



IN1010 - Seminar 3

- python → Java
 - I/O
- 



Oppmøteregistering:

<https://nettskjema.no/a/183204>



Praktisk

Oblig 1 er nå lagt ut, frist 1. februar kl 23.59

Husk å sjekke emnesiden regelmessig

Vise hvor ressurser for hver uke ligger

<https://www.uio.no/studier/emner/matnat/ifi/IN1010/v21/index.html>

Tips for å komme i gang med oblig 1

Tegn datastrukturtegning

- Hvilke klasser trenger man og hvilke referanser må eksistere?
- Bruk "Programdesign"-delen, mange hint her
- Tegn etterhvert som du leser gjennom oppgaveteksten
- Til slutt: sjekk at dersom du implementerer det du har tegnet, vil systemet da kunne gjøre alt oppgaven ber om?
- Les oppgaveteksten flere ganger

Datastrukturtegningen kan hjelpe deg å få bedre forståelse for systemet, og med bedre forståelse vil du gjør færre feil, og skrive koden raskere.

Repetisjon forrige uke

NullPointerException

Viktig å huske at array i java er fixed size, dette er annerledes enn liste i python, vi kan endre innhold, men ikke lengden

```
1 class ArrayEksempel{
2     public static void main(String[] args) {
3         Katt[] kattArray = new Katt[10];
4
5         kattArray[0] = new Katt("Pus", 1);
6     }
```

Navn: kattArray



Type: array



public Katt(String navn, int alder)

public String hentNavn()

Navn: navn



Type: String

Navn: alder



Type: int

"Pus"

NullPointerException

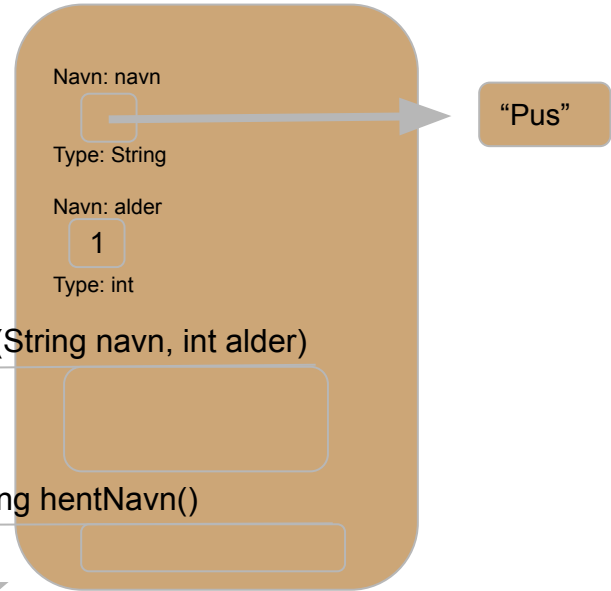
Viktig å huske at array i java er fixed size, dette er annerledes enn liste i python, vi kan endre innhold, men ikke lengden

```
1 class ArrayEksempel{
2     public static void main(String[] args) {
3         Katt[] kattArray = new Katt[10];
4
5         kattArray[0] = new Katt("Pus", 1);
6
7         kattArray[0].hentNavn() Evaluerer til "Pus"
8
9         kattArray[1].hentNavn() null.hentNavn()
                                Gir error:
                                java.lang.NullPointerException
10    }
11 }
```

Navn: kattArray



Type: array



Repetisjon denne uken

Arv

Både Fjellrev og Katt er Dyr, det hadde vært bra å kunne kommunisere det på en måte i koden vår!



