Velkommen!

...

Johanna
johannph på mattermost
johannph@uio.no på mail!

Kort: Praktisk informasjon

- Undervisningstilbud
  - https://www.uio.no/studier/emner/matnat/ifi/IN1010/v21/undervisningstilbud/
  - Jeg har konkrete spørsmål/problemer med min kode -> Labtime!
  - Jeg vil ha mer liveprogrammering -> Plenumstime!
  - Jeg vil jobbe med andre (og kanskje en kjapp recap av forelesning) -> Gruppetime!
  - Jeg vil ha en recap av de vanskeligste konseptene fra forelesning -> Repetisjonsgruppe!
- Oblig 2, frist Mandag 15.02 kl 23.59
  - Tips: tegn datastrukturtegning
  - Da får dere får bedre forståelse for programmet og dere unngår feil
  - Utsatt frist: se beskjed på emnesiden
  - Husk å skrive: “Jeg ønsker tilbakemelding” i devilry dersom dere vil ha tilbakemelding! Skriv gjerne hva dere vil ha tilbakemelding på.

Selvpleie

Tipper det er kjipt for mange å sitte hjemme hele dagen, spesielt hvis man er mye alene.

Legger med noen slides med ting som kanskje kan hjelpe litt.

En liten utblåsing: FYFAEN DET ER DRITT Å MÅTTE MOTIVERE SEG SELV TIL Å JOBBE HELE DAGEN, takk for meg ❤️

Dere er sååå flinke som dukker opp i gruppetime, masse masse kudos til dere ❤️❤️

Det vi driver med er skikkelig vanskelig: https://www.aftenposten.no/meninger/debatt/i/kRpoBk/studentene-overlates-til-seg-selv-under-koronaen

Noen å snakke med

UiO forvei

https://www.mn.uio.no/studier/forvei/

SiO

https://www.sio.no/helse/noen-%C3%A5-snakke-med

Oversikt over andre hjelpetelefoner og nettsteder

https://psykiskhelse.no/hjelpetelefoner-og-nettsteder

Oslo kommune

https://www.oslo.kommune.no/helse-og-omsorg/psykisk-helse/rask-psykisk-helsehjelp/

(Både UiO og SiO er super lavterskel)

Tips og triks

Noen tips fra Trondheim kommune om psykisk og psykososial helse:
https://www.trondheim.kommune.no/aktuelt/nyheter/korona/din-psykiske-og-psykososiale-helse-under-korona-pandemien/

Noen tips fra DN om hjemmekontor:
https://www.dn.no/d2/livsstil/hjemmekontor/koronaviruset/arbeidsliv/10-tips-slik-lykkes-du-med-hjemmekontoret/2-1-771731

Noen tips fra forskning.no om hjemmekontor
https://forskning.no/psykologi/slik-takler-du-dagene-med-hjemmekontor/1655019

Noen av mine favoritt morsomme yt-videoer

Broccoli farm:
https://www.youtube.com/watch?v=l7yMZGA2lzg

Wasabi kid:
https://www.youtube.com/watch?v=VCKg_tbYjgs

Rainbow sponge lady:
https://m.youtube.com/watch?v=V_OVxxIvqVw

The office first aid:
https://m.youtube.com/watch?v=Vmb1tqYqyII

De vanligste feilene på oblig 1

Array vs ArrayList

Array er raskere, men ArrayList har metoder, og man kan endre størrelse.

Så bruke Array hvis du vet antall elementer.

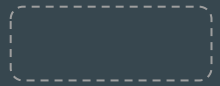
F.eks. i oblig1 så visste vi maks antall noder i nodelistene i array, og de fleste av dere (alle?) implementerte det slik at vi fylte opp et rack helt, før vi la til et nytt rack. Så de fleste racksene ville ha maskAntallNoder, bare det siste racket ville ha litt færre racks.

Datastrukturtegning

Anbefaler å lese hele dette dokumentet i sin helhet:

https://www.uio.no/studier/emner/matnat/ifi/IN1010/v21/notater/omdatastrukt-2021.pdf

Objekter skal ha heltrukket-linje-firkant: 

Alt som er statisk i en klasse skal inni en stiplet-linje-firkant: 

Returntype

Hvorfor er ikke dette lov? Send meg privat melding på chatten

```
1  public String metodeSomReturnererString(boolean returnerString){
2  ··// if (returnerString) er akkuratt det samme som if (returnerString == true)
3  ··if (returnerString){
4  ····return "Heihei"
5  ··}
6  }
```

While-loop vs for-loop vs for-each-loop

Vi bruker while hvis vi _ikke_ vet hvor mange ganger noe skal skje, ellers bruker vi for.
Altså: kan vi bruke for bruker vi for!

Skal vi gå gjennom elementer i en type liste (bokstaver i en string f.eks.) så bruker vi vanlig for hvis vi trenger indeksen til elementet, ellers bruker vi for-each.
Altså: kan vi bruke for-each på liste bruker vi for-each.

```
8  for (Katt katt : katter){¬
9  ··// Denne er mer leslig, men da har vi ikke tilgang til indeks¬
10 }¬
11 for (int indeks = 0; indeks < katter.size(); indeks++){¬
12 ··// Bruker denne hvis vi trenger indeksen¬
13 ··// Aksesserer elementet sånn for arraylist: katter.get(indeks)¬
14 }¬
15 for (int indeks = 0; indeks < katter.length; indeks++){¬
16 ··// Bruker denne hvis vi trenger indeksen¬
17 ··// Aksesserer elementet sånn for array: katter[indeks]¬
18 }¬
```

Ikke putt for mye i loopen din!

