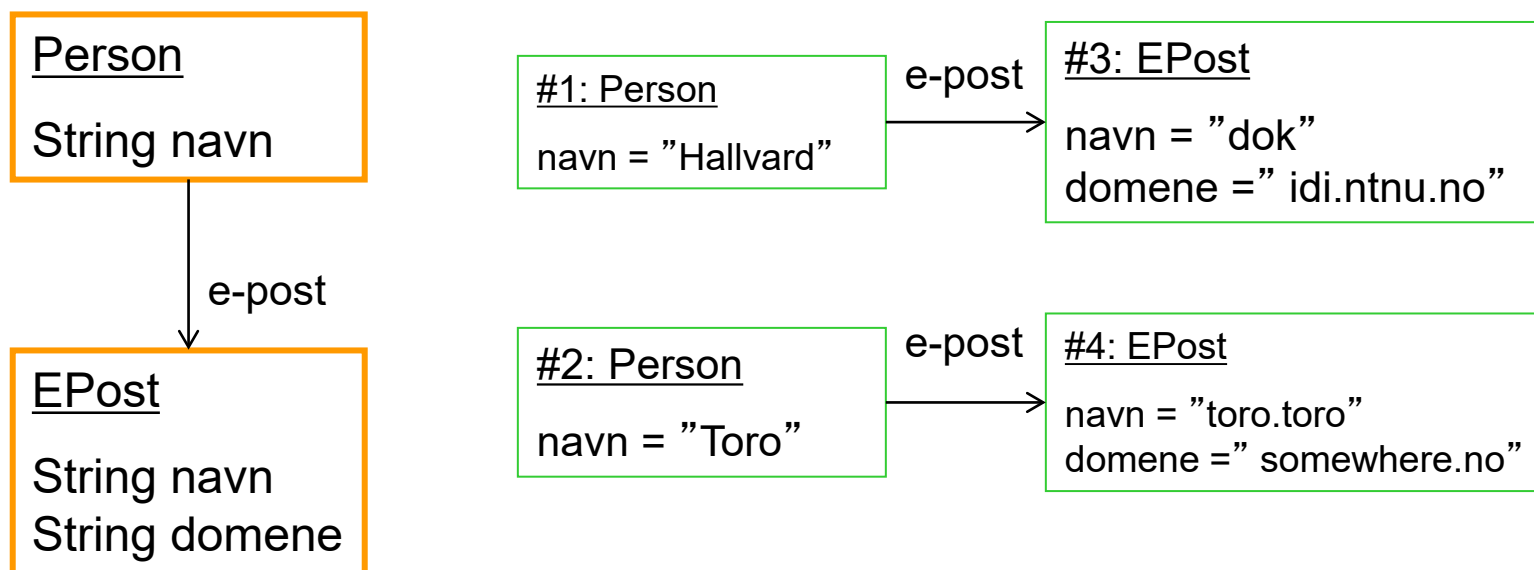
navn = "Marit"
e-post = "marit.reitan@svt.ntnu.no"

- Klassen beskriver det som er felles egenskaper i all instansene



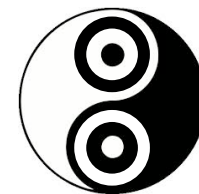
Eksempel: Person

- En *person* har **navn**, **e-post** osv.
- Er e-post "verdig" egen klasse?



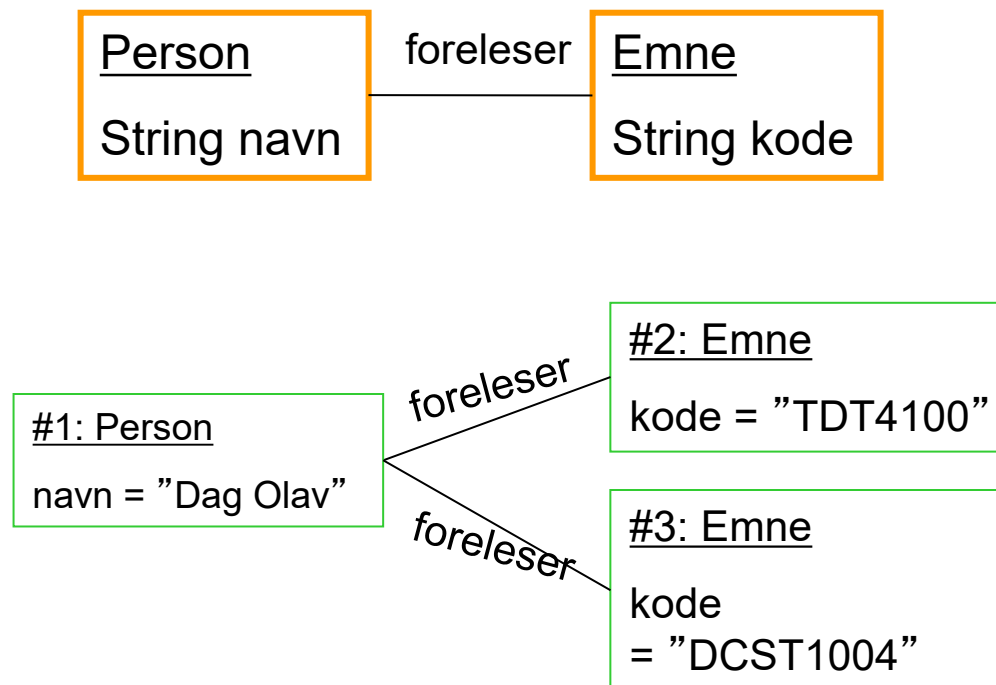
Når skal vi ha en egen klasse for å representere noe i modellen?

