

Oppsummering del 2

- Læringsmål
- Viktigste Java-elementer
- Eksamen
- Til sist

Læringsmål fra emnebeskrivelsen:

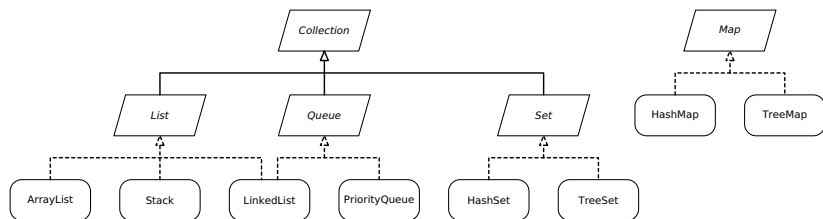
Etter å ha tatt IN1010:

- har du god oversikt over programmeringsspråket Java og du kan bruke det til å løse reelle problemer av middels størrelse
- behersker du avanserte objektorienterte mekanismer som subclasser, polymorfi og interface
- har du oversikt over noen grunnleggende datastrukturer (spesielt lenkede lister) og du kan programmere de viktigste operasjonene på dem
- kan du utvikle robuste og pålitelige programmer med godt objektorientert design, og du kan finne alternative løsninger for et gitt problem og vurdere fordeler og ulemper ved de ulike løsningene
- har du kunnskap om parallelle programmer med tråder og du kan benytte dette i enkle programmer
- kjenner du til hendelseshåndtering og kan skrive enkle programmer som håndterer hendelser

NB!

Læringskravet i IN1010 er ikke først og fremst å kunne et pensum, men å kunne lage programmer (rimelig raskt) ved å bruke programmeringsteknikker som er forelest og som innøves i de forskjellige studieaktivitetene. Det er viktig å vite at man ikke kan lese seg til å nå dette målet.

Datastrukturer



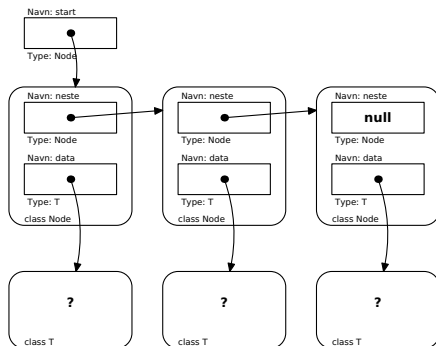
Vi har sett på **List**, **ArrayList**, **LinkedList** (som er aller viktigst!), **Stack**, **Queue** og **PriorityQueue**.

Liste.java

```
interface Liste<T> {  
    public abstract int size();  
    public abstract void add(T x);  
    public abstract void set(int pos, T x);  
    public abstract T get(int pos);  
    public abstract T remove(int pos);  
}
```

- Den generelle datastrukturen kan defineres med et *interface*.
- Interface og klasser kan gjøres *generiske* ved å gi *type-parametre*.

Lenkelister implementeres slik



```
class Lenkeliste<T> implements Liste<T> {
    class Node {
        Node neste = null;
        T data;

        Node(T x) { data = x; }
    }
    private Node start = null;
}
```

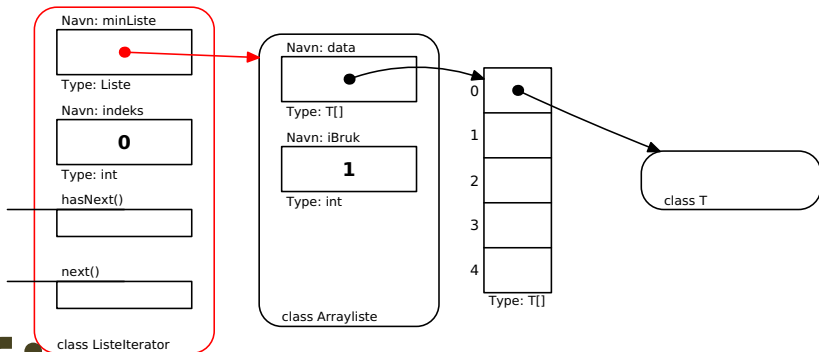
Noen grensesnitt er viktigere enn andre

Tre viktige grensesnitt

Comparable har en metode compareTo.

Iterable har en metode iterator.

Iterator har metodene hasNext og next.



Rekursjon

Programkode som kaller seg selv, kalles **rekursiv kode**.

- Noen problemer er rekursive av natur (for eksempel en kompilator eller Hanois tårn)
- Rekursiv programmering er spesielt nyttig for programmer som skal søke ulike veier til en mulig løsning (for eksempel en labyrint eller sjakkspill)
- Noen programmerere foretrekker rekursive løsninger.

