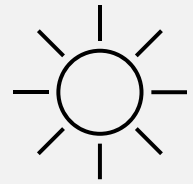
GRUPPE 11 IN1010 – UKE 15

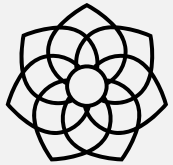
Sivert Fjeldstad Madsen

I DAG

- Programmeringsmønstre
- Behavioral patterns
- Creational patterns
- Structural patterns
- Tips til eksamensøving
- Oppgaver



PROGRAMERINGSMØNSTRE

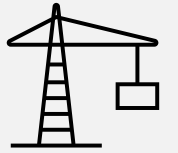


- Når man har programmert en stund vil man kunne begynne å kjenne igjen problemer man har støtt på før
  - Da kan det være lurt å gjenbruke løsninger fra tidligere
- Noen problemer oppstår såpass ofte at det nå finnes standarder for hvordan å løse dem
- Det er blitt definert ganske mange programmeringsmønstre, delt opp i ulike kateogrier
- Noen mener at det at programmeringsmønstre er nødvendige viser til mangler i programmeringsspråket

BEHAVIORAL PATTERNS

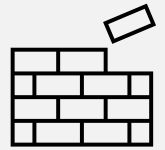
- Denne klassen med designmønstre handler i stor grad om **kommunikasjon** og **interaksjon** mellom objekter
- To mønstre i denne klassen er **Iterator**-mønsteret, og **Observatør**-mønsteret
- Iterator-mønsteret har vi vært borte i når vi kodet iteratorer for listene våre tidligere i semesteret
- Observatør-mønsteret har vi vært innom i forbindelse med ActionListener-objekter

CREATIONAL PATTERNS



- Denne klassen med designmønstre handler i stor grad om å **opprette** objekter på ulike måter
- Et eksempel på dette er **Dependency Injection**, som handler om hvordan objekter får tak i andre objekter (og data) de trenger for å utføre sin oppgave
  - Den enkleste formen for dette er dersom dataen blir sendt inn som parameter til konstruktøren til objektet
- Et annet eksempel er **Singleton**-mønsteret som handler om hvordan man skal passe på at det kan finnes én, og kun én, instans av en gitt klasse
  - Dette kan være nødvendig dersom man f.eks vil opprette en database i et program

STRUCTURAL PATTERNS



- Denne klassen med designmønstre handler i stor grad om **relasjonen** mellom ulike klasser
- Eksempler vi har vært borte i tidligere inkluderer Javas innebygde **Wrapper**-klasser (f.eks. Integer, Character, Boolean), som gjør det mulig å la primitive typer interagere med objekter som ArrayList.
- Et annet mønster fra denne klassen er **Decorator**-mønsteret, som gjør at objekter kan «endre klasse» under kjøretid

- Det er også en fjerde klasse mønstre kalt **Concurrency Patterns**, som handler om trådprogrammering

TIPS TIL EKSAMENSØVING

- Skriv kode uten hjelpemidler – gjerne i notepad eller lignende
- Gå gjennom Trix og gjør oppgaver på ting som ikke sitter helt
  - Gjør også gjerne ukesoppgaver
- **VIKTIG:** pass på at du venner deg til å ikke få fasit hele tiden!
  - Løs *i hvert fall* en hel oppgave før du sjekker en fasit eller lignende
  - Husk at det er sånn eksamen fungerer
  - Dette gjelder ***spesielt*** for tidligere gitte eksamensoppgaver

OPPSUMMERING

- Programmeringsmønstre handler om velkjente måter å løse vanlige problemer på
- Enkelte programmeringsspråk har disse mønstrene innebygd
- Kommer til å bli mer relevant i f.eks IN2000

- Ukas tips:
  - Moderne Java: deklarere variabler uten type