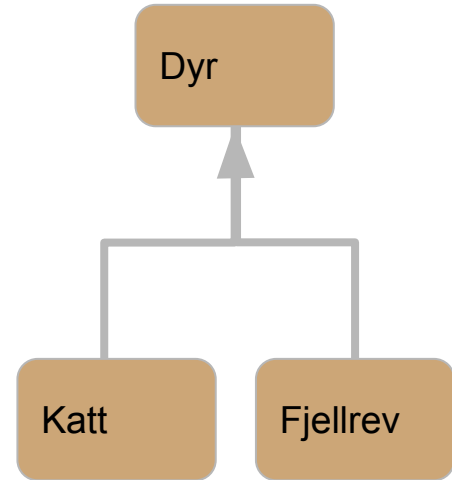
Klassehierarki

Alle Katter er Dyr

Alle Fjellrever er Dyr

Ingen Katter er Fjellrever

Ingen Fjellrever er katter



Klassehierarki

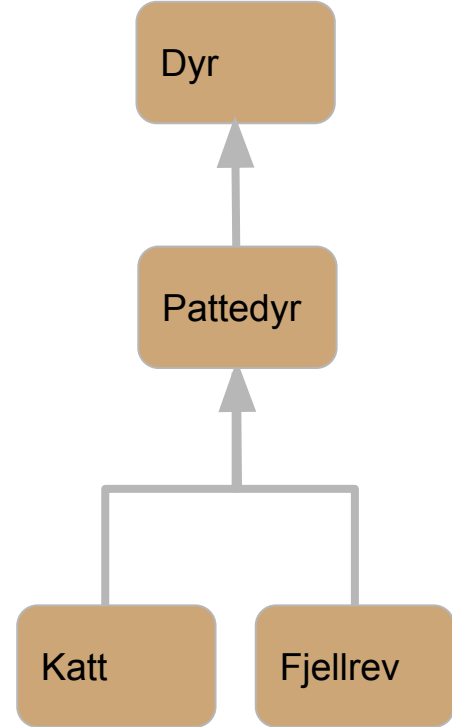
Alle Katter er Dyr

Alle Fjellrever er Dyr

Ingen Katter er Fjellrever

Ingen Fjellrever er katter

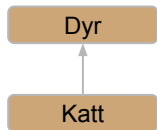
Og man kan ha mange nivåer, f.eks. kan vi legge til pattedyr



Katt og dyr

Her definerer jeg at katter har bostedsadresse, mens andre dyr ikke nødvendigvis har det.

Alle dyr har navn og alder!



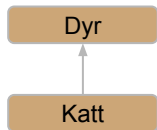
```
1 class Katt extends Dyr{
2
3     private String bostedsadresse;
4
5     public void settBosted(String bosted){
6         bostedsadresse = bosted;
7     }
8     public String hentBosted(){
9         return bostedsadresse;
10    }
11 }
```

```
1 class Dyr{
2     protected int alder;
3     protected String navn;
4
5     public void settAlder(int alder){
6         this.alder = alder;
7     }
8     public int hentAlder(){
9         return alder;
10    }
11    public void settNavn(String navn){
12        this.navn = navn;
13    }
14    public String hentNavn(){
15        return navn;
16    }
17    public void skrivUtInfo(){
18        System.out.println("Navn: " + navn);
19        System.out.println("Alder: " + alder);
20    }
21 }
```

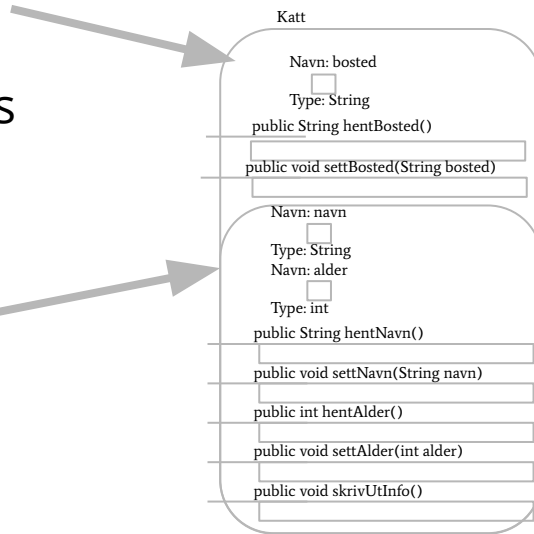
Katt og dyr

Her definerer jeg at katter har bostedsadresse, mens andre dyr ikke nødvendigvis har det.