Vi skal lese fra fil, men den første linja i filen er litt annerledes enn resten av filen. Vi bruker while loop for å lese fra fil. Men for den første linja må vi altså gjøre noe spesielt, den skal bare kjøre første gangen while-loopen kjører, og det er det første som skjer i while-loopen.
Mange prøvde seg med en if-sjekk ala: if (den linjen vi leser nå er første linje).

```
17          boolean forsteLinje = true;
18          while (fil.hasNextLine()){
19              if (forsteLinje){
20                  maxAntNoderRack = Integer.parseInt(fil.nextLine());
21                  rackArrayList.add(new Rack(maxAntNoderRack));
22                  forstelinje = false;
23              }
```

Ikke putt for mye i loopen din!

Vi skal lese fra fil, men den første linja i filen er litt annerledes enn resten av filen. Vi bruker while loop for å lese fra fil. Men for den første linja må vi altså gjøre noe spesielt, den skal bare kjøre første gangen while-loopen kjører, og det er det første som skjer i while-loopen.
Dette er en mye bedre løsning:

```
15          maxAntNoderRack = Integer.parseInt(fil.nextLine());
16          rackArrayList.add(new Rack(maxAntNoderRack));
17          while (fil.hasNextLine()){
18              // Lese resten av filen
19          }
```

Du trenger ikke putte all koden din i en try/catch!

Det er bare new Scanner som kan gi error, så det er bare den vi trenger å putte inne i try.

Deklarerer variabelen fil utenfor så den er definert utenfor

```
Scanner fil = null;
try{
  fil = new Scanner(new File(filnavn));
} catch (Exception e) {
  System.out.println(e);
  System.exit(1);
}
```

Får mye hjelp/får hjelp for raskt

Obligene er for deres del, det er dere som skal lære å programmere.

De dere får hjelp av kan sannsynligvis å programmere fra før, og ingen tjener på at de gjør obligene deres.

En stor del av programmering er å være stuck med et problem. Sitt litt med problemet ditt, ikke spør med en gang. Lær av feilene dine.

Repetisjon forrige uke

Sist

Lese fra fil

Arv

Tegne klassehierarkitegning

Referanser ved Arv

Casting (hva er lov og hva er ikke lov?)

Lese fra fil

```
class Hund{
  private static Boolean rodlistet = false;
  private String navn;
  private int alder;
  private String favorittgodis = null;

  public Hund(String navn, int alder){
    this.navn = navn;
    this.alder = alder;
  }
  public void settFavorittGodis(String favorittgodis){
    this.favorittgodis = favorittgodis;
  }
  public void skrivUt(){
    System.out.println("Mitt navn er: " + navn + " og jeg er " + alder + " år
    if (favorittgodis != null){
      System.out.println("Min favorittgodis er " + favorittgodis);
    }
    System.out.println("\n");
  }
}
```

```
6
Navn alder favorittgodis
Max 15 eple
Charlie 1 gulrot
Buddy 3 reker
Milo 10 tomat
Oscar 8 ost
Archie 5 Brød
```

```
      Hund[] alleHunder = new Hund[filScanner.nextInt()];
      int nesteLedig = 0;
      filScanner.nextLine();
      filScanner.nextLine();
      while(filScanner.hasNextLine()){
        String[] nesteLinjeArray = filScanner.nextLine().split(" ");
        String navn = nesteLinjeArray[0];
        System.out.println(nesteLinjeArray[1]);
        int alder = Integer.parseInt(nesteLinjeArray[1]);
        String favorittgodis = nesteLinjeArray[2];
        Hund hund = new Hund(navn, alder);
        hund.settFavorittGodis(favorittgodis);
        alleHunder[nesteLedig++] = hund;
      }
```

```
    Hund[] alleHunder = new Hund[filScanner.nextInt()];
    int nesteLedige = 0;
    filScanner.nextLine();
    filScanner.nextLine();
    while(filScanner.hasNextLine()){
      String navn = filScanner.next();
      int alder = filScanner.nextInt();
      String favorittgodis = filScanner.next();
      filScanner.nextLine();
      Hund hund = new Hund(navn, alder);
      hund.settFavorittGodis(favorittgodis);
      //alleHunder[nesteLedige++]; er samme som de to neste linjene
      alleHunder[nesteLedige] = hund;
      nesteLedige++;
    }
```

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
```

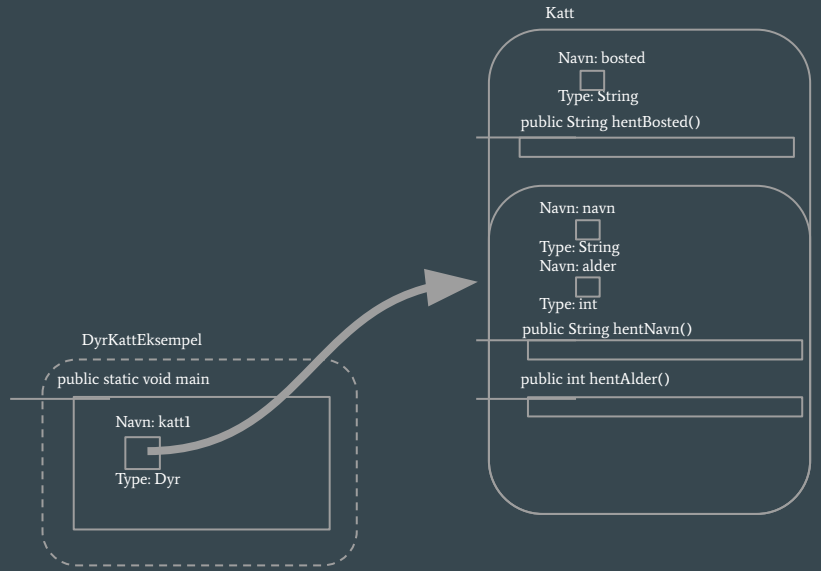
Casting eksempel