- Trenger epost være en egen type objekt?
 - Ok, et String er jo også en klasse, men her tenker vi på det som en grunnleggende type.
- Hvor mye skal til for at noe skal fortjene sin egen klasse?
- En må se på hva en trenger, og også hva slags objekter en allerede har tilgjengelig.

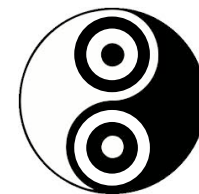


Eksempel: Person++

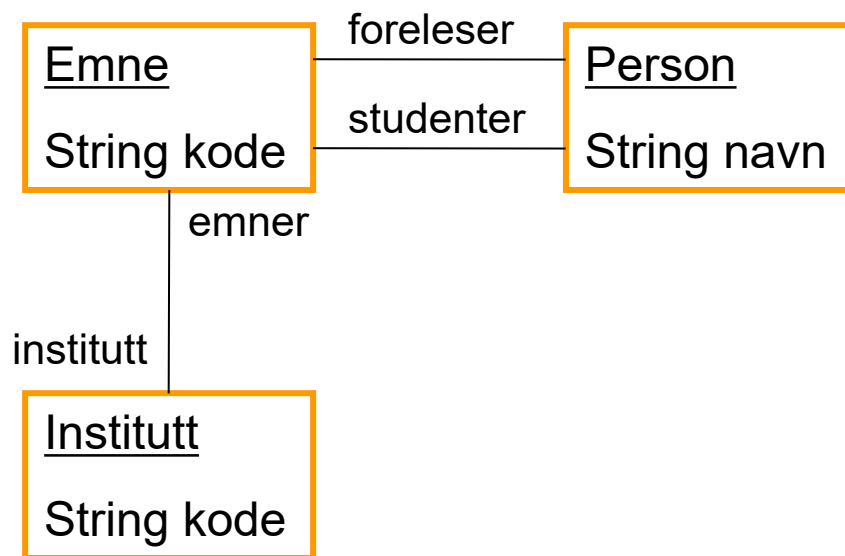
- ...
- En person kan være *foreleser* i *emner*



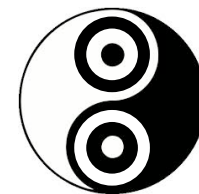
Eksempel: Person++



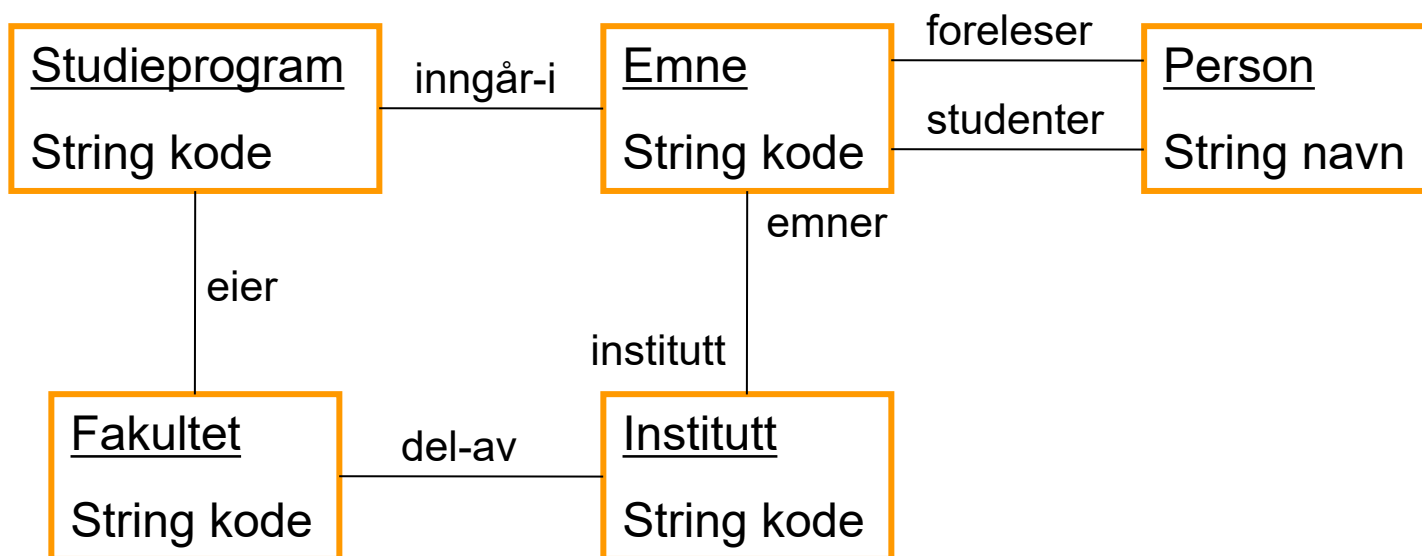
- ...
- En person kan være *foreleser* i *emner*
- Emner tas av *studenter* og gis av *institutt*



Eksempel: Person++



- ...
- En person kan være *foreleser* i *emner*
- Emner tas av *studenter* og gis av *institutt*
- Emner inngår i studieprogram, som eies av et fakultet...