Alle dyr har navn og alder!



```
1 class Katt extends Dyr{
2
3     private String bostedsadresse;
4
5     public void settBosted(String bosted){
6         bostedsadresse = bosted;
7     }
8     public String hentBosted(){
9         return bostedsadresse;
10    }
11 }
```



```
1 class Dyr{
2     protected int alder;
3     protected String navn;
4
5     public void settAlder(int alder){
6         this.alder = alder;
7     }
8     public int hentAlder(){
9         return alder;
10    }
11    public void settNavn(String navn){
12        this.navn = navn;
13    }
14    public String hentNavn(){
15        return navn;
16    }
17    public void skrivUtInfo(){
18        System.out.println("Navn: " + navn);
19        System.out.println("Alder: " + alder);
20    }
21 }
```

Alle Katter er Dyr

Alle katter får automatisk alle egenskapene til Dyr!

Husk at metoden settNavn() og HentNavn() var metoder i klassen Dyr, og ikke metoder i klassen Katt, siden alle Dyr har navn (i dette programmet).

Det går helt fint å legge en referanse til et objekt av klassen Katt i variabelen katt2, fordi Katt er et Dyr

```
1 class TestArv{
2     public static void main(String[] args) {
3         Katt katt1 = new Katt();
4         Dyr katt2 = new Katt();
5
6         // Dette er lov siden begge er Dyr
7         katt1.settNavn("Tiger");
8         katt2.settNavn("Max");
9         System.out.println(katt1.hentNavn());
10        System.out.println(katt2.hentNavn());
}
```

Arv

En referanse til et Katteobjekt
kan ligge i en variabel med type
Dyr og Katt

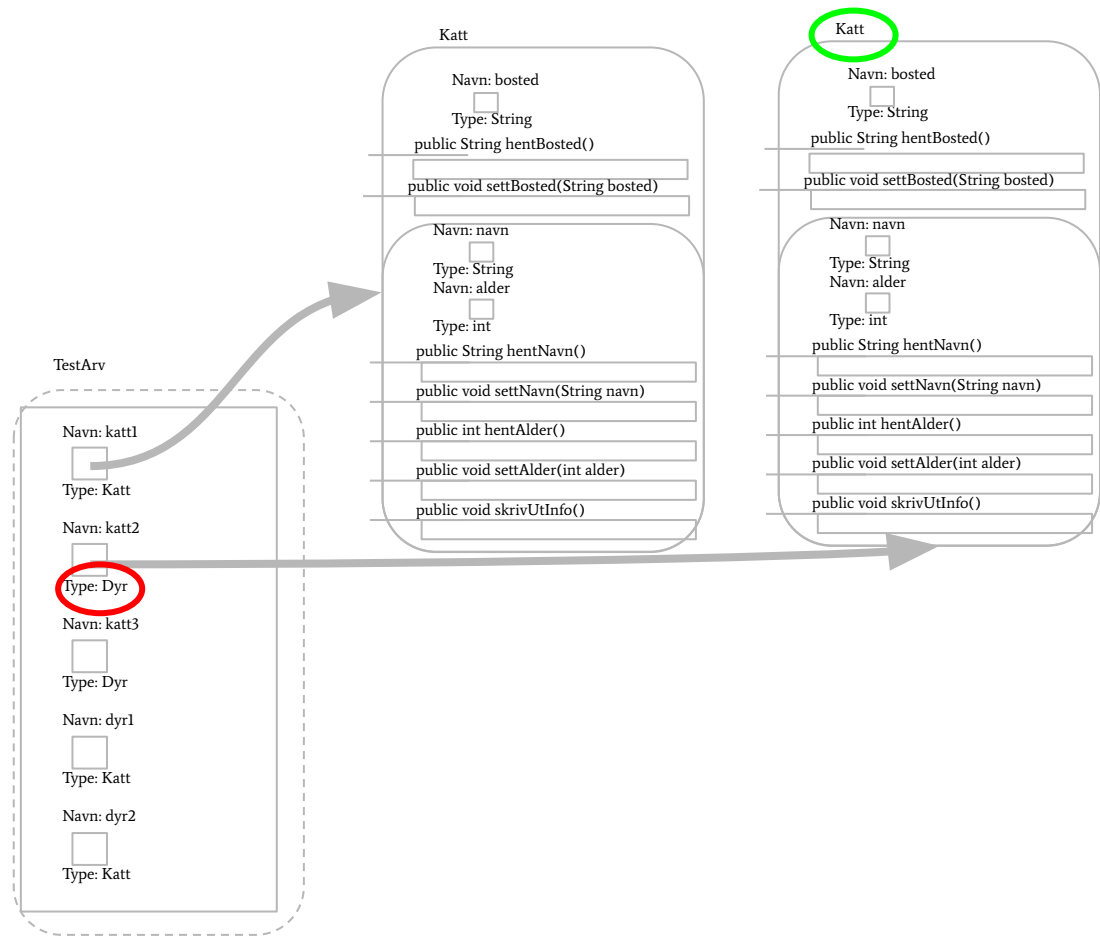
En referanse til et Dyreobjekt
kan **ikke** ligge i en variabel med
type Katt, heller ikke hvis vi
caster.