```java
class Katt extends Dyr{

    private String bosted;

    public Katt(String navn, int alder, String bosted){
        super(navn, alder);
        this.bosted = bosted;
    }

    public String hentBosted(){
        return bosted;
    }
}
```

```java
abstract class Dyr {

    protected String navn;
    protected int alder;

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

    public String hentNavn(){
        return navn;
    }

    public int hentAlder(){
        return alder;
    }
}
```

```java
class Katt extends Dyr{

    private String bosted;

    public Katt(String navn, int alder, String bosted){
        super(navn, alder);
        this.bosted = bosted;
    }

    public String hentBosted(){
        return bosted;
    }
}
```

```java
abstract class Dyr {

    protected String navn;
    protected int alder;

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

    public String hentNavn(){
        return navn;
    }

    public int hentAlder(){
        return alder;
    }
}
```

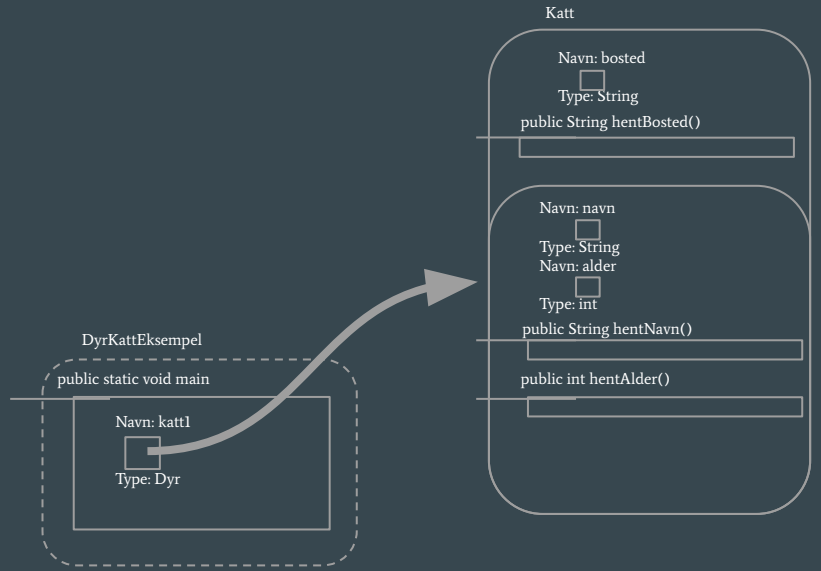



```java
class DyrKattEksempel{

    public static void main(String[] args) {

        Dyr katt1 = new Katt("Pus", 1, "Oslo");

        System.out.println(katt1.hentNavn())
        System.out.println(katt1.hentAlder())
        System.out.println(katt1.hentBosted())
    }
}
```

```java
class Katt extends Dyr{

    private String bosted;

    public Katt(String navn, int alder, String bosted){
        super(navn, alder);
        this.bosted = bosted;
    }

    public String hentBosted(){
        return bosted;
    }
}
```

```java
abstract class Dyr {

    protected String navn;
    protected int alder;

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

    public String hentNavn(){
        return navn;
    }

    public int hentAlder(){
        return alder;
    }
}
```

Siden variabelen er av type dyr må vi ta på oss dyre brillene

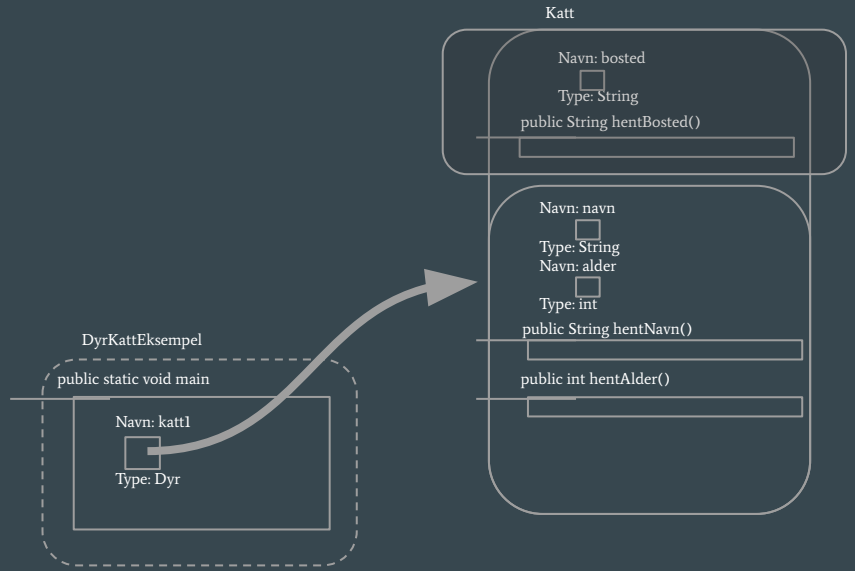



```java
class DyrKattEksempel{

    public static void main(String[] args) {

        Dyr katt1 = new Katt("Pus", 1, "Oslo");

        System.out.println(katt1.hentNavn())
        System.out.println(katt1.hentAlder())
        System.out.println(katt1.hentBosted())
    }
}
```

```java
class Katt extends Dyr{

    private String bosted;

    public Katt(String navn, int alder, String bosted){
        super(navn, alder);
        this.bosted = bosted;
    }

    public String hentBosted(){
        return bosted;
    }
}
```

```java
abstract class Dyr {

    protected String navn;
    protected int alder;