Sosiale medier

- Hva slags data har vi her?
- Hvordan skal vi beskrive det, så vi kan lage koden for å representere dataene?

Not affiliated with Facebook

2K 111 Comments 319 Shares 61K Views

Like Comment Share

Petter Paus ▸ **Sommersetra Randonnéeklubb**
February 5 at 1:27pm · 🌐

Siste opptelling viste 9 påmeldte til Romsdalen. Vi har nå bestilt to rom med tilsammen 11 soveplasser. Det er med andre ord ennå plass til to friskusere som flytter ru...



Steinar Lien

31 mutual friends including Marit Reitan and Per Arne Hamre

Lives in Trondheim, Norway

Friends Following Message

Steinar Lien Jeg kjører - som tidligere nevnt - gjerne avgårde i retning Romsdalen tidlig en fredags morgen. Pga av intern familiær turkollisjon, er jeg blitt satt opp med 2-hjulsdrevet takstativløs bil denne helgen. Men jeg kan ta med masse skiutstyr inne i bilen og to friskusere evt. en friskus og en sykklus evt. to sykkluser i tillegg til chaufføren.

Like · Reply · 2d

Hallvard Trætteberg Jeg vil gjerne være friskus!

Like · Reply · 1m

What are your thoughts?

Facebook-begreper

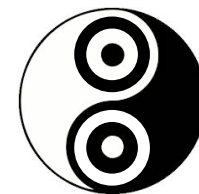
- Person
- Friend
- Group
- Wall
- Page
- Public
- Friend of friend
- Post
- Share
- Comment
- Message
- Like

Spørsmål om assosiasjoner

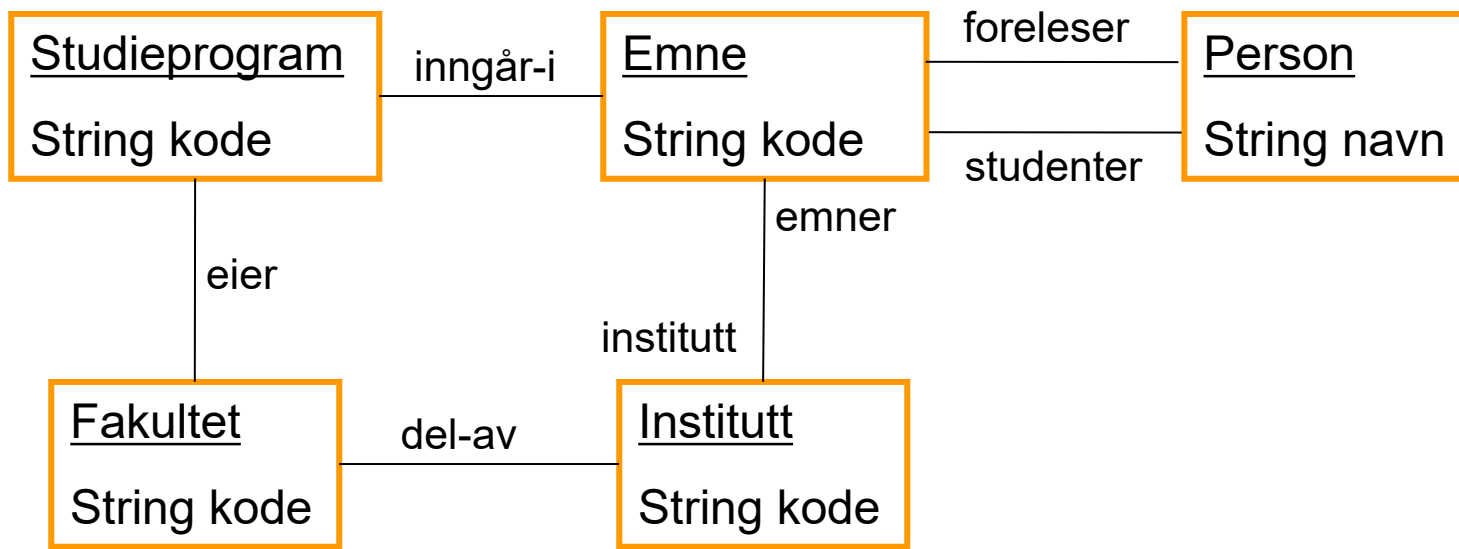


- Hvor mange assosiasjoner tillates?
 - **multiplisitet:**
én til én (eller ingen) eller
én til mange (bestemt antall eller flertall)?
- Er assosiasjoner to-veis?
 - **navigerbarhet:** skal begge ender vite om den andre?
 - **roller:** bruker en egne navn på hver retning?
- Impliserer assosiasjoner **innholdt-i**-logikk
 - et objekt kan bare være direkte **innholdt-i** ett annet objekt
 - når et objekt slettes, så slettes objekter som er **innholdt-i** det også