Generelt: god stil å unngå
casting så mye som du kan!

```
1 class TestArv{
2     public static void main(String[] args) {
3         // Lov!
4         Katt katt1 = new Katt();
5         Dyr katt2 = new Katt();
6         // Lov
7         Katt katt3 = (Katt) katt2;
8         // Ikke lov
9         Katt Dyr1 = new Dyr(); Error
10        // heller ikke lov
11        Katt Dyr2 = (Katt) new Dyr(); Error
12    }
```


Datastrukturtegning

```
1 class TestArv{
2     public static void main(String[] args) {
3         // Lov!
4         Katt katt1 = new Katt();
5         Dyr katt2 = new Katt();
6         // Lov
7         Katt katt3 = (Katt) katt2;
8         // Ikke lov
9         Katt dyr1 = new Dyr(); Error
10        // heller ikke lov
11        Katt dyr2 = (Katt) new Dyr(); Error
12    }
```

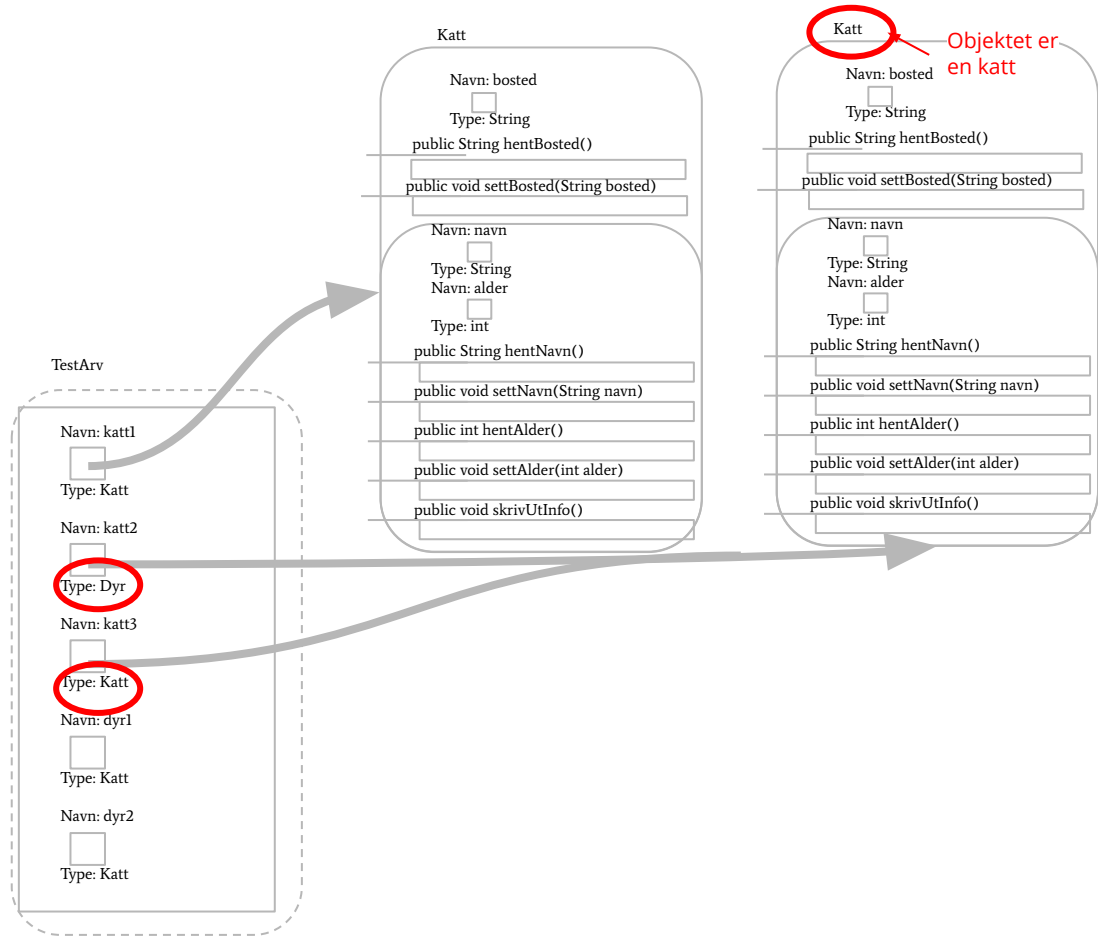
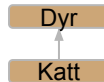


Datastrukturtegning

```
1 class TestArv{
2     public static void main(String[] args) {
3         // Lov!
4         Katt katt1 = new Katt();
5         Dyr katt2 = new Katt();
6         // Lov
7         Katt katt3 = (Katt) katt2;
8         // Ikke lov
9         Katt dyr1 = new Dyr(); Error
10        // heller ikke lov
11        Katt dyr2 = (Katt) new Dyr(); Error
12    }
```

Casting betyr at vi forteller kompilatoren at vi vet hva vi driver med! Vi vet at objektet som katt2 referer til er en Katt. Hvis vi ikke caster får vi kompilingsfeil

Må caste her fordi typen til variabelen katt2 er Dyr.

