# Kursusafslutning

# Plan

- Opgaveseminar
- Kursusevaluering

# Disposition

1. Hvad går opgaven ud på?

2. Hvor langt er gruppen nået?

3. Hvilke delopgaver mangler at blive løst?

10 minutter +
5 minutter til spørgsmål og kommentarer

# Råd om afleveringsopgaven

**Rapporten**

- Vær ekstra omhyggelig med introduktion og kravspecifikation
  ("Godt begyndt er halvt fuldendt")
- Giv overblik (frem for detaljer)
- Benyt UML og klasseskeletter
- Benyt illustrative figurer

**Program**

- Sørg for god formatering og typografi (benyt IntelliJ's `reformat`)
- Benyt JavaDoc-kommentarer, men medtag ikke HTML-udskrift

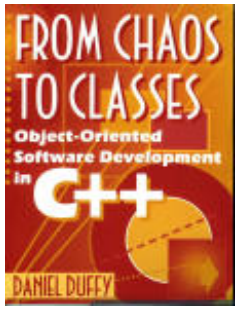
**Eksamen**

- Individuel mundtlig eksamen, 20 minutter inklusiv votering.
- Bedømmelsesgrundlaget er dokumentation, program og mundtlig præstation. Karakteren gives ud fra helhedsindtrykket.

# Plan 1

- Generelt om programmeludvikling
- Objekter og klasser (principper og begreber)
- Objektorienteret programmeludvikling
- Programmering i Java

# Plan 2




- Typer
- Sætninger
- **Klasser**
- Strenge
- **Pakker**
- **Undtagelser**

# Plan 3

- Overlæsning af metoder og konstruktører
- Nedarvning fra klasser
- Nedarvning fra og implementering af grænseflader
- Retningslinjer for design af klasser
- Animering i appletter

# Plan 4

- Introduktion til designmønstre

- Design af generiske komponenter
  - faktorisering
  - generalisering
  - abstrakt kobling

- Design case: animering af algoritmer til sortering

# Plan 5

**Frameworks**

- Kollektioner
- Input/output

Nyt designmønster: Decorator

# Plan 6

- Grafiske komponenter
- Layout
- Hændelser og lyttere
- Rammer og dialoger

Nye designmønstre:
Composite
Command

# Plan 7

- MVC (Model View Controller)
  - designmønsteret Observer

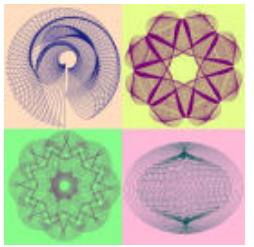
- Iterativ udvikling af et tegneværktøj
  - muselyttere
  - applet/applikation-idiomet
  - designmønsteret State
  - designmønsteret Factory Method

# Plan 8

- Trådbegrebet
- Synkronisering
- Koordinering
- Eksempel: et flertrådet spil

# Plan 9

- Socket-baseret kommunikation
- Fjernmetodekald (RMI)
  - Designmønsteret Proxy
- Databasetilgang (JDBC)



# Designmønstre

For at sikre, at der kun skabes én instans af en klasse:

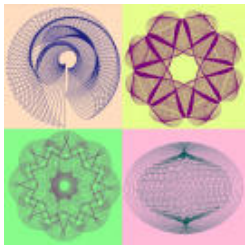
**Singleton**

For at implementere de invariante dele af en algoritme én og kun én gang og overlade det til underklasser at implementere den adfærd, der kan variere:

**Template Method**

For at gøre algoritmer dynamisk udskiftelige:

**Strategy**



For at gennemløbe en samling objekter uden at afsløre deres interne repræsentation:

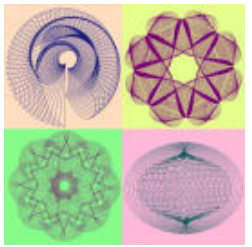
**Iterator**

For at opnå, at et system er uafhængigt af, hvorledes dets produkter skabes:

**Factory**

For at overlade det til underklasser at afgøre, hvilke objekter, der skal skabes:

**Factory Method**



For dynamisk at kunne tilføre et objekt en ekstra funktionalitet:

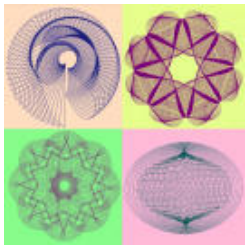
**Decorator**

For at gøre det muligt for klienter at behandle individuelle objekter og samlinger af objekter på en ensartet måde:

**Composite**

For at indkapsle en handling i et objekt, således at handlinger kan overføres som parametre, sættes i kø og eventuelt trækkes tilbage:

**Command**



For at gøre det muligt, at et objekt kan ændre adfærd, når dets interne tilstand ændres:

**State**

Når et objekt skal erstatte et andet objekt:

**Proxy**

Når en ændring i et objekt kræver ændringer i andre objekter:

**Observer**

# Evaluering

Nævn 3 ting, som du er specielt **tilfreds** med i kurset (3 plusser)

Nævn 3 ting, som du er specielt **utilfreds** med i kurset (3 minusser)

# Ugeseddel 10

**2. november - 9. november**

Der arbejdes med afleveringsopgaven.

Rapporten afleveres i 3 eksemplarer til Keld Helsgaun senest **mandag den 9. november klokken $15^{00}$**.

Ved hvert eksemplar skal vedlægges en CD med det udviklede programmel samt en kørselsvejledning.

Eksamen foregår **mandag den 4. januar**.