Pragmatisk programmering

Generiske beholdere

Tema for uken

- Bittelitt pensum
  - Generiske beholdere
  - To (tre) nye interfaces:
    - Comparable
    - Iterator / Iterable
- Oppgaver / samkoding
- Quiz! Typekonvertering
- Andre ønsker/behov?
  - Behov for å bruke mer tid på forrige ukes pensum?
  - (dette tar forrang!)

Generiske beholdere

- Noen ganger ønsker vi at en klasse/grensesnitt skal kunne operere på flere ulike datatyper

```
1   interface Liste<T> {
2       public int antall();
3       public void settInnFørst(T elem);
4       public void settInnSist(T elem);
5       public void fjernFørste();
6       public void fjernSiste();
7       public T hentFørste();
8       public T hentSiste();
9       public T hent(int indeks);
10  }
```

Typevariabel
(én stor bokstav)

Generiske beholdere

- Nøyaktig datatype instansieres idet objektet opprettes

```
class LenkeListe<T> implements Liste<T> {
    // implementasjon av alle metoder i interfacet
}
```

```
class Main {
    public static void main(String[] args) {
        LenkeListe<String> liste = new LenkeListe<>();
    }
}
```

`Comparable`

- Grensesnittet `Comparable<T>` brukes for å definere rekkefølgen av objekter i en gitt klasse
- Krever implementasjon av én metode: `compareTo()`
- Returnerer et *heltall*

Comparable

```java
class Person implements Comparable<Person> {
    String navn;
    int alder;

    public Person(String navn, int alder) {
        this.navn = navn;
        this.alder = alder;
    }

    @Override
    public int compareTo(Person annen) {
        return alder - annen.alder;
    }
}
```

Iterable og Iterator

- Et objekt av en klasse som implementerer `Iterable<T>` er itererbar
  - Å iterere over et objekt vil si å løpe gjennom elementene
  - Hvis `strengliste` er en itererbar liste, kan vi gå gjennom strengene slik:

```
for (String s : strengliste) {
    System.out.println(s);
}
```

Iterable og Iterator

- `Iterable<T>`

Metodenavn	**Returtype**	**Beskrivelse**
iterator()	Iterator	Returnerer en iterator over elementer av typen T

- `Iterator<T>`

Metodenavn	**Returtype**	**Beskrivelse**
hasNext()	boolean	Returnerer `true` hvis det er flere elementer å iterere over
next()	T	Returnerer det neste elementet i iterasjonen

```java
import java.util.Iterator;

class EnkelArrayListe<T> implements Iterable<T> {

    Node første;

    private class Node {
        T elem;
        Node neste;

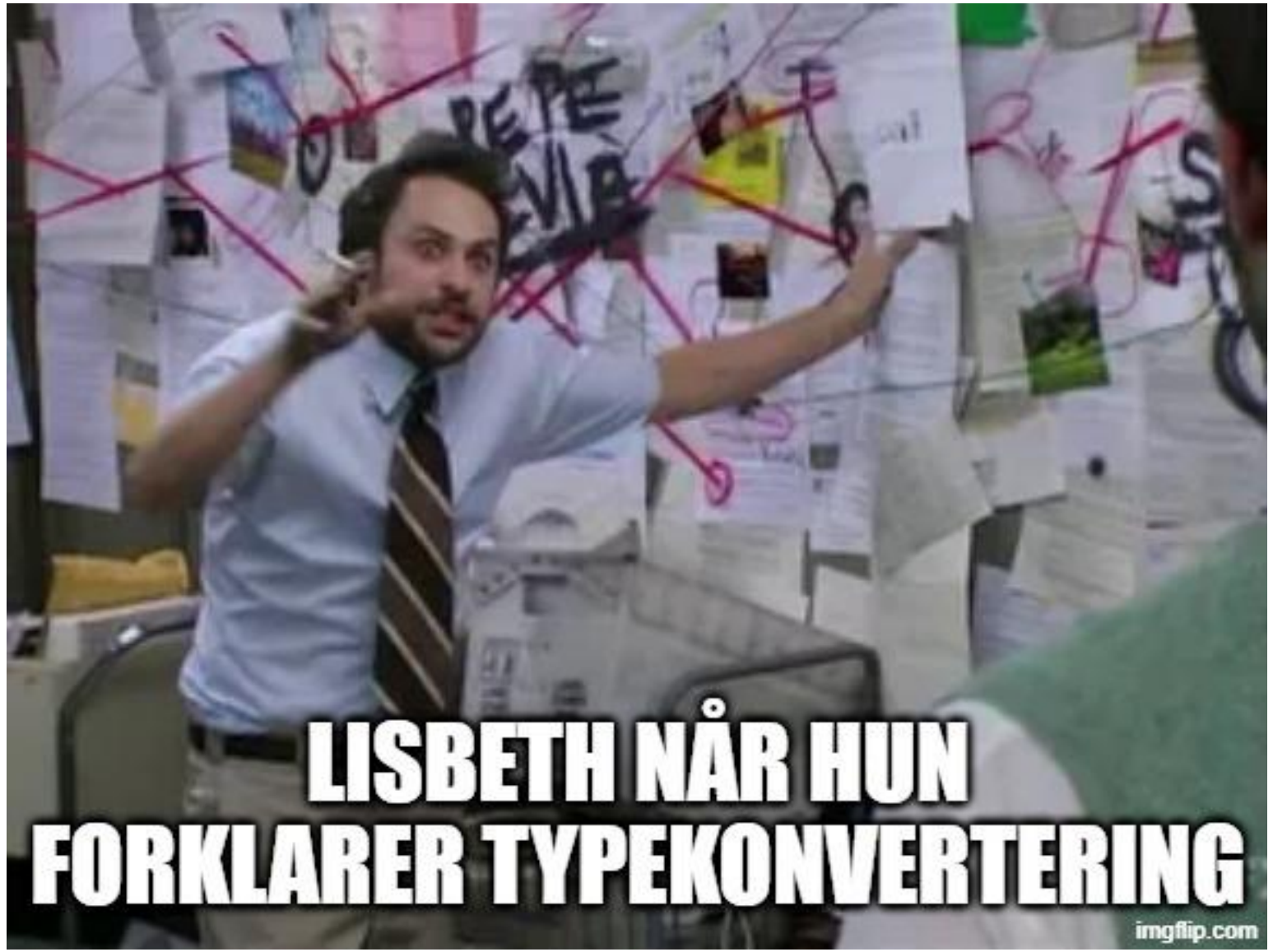
        Node(T elem) {
            this.elem = elem;
        }
    }

    private class ListeIterator implements Iterator<T> {
        Node it = første;

        @Override
        public boolean hasNext() {
            return it != null;
        }

        @Override
        public T next() {
            T elem = it.elem;
            it = it.neste;
            return elem;
        }
    }

    @Override
    public Iterator iterator() {
        return new ListeIterator();
    }
}
```