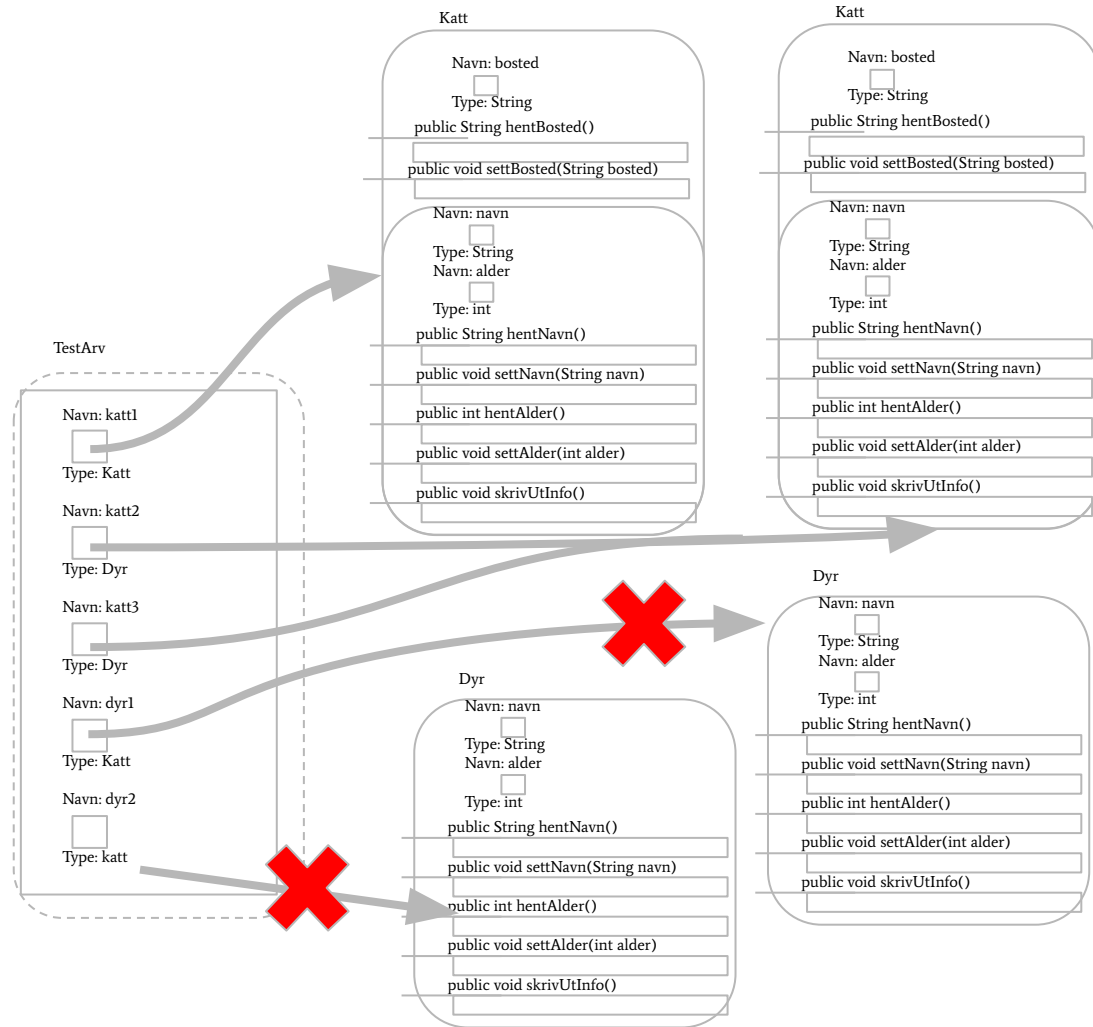


Objektet er en katt

Datastrukturtegning

```
1 class TestArv{
2     public static void main(String[] args) {
3         // Lov!
4         Katt katt1 = new Katt();
5         Dyr katt2 = new Katt();
6         // Lov
7         Katt katt3 = (Katt) katt2;
8         // Ikke lov
9         Katt dyr1 = new Dyr(); Error
10        // heller ikke lov
11        Katt dyr2 = (Katt) new Dyr(); Error
12    }
```