Spørsmål om assosiasjoner



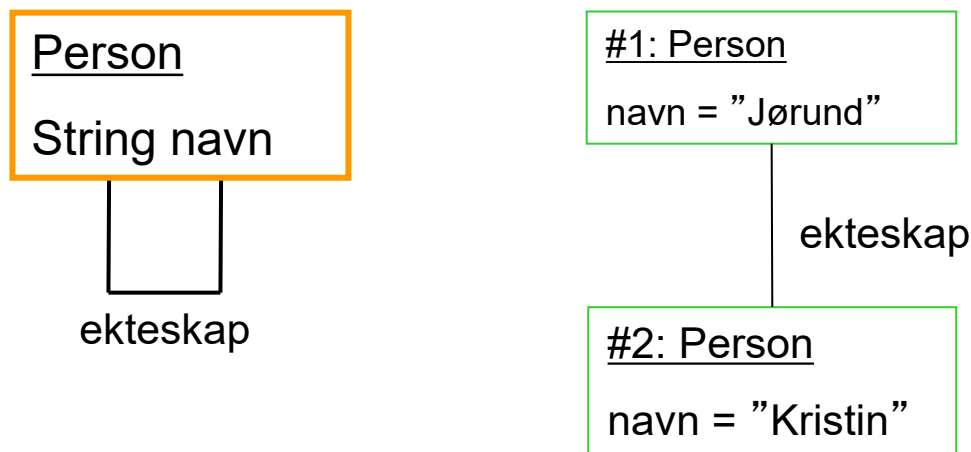
- **multiplisitet:** antall koblinger
- **navigerbarhet og roller:** retning og navn på kobling
- **aggregering/komposisjon:** eierskap





Assosiasjon innen klasse

- En *person* har **navn**, **e-post** osv.
- En person kan være *knyttet* til en annen person gjennom *ekteskap/partnerskap*



- En spesifikk person kan ikke være sin egen ektefelle/partner!

Ekstra spørsmål (beskrankninger/constraints)

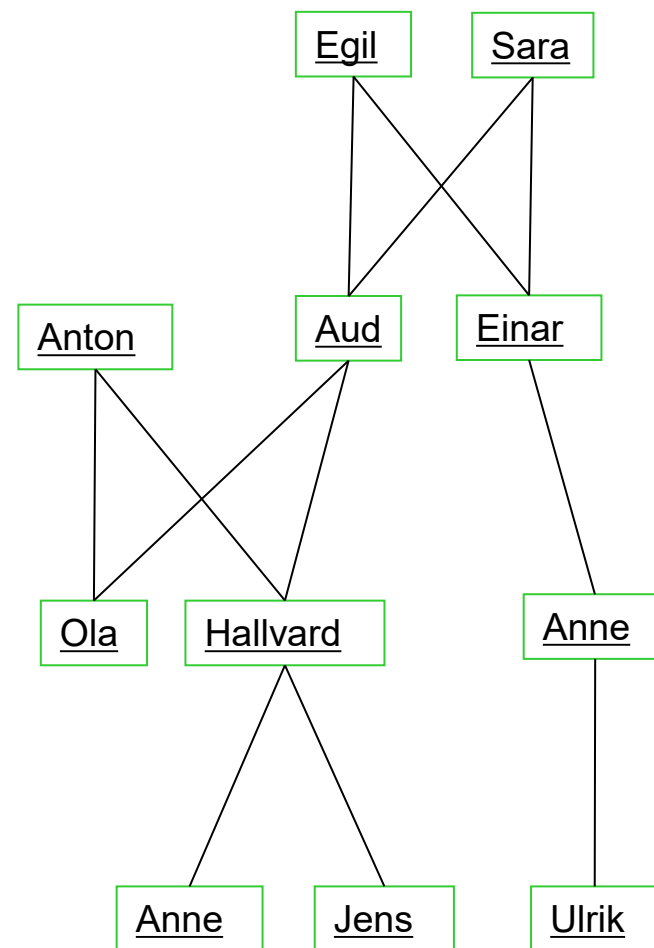


- Refleksivitet: $A \rightarrow A \ (\forall x \in X : x R x)$
 - “ser lik ut”, “er lik” for alle naturlige tall
 - anti-refleksiv: kobling til seg selv er ulovlig
 - A er ikke gift med seg selv.
- Symmetri: $A \rightarrow B \Rightarrow B \rightarrow A$
 - A gift med B
 - anti-symmetrisk: kobling tilbake er ulovlig
 - Armen er en del av meg, men jeg er ikke en del av armen
- Transitivitet: $A \rightarrow B \ \& \ B \rightarrow C \Rightarrow A \rightarrow C$
 - Hånda er en del av armen, armen er en del av meg... hånda del av meg

Inverse og avledede assosiasjoner

- Familiebegreper:

- **søsken**: barn av forelder
- **besteforelder**: forelder til forelder
- **tante/onkel**: søster/bror til forelder evt. deres ektemake
- **niese/nevø**: sønn/datter av søsken
- **kusine/fetter**: datter/sønn av onkel eller tante
- **filleonkel/tante**: kusine/fetter til forelder
- **tremenning**: barn av fille-onkel eller -tante evt. barnebarn av oldeforeldre (tre nivå opp til forforelder)



Person/familie-relasjoner

	Refleksiv	Symmetrisk	Transitiv
Søsken			
Halvsøsken			
Partnerskap			
Etterkommer			
Slektskap			
Venn			

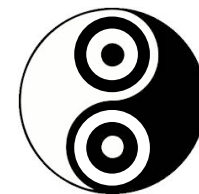
Notasjon for assosiasjoner



- **multiplisitet:** antall koblinger
- **navigerbarhet og roller:** retning og navn på kobling