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

    public String hentNavn(){
        return navn;
    }

    public int hentAlder(){
        return alder;
    }
}
```

Katt1 ser nå kun egenskapen til klassen Dyr

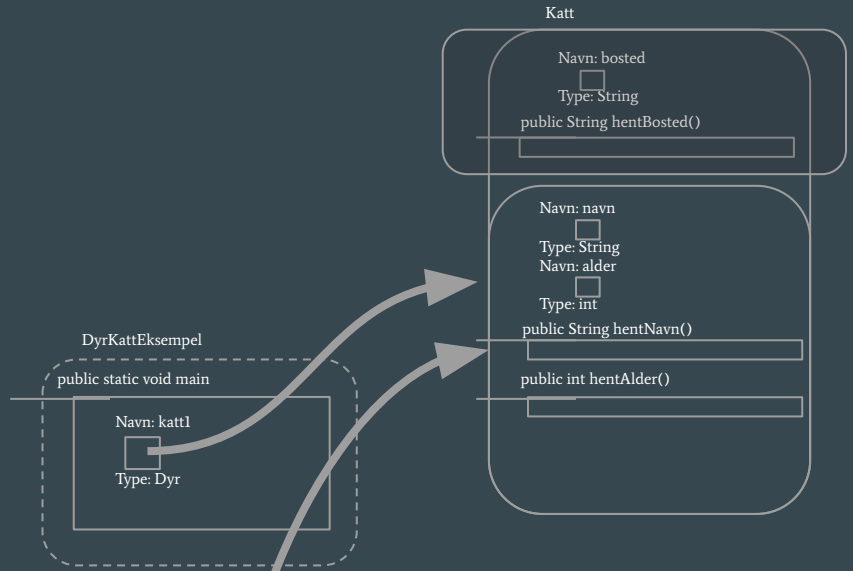



```java
class DyrKattEksempel{

    public static void main(String[] args) {

        Dyr katt1 = new Katt("Pus", 1, "Oslo");

        System.out.println(katt1.hentNavn())
        System.out.println(katt1.hentAlder())
        System.out.println(katt1.hentBosted())
    }
}
```

```java
class Katt extends Dyr{

    private String bosted;

    public Katt(String navn, int alder, String bosted){
        super(navn, alder);
        this.bosted = bosted;
    }

    public String hentBosted(){
        return bosted;
    }
}
```

```java
abstract class Dyr {

    protected String navn;
    protected int alder;

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

    public String hentNavn(){
        return navn;
    }

    public int hentAlder(){
        return alder;
    }
}
```

Output: Pus

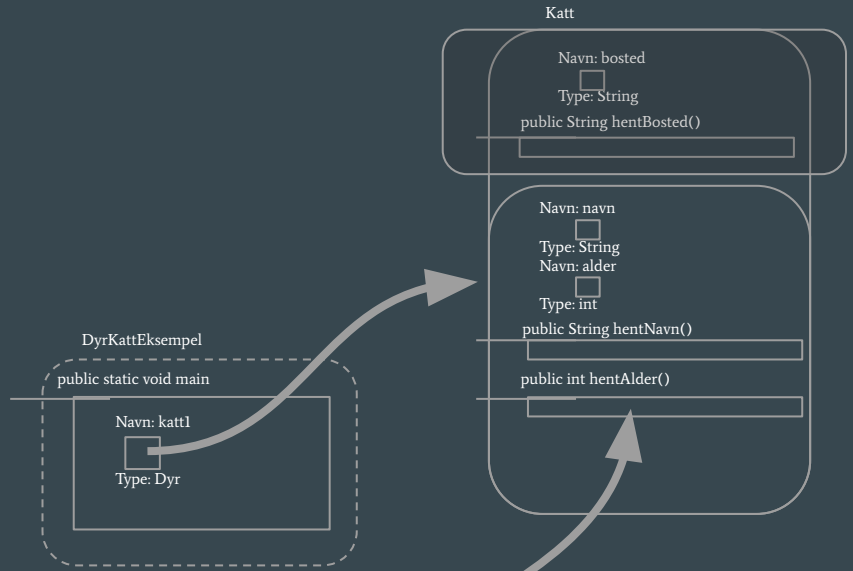



```java
class DyrKattEksempel{

    public static void main(String[] args) {

        Dyr katt1 = new Katt("Pus", 1, "Oslo");

        System.out.println(katt1.hentNavn())
        System.out.println(katt1.hentAlder())
        System.out.println(katt1.hentBosted())
    }
}
```

```java
class Katt extends Dyr{

    private String bosted;

    public Katt(String navn, int alder, String bosted){
        super(navn, alder);
        this.bosted = bosted;
    }

    public String hentBosted(){
        return bosted;
    }
}
```

```java
abstract class Dyr {

    protected String navn;
    protected int alder;

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

    public String hentNavn(){
        return navn;
    }

    public int hentAlder(){
        return alder;
    }
}
```

Output: 1

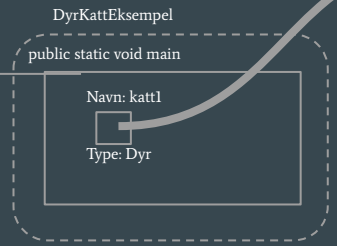


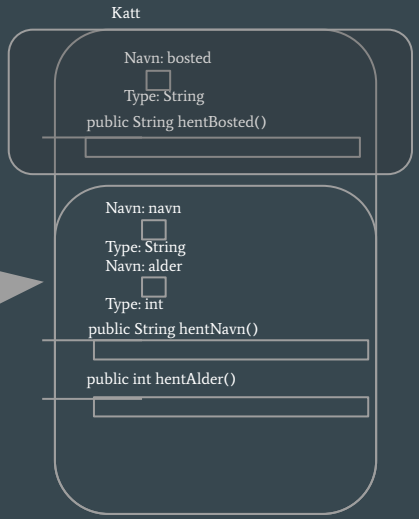