GUI-programmering med JavaFX

JavaFX bruker begreper fra teateret:

Teater



class Stage

Scene



class Scene

Kulisser

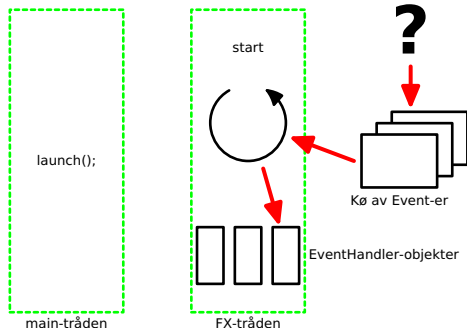


class Pane

Hendelsesorientert programmering

I dette *paradigmet* ligger FX-tråden og venter på hendelser som kommer via ActionEvent-objekter i en kø.

Hendelsesløyken i FX-tråden vil ta ActionEvent-ene etter tur, og den korrekte EventHandler-en vil bli kalt.



Vi vil gjerne vite hva dere mener om kurset

Kurskritikken

Fagutvalget ved Ifi (dvs studentene) arrangerer hvert semester **kurskritikken**.

Vi faglærere ber dere svare på den. Vi vil *veldig gjerne* vite deres ærlige mening om emnet IN1010 og undervisningen.

Og hvordan trives dere?

Hvordan trives du på Ifi?

Hva bidrar positivt/ negativt til motivasjon, innsats, læring?

Fikk du en bra velkomst som ny student?

Har vi et studentmiljø som inkluderer alle?

Har du fått venner på studiet?

Hvilke rom bruker du, hva savner du?

Savner du utfordringer??

Og hvordan trives dere?

- Studielabens læringsmiljøundersøkelse gjennomføres for andre gang i disse dager
- Sees i sammenheng med studiestart-undersøkelsen om motivasjon og forventninger
- SUPPLERER Fagutvalgets kurskritikk (som kommer om noen uker og er VIKTIG tilbakemelding på undervisning i enkeltemner)
- Resultatene blir tilgjengelig for dere – og brukes av administrasjon og faglærere!
- Alle førsteårsstudenter bør spandere disse 10 minuttene av hensyn til studiehverdagen fremover - og kommende studenter.

Resultater fra i fjor:

<http://www.mn.uio.no/ifi/studier/ressurser/studielaben/leringsmiljo/>

Hvor og når?

Eksamen

Når

4. juni kl 09:00-15:00

Hvor

Silurveien 2



Hvor og når?

Hvordan komme dit

- T-banen linje 3 mot Kolsås til Åsjordet
- Buss 23 til **Blokkajordet**



Tegning

Tegning

Datastrukturer etc kan tegnes med svart eller blå kulepenn på spesialark. Husk å fylle ut kodenummeret og annen informasjon *med én gang*.

Fyll inn oppgavetide og annen informasjon på alle arkene. Fill in question code and other information on every sheet.

Oppgavetide Question code	Dato Date	Emnekode Subject code	Kandidatnummer Candidate number	Oppgavenummer Question number	Sidenr Page number

Tegningsark Drawing area

Programmering

Programmering skjer i en egen editor som forstår litt av Java:

- Reserverte ord, kommentarer etc fargekodes.
- Editoren hjelper med parenteser og innrykk.

Men ...

- Det skjer ingen navnebinding, typesjekk eller lignende.
- Det er ikke mulig å kjøre koden.

Prøveeksamen

Fjorårets eksamen er tilgjengelig som prøveeksamen.
Gå til nettsiden

<http://uio.inspera.no/>

Den blir gjennomgått i Simula-auditoriet torsdag 31. mai
kl 10.15.

Pensum

Se nettsiden; viktigst er

- Læreboken til Horstmann (inkludert kapittel 20 om tråder)
- Forelesningene
- De obligatoriske oppgavene

Hjelpemidler under eksamen

Alle trykte og skrevne hjelpemidler (som dere får plass til på skrivebordet).

En enkel kalkulator er tillatt, men vil ikke være til noen hjelp.

Stein og jeg går gjennom lokalet ca kl 10 for å oppklare uklarheter i oppgavene.

Noen tips til sist

- Les over hele oppgavesettet før du begynner å svare.
- Det viktigste er å overbevise *sensor* om at dere kan programmere.
- Kommentarer er viktige der dere gjør noe spesielt; ellers kan de droppes.
- Småfeil (manglende `;`, manglende **import**, blande `'` og `"` etc) teller vanligvis ikke.
- Det er bedre å svare litt på alt enn perfekt på noe og ha det andre blankt.

Fra Stein og meg:
Lykke til på eksamen!