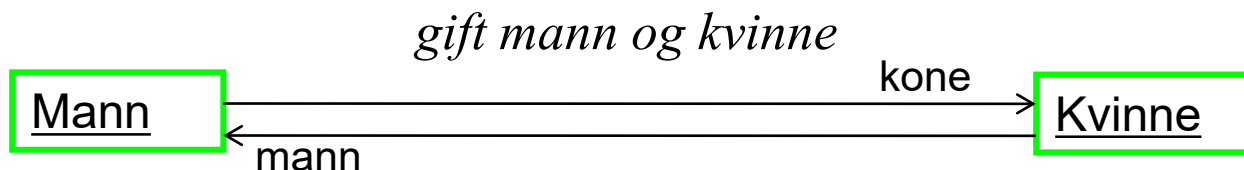
- En instans av Klasse1 har minst **min2** og maks **max2** **rolle2**-koblinger til instanser av Klasse2.
- En instans av Klasse2 har minst **min1** og maks **max1** **rolle1**-koblinger til instanser av Klasse1.
- *Notasjon:*
 - Når max er ubegrenset, så brukes **n** eller *
 - assosiasjonsnavnet utelates ofte



Eksempel: 1-1

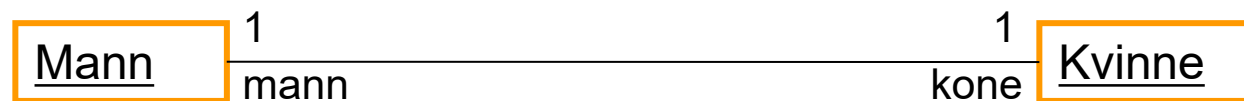


- En instans av Mann har minst **0** og maks **1** **kone**-kobling til instanser av Kvinne og *motsatt*

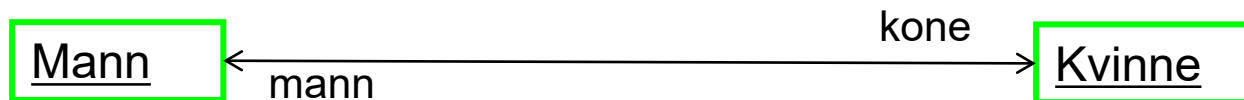


- Forenklet notasjon

*når **min** er 0, kan den utelates*



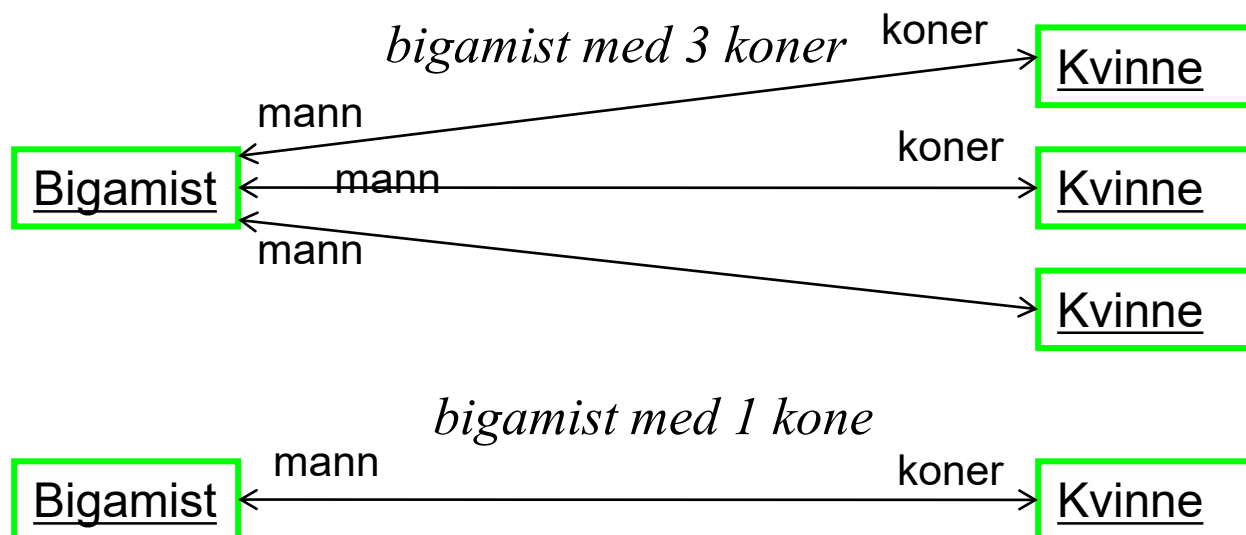
når koblingen går begge veier, så trengs bare én strek



Eksempel: 1-n



- En instans av Bigamist har (minst **0** og) ubegrenset antall **koner**-koblinger til instanser av Kvinne.
- En instans av Kvinne har (minst **0** og) og maks 1 **mann**-kobling til instanser av Bigamist.