```java
class DyrKattEksempel{

    public static void main(String[] args) {

        Dyr katt1 = new Katt("Pus", 1, "Oslo");

        System.out.println(katt1.hentNavn())
        System.out.println(katt1.hentAlder())
        System.out.println(katt1.hentBosted())
    }
}
```

```java
class Katt extends Dyr{

    private String bosted;

    public Katt(String navn, int alder, String bosted){
        super(navn, alder);
        this.bosted = bosted;
    }

    public String hentBosted(){
        return bosted;
    }
}
```

```java
abstract class Dyr {

    protected String navn;
    protected int alder;

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

    public String hentNavn(){
        return navn;
    }

    public int hentAlder(){
        return alder;
    }
}
```

Feilmelding. Siden katt1 ikke ser egenskapene til Katt objektet og dermed heller ikke metoden hentBosted, derfor må vi caste!





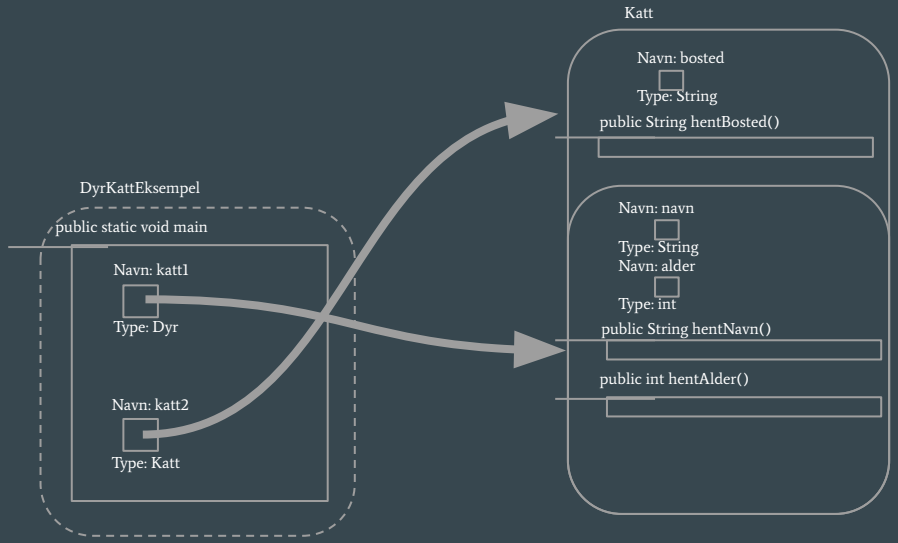

```java
class DyrKattEksempel{

    public static void main(String[] args) {

        Dyr katt1 = new Katt("Pus", 1, "Oslo");

        System.out.println(katt1.hentNavn())
        System.out.println(katt1.hentAlder())
        System.out.println(katt1.hentBosted())
    }
}
```

?? X

```java
class Katt extends Dyr{

    private String bosted;

    public Katt(String navn, int alder, String bosted){
        super(navn, alder);
        this.bosted = bosted;
    }

    public String hentBosted(){
        return bosted;
    }
}
```

```java
abstract class Dyr {

    protected String navn;
    protected int alder;

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

    public String hentNavn(){
        return navn;
    }

    public int hentAlder(){
        return alder;
    }
}
```

```java
class DyrKattEksempel{

    public static void main(String[] args) {

        Dyr katt1 = new Katt("Pus", 1, "Oslo");

        System.out.println(katt1.hentNavn())
        System.out.println(katt1.hentAlder())
        //System.out.println(katt1.hentBosted())

        Katt katt2 = (Katt) katt1;
        System.out.println(katt2.hentBosted());
        System.out.println(katt2.hentNavn());

    }
}
```

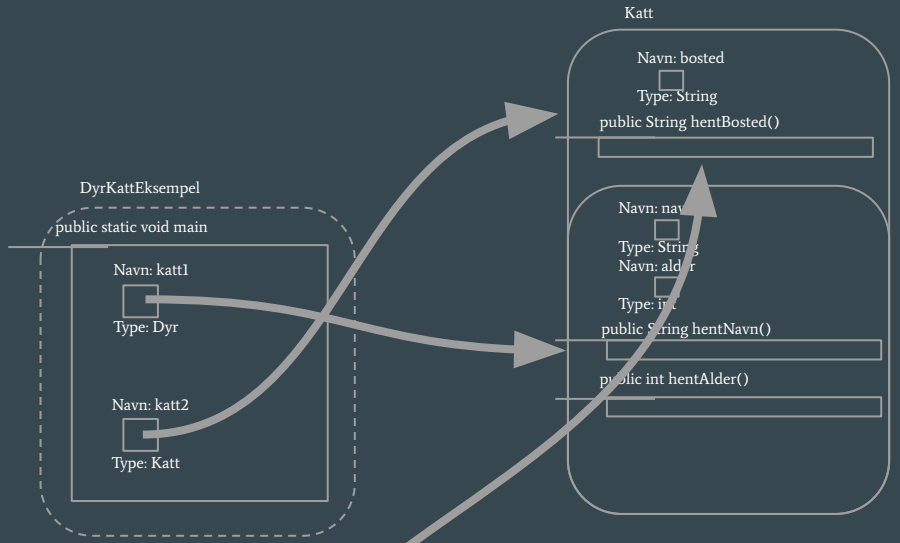


Når vi nå ser på katte objektet men katt2 sine briller får vi også alle egenskapene til en Katt

PS: katt1 kan fortsatt bare se dyreegenskapene