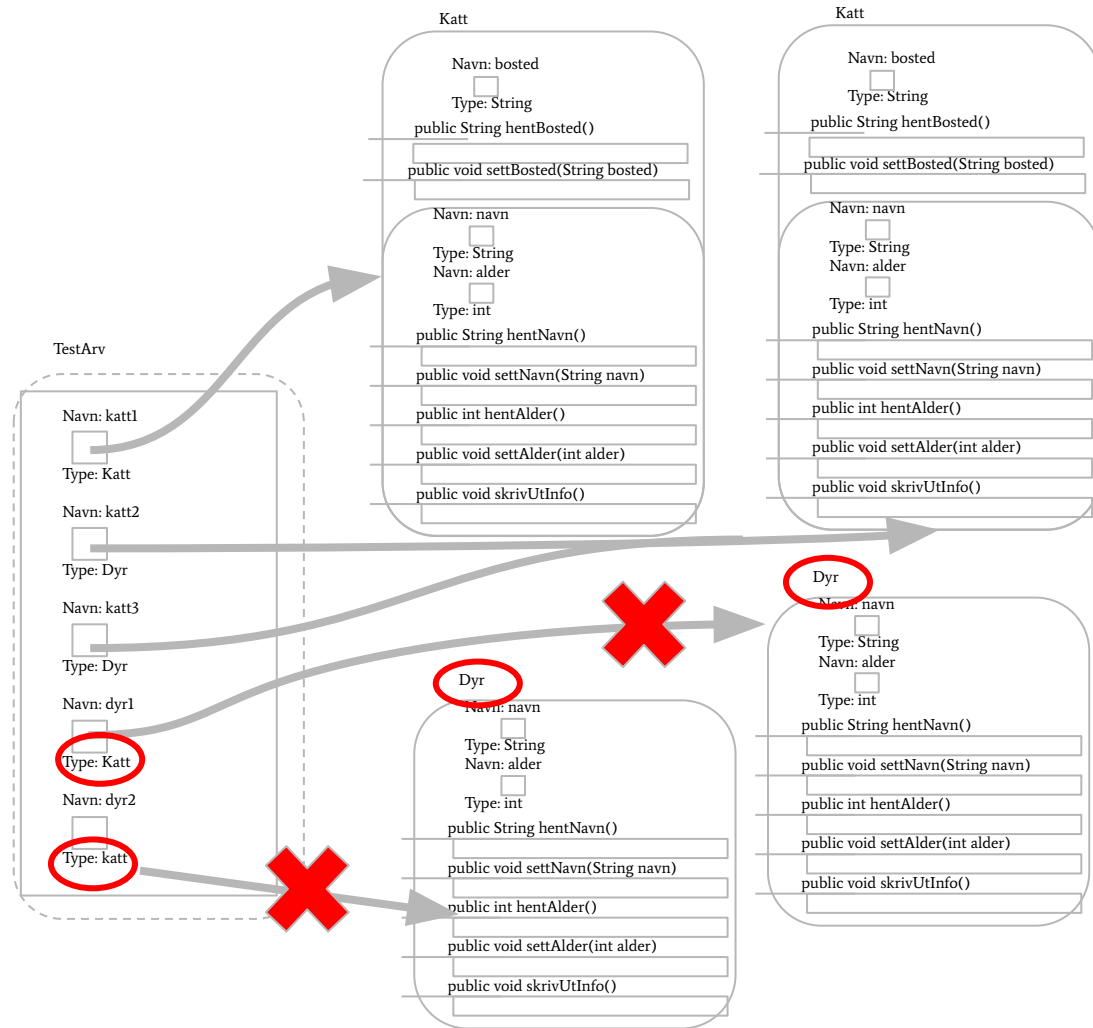
1. Vi får kompileringsfeil
 - "Løses" med casting, kompilatoren klager ikke fordi vi har sagt at vi vet hva vi driver med
2. Dyr er ikke Katt: run time error
 - kan ikke løses



Datastrukturtegning

```
1 class TestArv{
2     public static void main(String[] args) {
3         // Lov!
4         Katt katt1 = new Katt();
5         Dyr katt2 = new Katt();
6         // Lov
7         Katt katt3 = (Katt) katt2;
8         // Ikke lov
9         Katt dyr1 = new Dyr(); Error
10        // heller ikke lov
11        Katt dyr2 = (Katt) new Dyr(); Error
12    }
```

1. Vi får kompileringsfeil
 - "Løses" med casting, kompilatoren klager ikke fordi vi har sagt at vi vet hva vi driver med
2. Dyr er ikke Katt: run time error
 - kan ikke løses



Arv

En referanse til et Katteobjekt
kan ligge i en variabel med type
Dyr og Katt

En referanse til et Dyreobjekt
kan **ikke** ligge i en variabel med
type Katt, heller ikke hvis vi
caster.