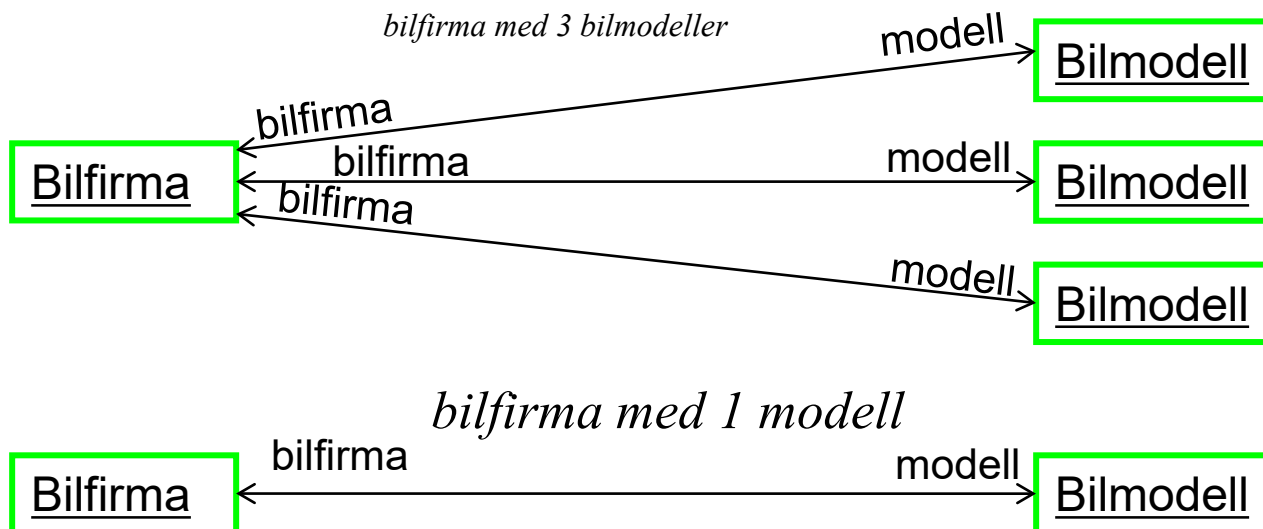


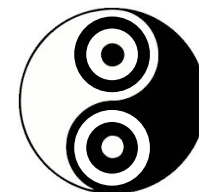


Eksempel: 1-n

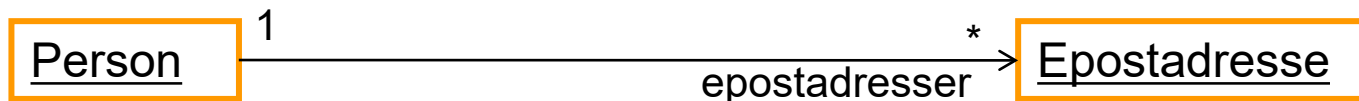


- En instans av Bilfirma har (minst **0** og) ubegrenset antall **bilmodeller**-koblinger til instanser av Bilmodell.
- En instans av Bilmodell har (minst **0** og) og maks 1 **bilfirma**-kobling til instanser av Bilfirma.

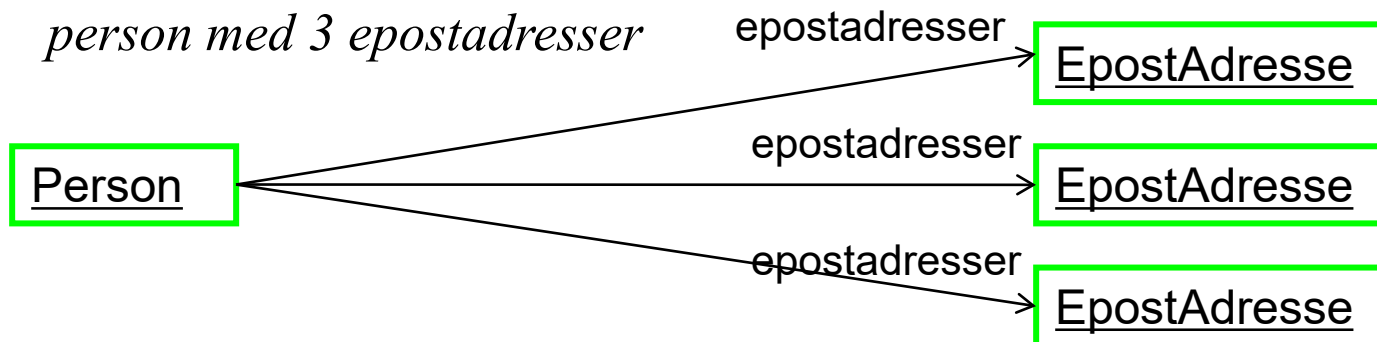
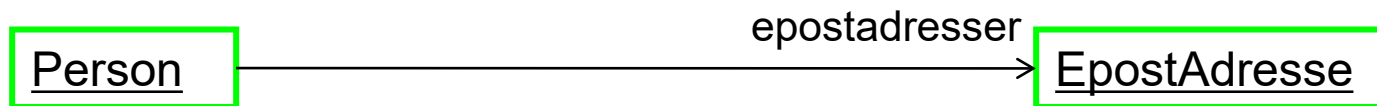




Eksempel: enveis 1-n

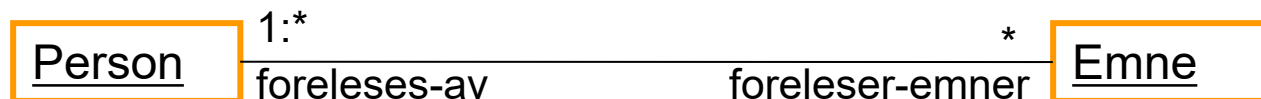


- En instans av Person har ubegrenset antall **ePostAdresser**-koblinger til instanser av Epostadresse.
- Spesialnotasjon: enveis-assosiasjoner tegnes med pil
person med 1 epostadresse