```java
class Katt extends Dyr{

    private String bosted;

    public Katt(String navn, int alder, String bosted){
        super(navn, alder);
        this.bosted = bosted;
    }

    public String hentBosted(){
        return bosted;
    }
}
```

```java
abstract class Dyr {

    protected String navn;
    protected int alder;

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

    public String hentNavn(){
        return navn;
    }

    public int hentAlder(){
        return alder;
    }
}
```



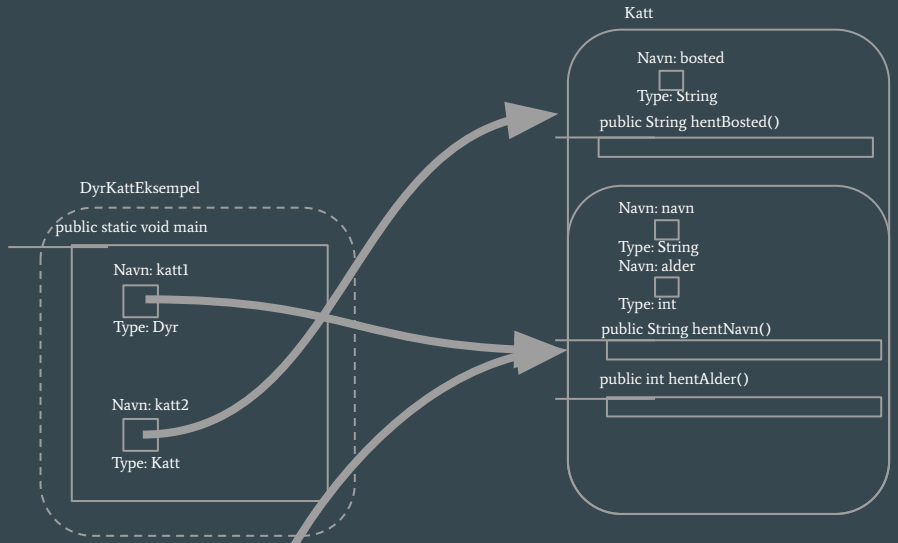

```java
class DyrKattEksempel{

    public static void main(String[] args) {

        Dyr katt1 = new Katt("Pus", 1, "Oslo");

        System.out.println(katt1.hentNavn());
        System.out.println(katt1.hentAlder());
        //System.out.println(katt1.hentBosted())

        Katt katt2 = (Katt) katt1;
        System.out.println(katt2.hentBosted());
        System.out.println(katt2.hentNavn());

    }
}
```

Output: Oslo

```java
class Katt extends Dyr{

    private String bosted;

    public Katt(String navn, int alder, String bosted){
        super(navn, alder);
        this.bosted = bosted;
    }

    public String hentBosted(){
        return bosted;
    }
}
```

```java
abstract class Dyr {

    protected String navn;
    protected int alder;

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

    public String hentNavn(){
        return navn;
    }

    public int hentAlder(){
        return alder;
    }
}
```

```java
class DyrKattEksempel{

    public static void main(String[] args) {

        Dyr katt1 = new Katt("Pus", 1, "Oslo");

        System.out.println(katt1.hentNavn())
        System.out.println(katt1.hentAlder())
        //System.out.println(katt1.hentBosted())

        Katt katt2 = (Katt) katt1;
        System.out.println(katt2.hentBosted());
        System.out.println(katt2.hentNavn());