LISBETH NÅR HUN
FORKLARER TYPEKONVERTERING
imgflip.com

```java
class FlaskeBruk {
    public static void main (String[] a) {
        Drikkeflaske drikke = new Drikkeflaske();
        Plastflaske plast = new Plastflaske();
        Miljøplastflaske miljø = new Miljøplastflaske();
        Pappflaske papp = new Pappflaske();
        Glassflaske glass = new Glassflaske();
        PappflaskeMedPant pappPant = new PappflaskeMedPant();

        Flaske f1 = pappPant;
        Plastflaske pl1 = miljø;

        // Hvilke av disse setningene er gyldige som neste setning?
        Pant p1 = plast;
        Pant p2 = pappPant;
        Pant p3 = miljø;
        Pant p4 = (Pant)f1;
        Pant p5 = (Pant)pl1;
        Pant p6 = f1;
        Pant p7 = pl1;
        Papirsøppel ps1 = papp;
        Papirsøppel ps2 = pappPant;
        Papirsøppel ps3 = (Papirsøppel)f1;
        Papirsøppel ps4 = f1;
        Flaske f2 = drikke;
        Flaske f3 = pl1;
        Plastflaske pl2 = miljø;
        Pappflaske pf1 = f1;
    }
}

interface Pant {}
interface Papirsøppel {}

abstract class Flaske {}
class Drikkeflaske extends Flaske {}
class Plastflaske extends Flaske implements Pant {}
class Miljøplastflaske extends Plastflaske {}
class Pappflaske extends Flaske implements Papirsøppel {}
class Glassflaske extends Flaske {}
class PappflaskeMedPant extends Pappflaske implements Pant {}
```

Pant p1 = plast;

Pant p2 = pappPant;

Pant p3 = miljø;

Pant p4 = (Pant)f1;

Pant p5 = (Pant)pl1;

Pant p6 = f1;

Pant p7 = pl1;

Papirsøppel ps1 = papp;

Papirsøppel ps2 = pappPant;

Papirsøppel ps3 = (Papirsøppel)f1;

Papirsøppel ps4 = f1;

Flaske f2 = drikke;

Flaske f3 = pl1;

Plastflaske pl2 = miljø;

Pappflaske pf1 = f1;

```java
class FlaskeBruk {
    public static void main (String[] a) {
        Drikkeflaske drikke = new Drikkeflaske();
        Plastflaske plast = new Plastflaske();
        Miljøplastflaske miljø = new Miljøplastflaske();
        Pappflaske papp = new Pappflaske();
        Glassflaske glass = new Glassflaske();
        PappflaskeMedPant pappPant = new PappflaskeMedPant();

        Flaske f1 = pappPant;
        Plastflaske pl1 = miljø;

        // Hvilke av disse setningene er gyldige som neste setning?
        Pant p1 = plast;
        Pant p2 = pappPant;
        Pant p3 = miljø;
        Pant p4 = (Pant)f1;
        Pant p5 = (Pant)pl1;
        Pant p6 = f1;
        Pant p7 = pl1;
        Papirsøppel ps1 = papp;
        Papirsøppel ps2 = pappPant;
        Papirsøppel ps3 = (Papirsøppel)f1;
        Papirsøppel ps4 = f1;
        Flaske f2 = drikke;
        Flaske f3 = pl1;
        Plastflaske pl2 = miljø;
        Pappflaske pf1 = f1;
    }
}

interface Pant {}
interface Papirsøppel {}

abstract class Flaske {}
class Drikkeflaske extends Flaske {}
class Plastflaske extends Flaske implements Pant {}
class Miljøplastflaske extends Plastflaske {}
class Pappflaske extends Flaske implements Papirsøppel {}
class Glassflaske extends Flaske {}
class PappflaskeMedPant extends Pappflaske implements Pant {}
```

	OK	Feil
Pant p1 = plast;	✔	
Pant p2 = pappPant;	✔	
Pant p3 = miljø;	✔	
Pant p4 = (Pant)f1;	✔	
Pant p5 = (Pant)pl1;	✔	
Pant p6 = f1;		✔
Pant p7 = pl1;	✔	
Papirsøppel ps1 = papp;	✔	
Papirsøppel ps2 = pappPant;	✔	
Papirsøppel ps3 = (Papirsøppel)f1;	✔	
Papirsøppel ps4 = f1;		✔
Flaske f2 = drikke;	✔	
Flaske f3 = pl1;	✔	
Plastflaske pl2 = miljø;	✔	
Pappflaske pf1 = f1;		✔