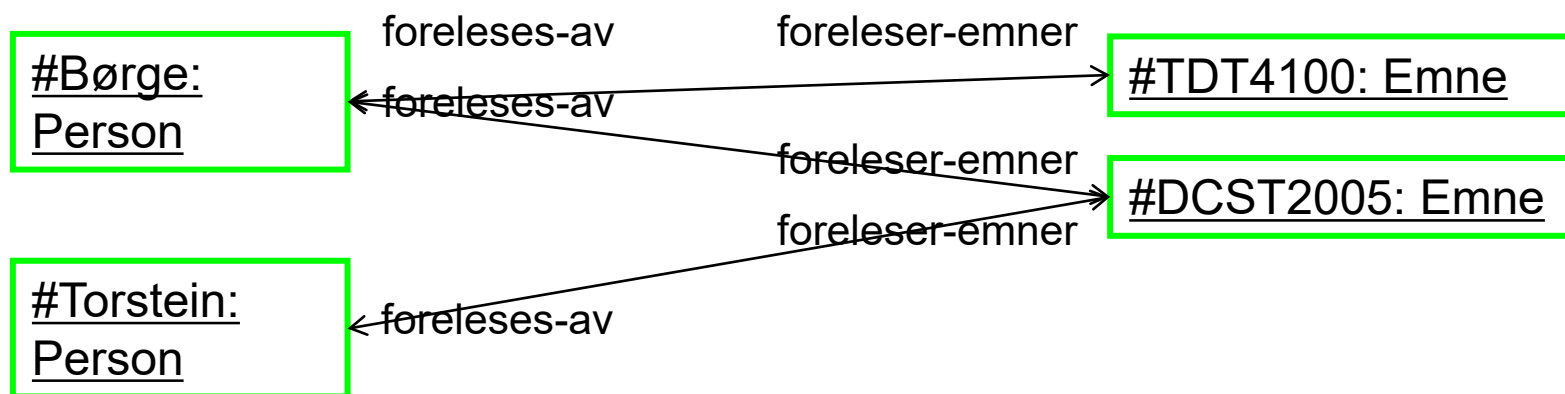




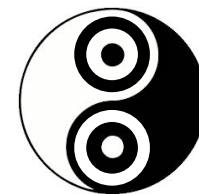
Eksempel: n-n



- En instans av Person har ubegrenset antall **foreleser-emner**-koblinger til instanser av Emne.
- En instans av Emne har minst 1 og ubegrenset antall **forelesere**-koblinger til instanser av Person.

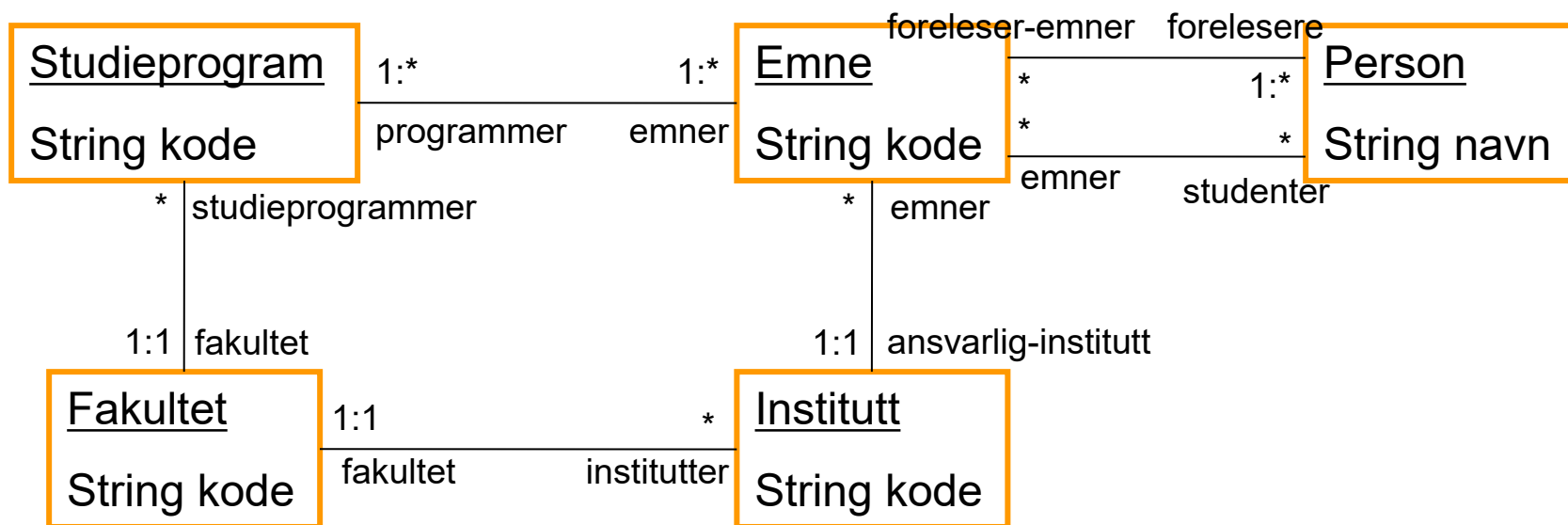


Spørsmål om assosiasjoner

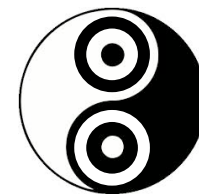


- **multiplisitet:** antall koblinger
- **navigerbarhet og roller:** retning og navn på kobling
- **aggregering/komposisjon:** eierskap

