Repetisjonsforelesning TDAT 1005

# Designmønster

-Hva er et design mønster?

Hvordan vi løser problemer.

-Creational pattern

Hva er god praksis når det kommer til oppretting av objekter

-Structural pattern

Hvordan klassene er bygd opp

-Behavioral pattern

*-Factory method design pattern*

Produsere og delegerer

# Innkapsling

* *Meget viktig, stor grunn til at vi programmerer objekt orientert.*
* Hindre direkte tilgang til tilstanden til et objekt
* Motivasjon
  + Sikre at tilstanden til alle objektene er gyldig
  + Koden for en klasse er ikke avhengig av detaljer i en annen
* Tilgangsmodifikatorer
  + Private
  + Public
  + Protected
  + default

# klasse relasjoner

* Association
  + en til en, en til mange, mange til en, mange til mange.
* Spesial tilfeller
  + Aggregation
    - Forholdet er enveis
      * F.eks. en lommebok som kan ha penger. Vi trenger ikke å ha en lommebok for å ha penger.
  + Compostiton
    - Høy avhengiget mellom to elementer
      * F.eks. et menneske som må ha et hjerte for å leve. Begge er avhengige av hverandre for å eksistere-

# Abstrakte metoder og klasser

* Abstrakte klasser er klasser vi ikke kan lage instanser av
* Poenget med abstrakte klasser er å lage en slags mal som blir arvet av andre klasser for å oppnå polymorfisme.
* Om en klasse inneholder abstrakte metoder vil den være abstrakt men den kan også defineres som abstrakt uten abstrakte metoder.

# Grensesnitt

* Interface klasser brukes for å oppnå multiple arv, som ellers ikke er mulig i java uten.
* Metoder i interfacen må være abstrakte

# Arv og polymorfi

* Vi har polymorfisme når mange klasser er knyttet til hverandre ved arv

# Java

-Arv og polymorfi

-identifisere objekter og se forholdet mellom disse.

-ArrayList

-Unntakshåndtering.

Spørsmål 1 : Forskjellen på extends og implements

-når du bruker interface klasser bruker vi implements

-extand brukes når arver fra en vanlig klasse (abstrakt eller ikke)