Generelt: god stil å unngå
casting så mye som du kan!

```
1  class TestArv{
2      public static void main(String[] args) {
3          // Lov!
4          Katt katt1 = new Katt();
5          Dyr katt2 = new Katt();
6          // Lov
7          Katt katt3 = (Katt) katt2;
8          // Ikke lov
9          Katt Dyr1 = new Dyr();   Error
10         // heller ikke lov
11         Katt Dyr2 = (Katt) new Dyr();   Error
12     }
```

Error-håndtering

Her ser dere to eksempler på kode som gjør det samme (åpner en fil)

- 1) Med error håndtering (try/catch)
- 2) Uten error-håndtering, sender problemet viderer (throws).

Da trenger man try/catch når man kaller på denne metoden ellers vil programmet terminere (avslutte). (Siden dette er main som "Throws exception" vil programmet terminere ved error.)

```
1 import java.util.Scanner;
2 import java.io.File;
3
4 // Med error-håndtering
5 class FraFil1{
6     public static void main(String[] args){
7         // Åpne datafilen:
8         Scanner fil = null;
9         try {
10             fil = new Scanner(new File(filNavn));
11         } catch (Exception e) {
12             System.out.println("Kan ikke lese " + filNavn + "!");
13             System.exit(1);
14         }
15     }
16 }
17
18 // Uten error-håndtering
19 class FraFil2{
20     public static void main(String[] args) throws Exception {
21         Scanner fil = new Scanner(new File("tekst.txt"));
22     }
23 }
24 }
```

```
jonbon@jons-macbook-pro uke2 % java FraFil1
Med errorhåndtering
Kan ikke lese filen !
jonbon@jons-macbook-pro uke2 % java FraFil2
Uten errorhåndtering
Exception in thread "main" java.io.FileNotFoundException: tekst.txt (No such file or directory)
at java.base/java.io.FileInputStream.open0(Native Method)
at java.base/java.io.FileInputStream.open(FileInputStream.java:219)
at java.base/java.io.FileInputStream.<init>(FileInputStream.java:157)
at java.base/java.util.Scanner.<init>(Scanner.java:639)
at FraFil2.main(Errorhåndtering.java:21)
```



Oppmøteregistering:

<https://nettskjema.no/a/183204>



Send Marlen (chat og tegning) en direkte melding i chatten

Vil du jobbed sammen med noen andre ? (ja /nei)

Svar gjerne også om svaret skulle være nei

Jobbe med oppgaver

Breakoutrooms

1. Slå på kamera og ha en presentasjonsrunde
2. Diskuter:
 - a. Hva betyr det at en variabel med type Dyr kan ha en referanse til et objekt av klassen Katt, men ikke omvendt?
(Variabel av type A kan referere til objekt av B, men ikke omvendt)
 - b. Hva betyr kodeordet Protected? (Google gjerne eller se i forelesning hvis dere ikke husker!)
Hva er forskjellen på private, public og protected?
 - c. Hva gjør metoden instanceof()?
3. Jobb sammen med ukesoppgavene, de ligger på emnesiden -> grupper
 - a. Enten ved at én deler skjerm eller med jdoodle.com eller med codecollab.io/
 - b. OBS: både jdoodle og codecollab er gratistjeneseter som UiO ikke har avtale med, sannsynligvis vil de samle data om dere. Dere kan fint løse oppgavene uten å bruke disse tjenestene!
4. Bruk "ask for help"-knappen for å få hjelp 😊
5. Vi møtes her igjen for å gå gjennom oppgavene til slutt (dere bestemmer hvilke)

Ris, ros, forslag ?

<https://nettskjema.no/a/181775>

Tegning oppgave 4

Parkeringshus