– Aggregering: samling av noe – “har en”-relasjon



Assosiasjoner og koding

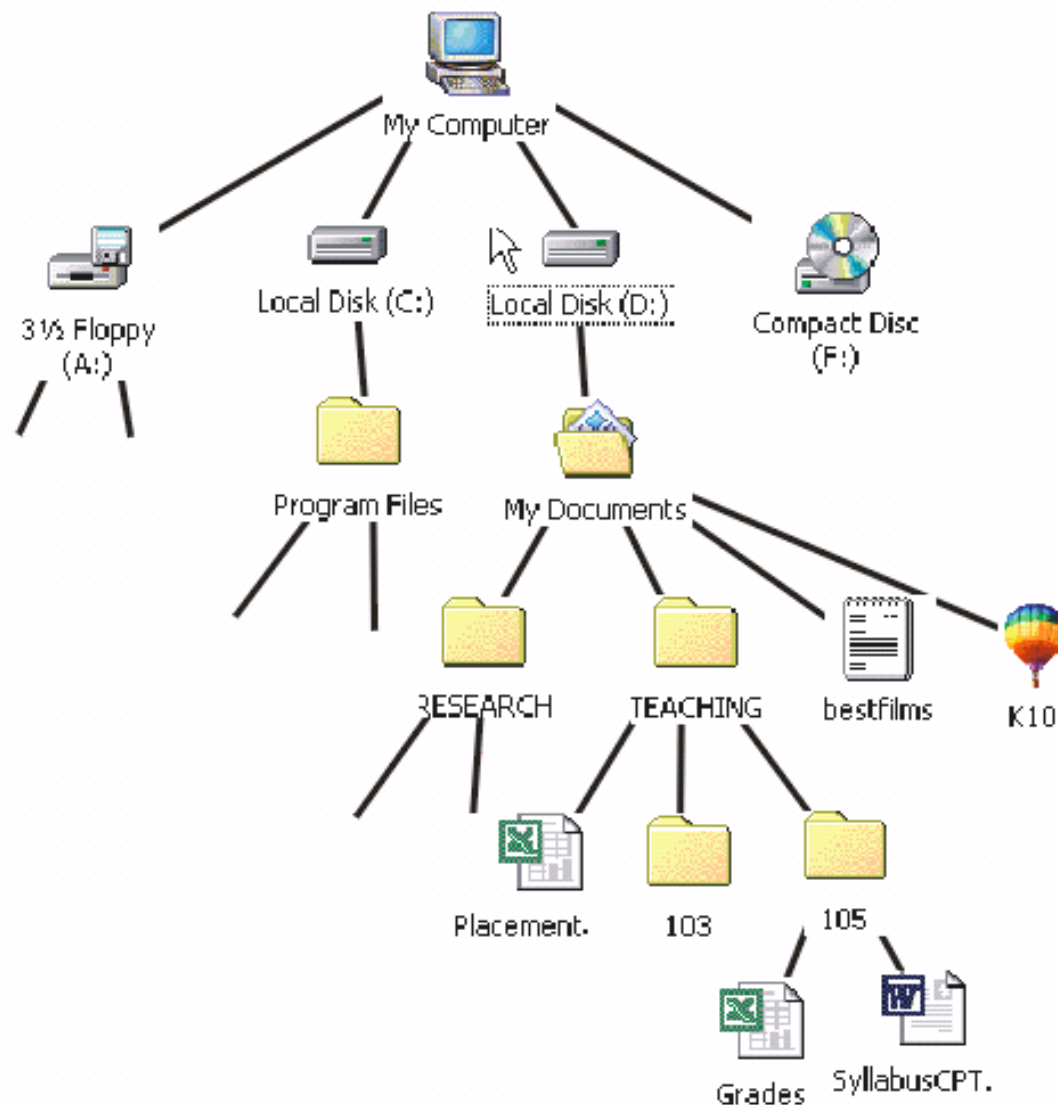


- Svarene på spørsmål om...
 - **multiplisitet**: antall koblinger
 - **navigerbarhet og roller**: retning og navn på kobling
 - **aggregering/komposisjon**: eierskap
 - andre assosiasjonsbeskrankninger
- ...styrer i stor grad hvordan klassen kodes
 - **type felt**, f.eks. enkeltverdi vs. List
 - **konstruktør** med eller uten argumenter for initielle verdier
 - **innkapsling**, f.eks. enkel getter vs. getCount og getElement
 - **validering** og håndtering av **konsistens**

Hierarkiske data

- En veldig vanlig form for assosiasjon
 - mappestruktur
 - organisasjonsstruktur
 - familietre
 - grafikk (HTML, JavaFX, OpenGL)
- To viktige aspekter
 - objekt kan kun være inneholdt i ett objekt
 - strukturen er ofte rekursiv, med ukjent antall nivåer
- Litt mer kinkig koding enn ellers...

Eksempel: Mappestruktur





Mapper og filer

- Hvordan (data)modelleres dette?
- Hvilke kodingsvalg har vi?
- Er assosiasjonene
 - (anti)refleksive: $A \rightarrow A$
 - (anti)symmetriske: $A \rightarrow B \Rightarrow B \rightarrow A$
 - transitive: $A \rightarrow B \ \& \ B \rightarrow C \Rightarrow A \rightarrow C$
- Hva har dette å si for validering?