    }
}
```



Output: Pus

Repetisjon denne uken

I dag

Abstract class

.super

super()

Polymorfi

Ugg hjelp, skjønte ingenting av mentimeter-quizen i forelesningen!

Liveprogrammer og tegning

Abstrakt klasse Dyr

- Navn og alder
- Abstrakt metode lagLyd
- Metode skrivUtInfo

Klasse Katt extends Dyr

- Bosted
- Metode lagLyd (det må vi jo)
- Metode skrivUtInfo

Tre regler fra forelesning

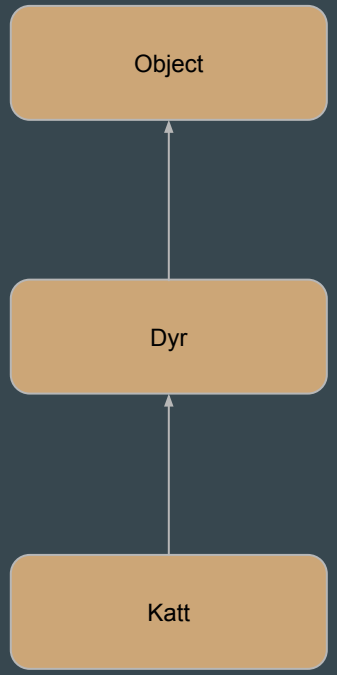
- Et kall på super **må** legges **helt i begynnelsen av** konstruktøren.
- Kaller man ikke super eksplisitt, vil Java **selv legge inn kall på super( )** helt først i konstruktøren når programmet kompileres.
- Hvis en klasse ikke har noen konstruktør, legger Java inn en tom konstruktør med kallet super();

Object, og metodene toString og equals

Object er eller klassers superklasse!

Og den har metoder man kan overskrive, bl.a. toString og equals!

Vi skriver om skrivUtInfo() i klassene våre.



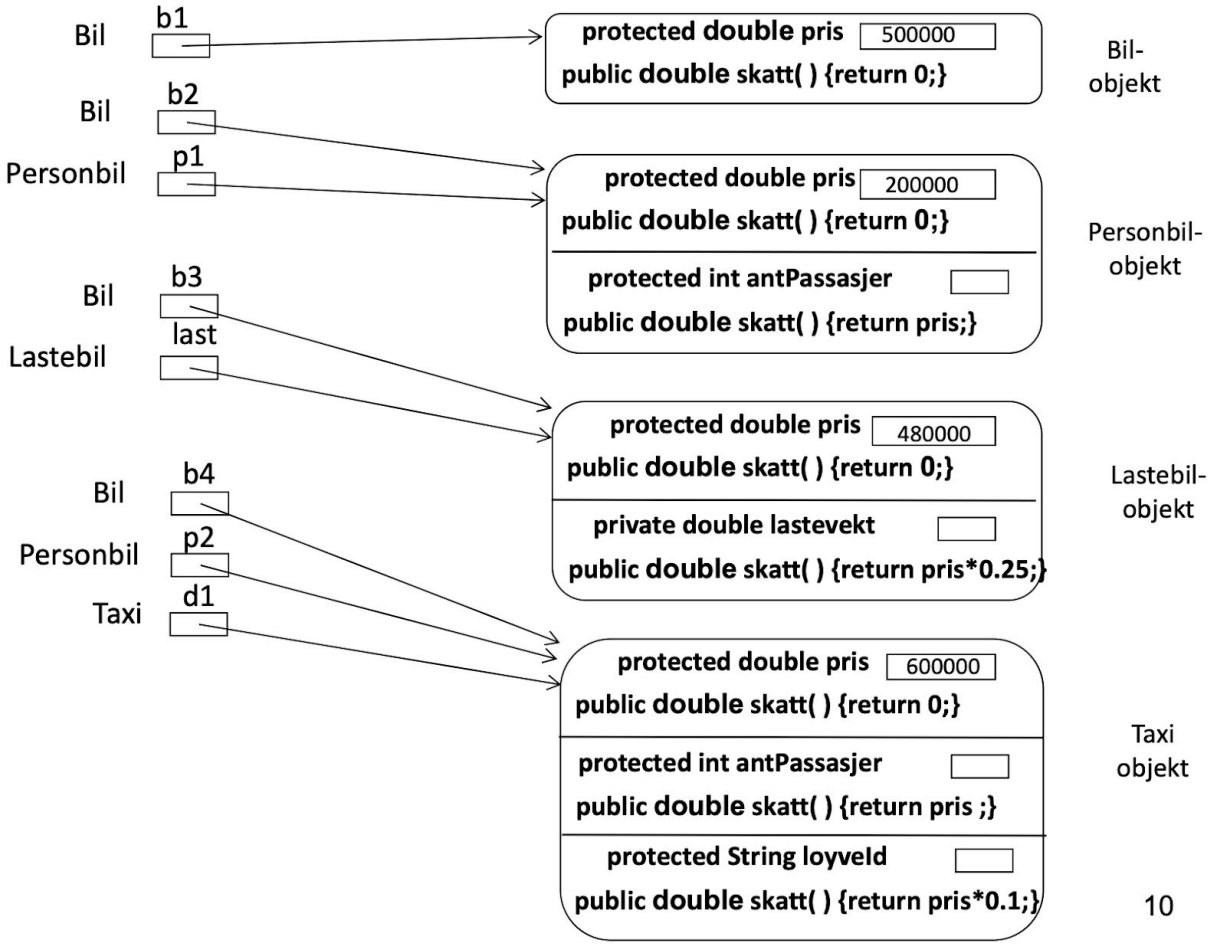
Ugg hjelp, skjønte ingenting av mentimeter-quizen i forelesningen!

Når vi skal avgjøre hvilken virtuell metode som blir kalt så må vi se på typen til objektet! Typen til pekeren er irrelevant!

Vi tegner først klassehierarki

```
class Bil {
    protected double pris;
    protected String regNr;
    public double skatt(){return 0;}
}

class Personbil extends Bil {
    protected int antPass;
    public double skatt( ){return pris;}
    . . .
}

class Lastebil extends Bil {
    protected double lasteVekt;
    public  double skatt(){return pris * 0.25;}
    . . .
}

class Taxi extends Personbil {
    protected int loyveNr;
    public double skatt(){return pris * 0.1; }
    . . .
}
```

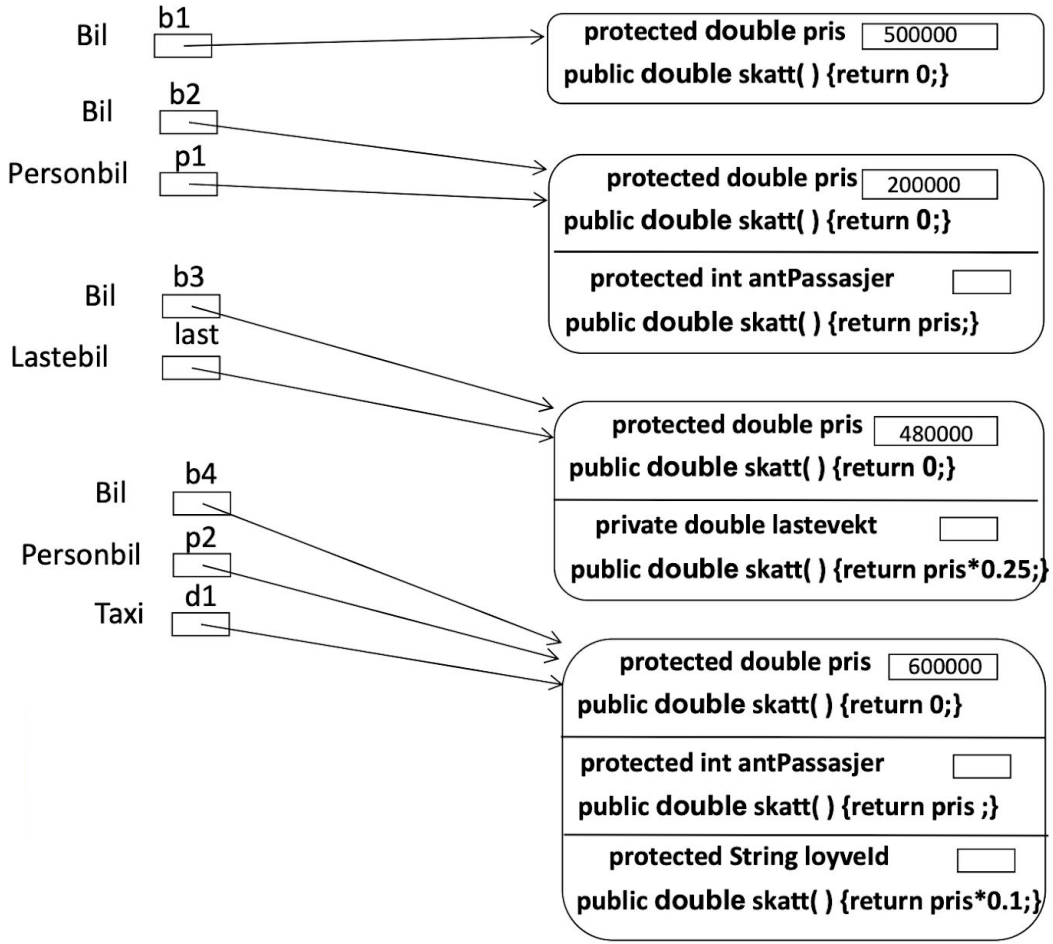
Polymorfi: eksempel



Hva blir:

```
b1.skatt()
b2.skatt()
p1.skatt()
b3.skatt()
last.skatt()
b4.skatt()
p2.skatt()
d1.skatt()
```

Polymorfi: eksempel

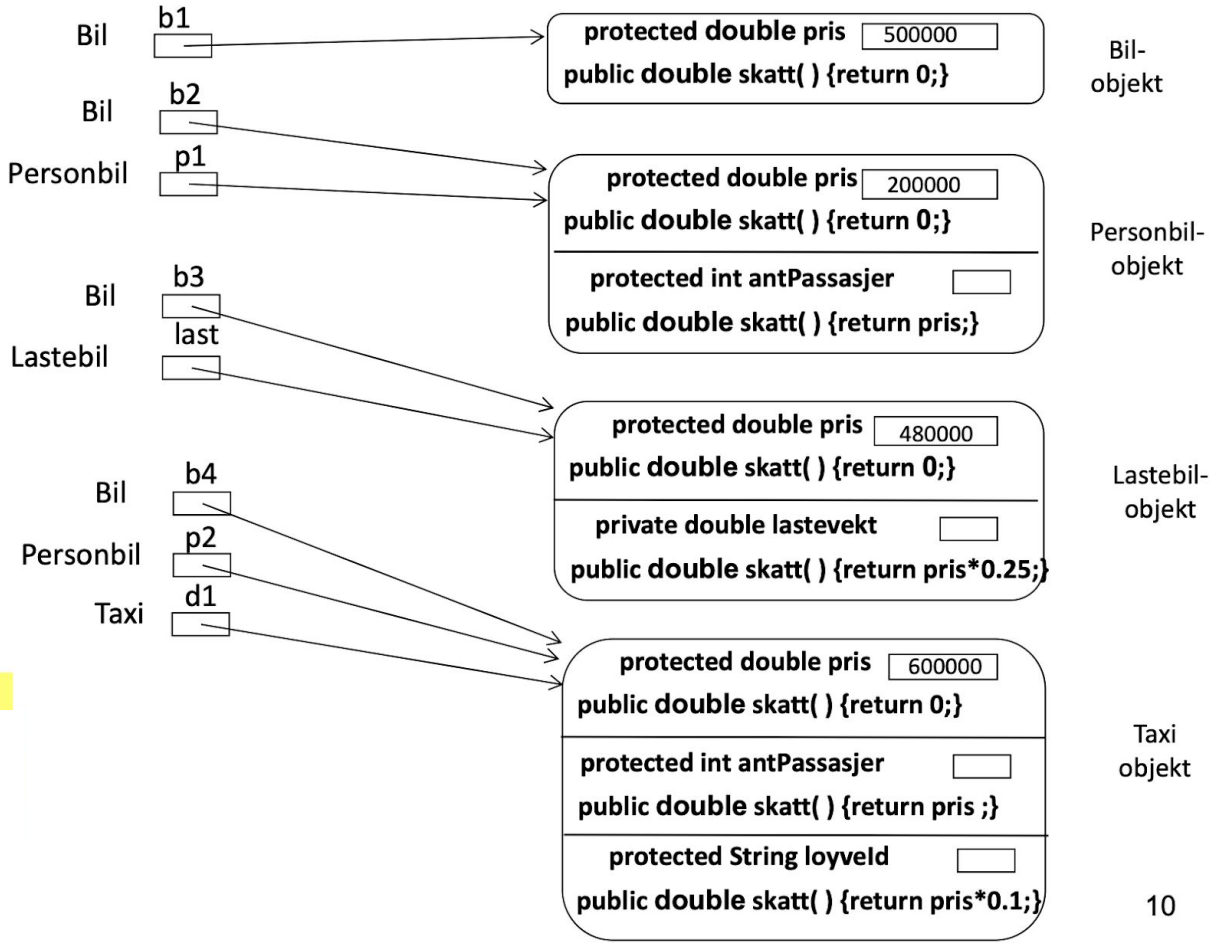


Hva blir:

```
b1.skatt()
b2.skatt()
p1.skatt()
b3.skatt()
last.skatt()
b4.skatt()
p2.skatt()
d1.skatt()
```

0

Polymorfi: eksempel

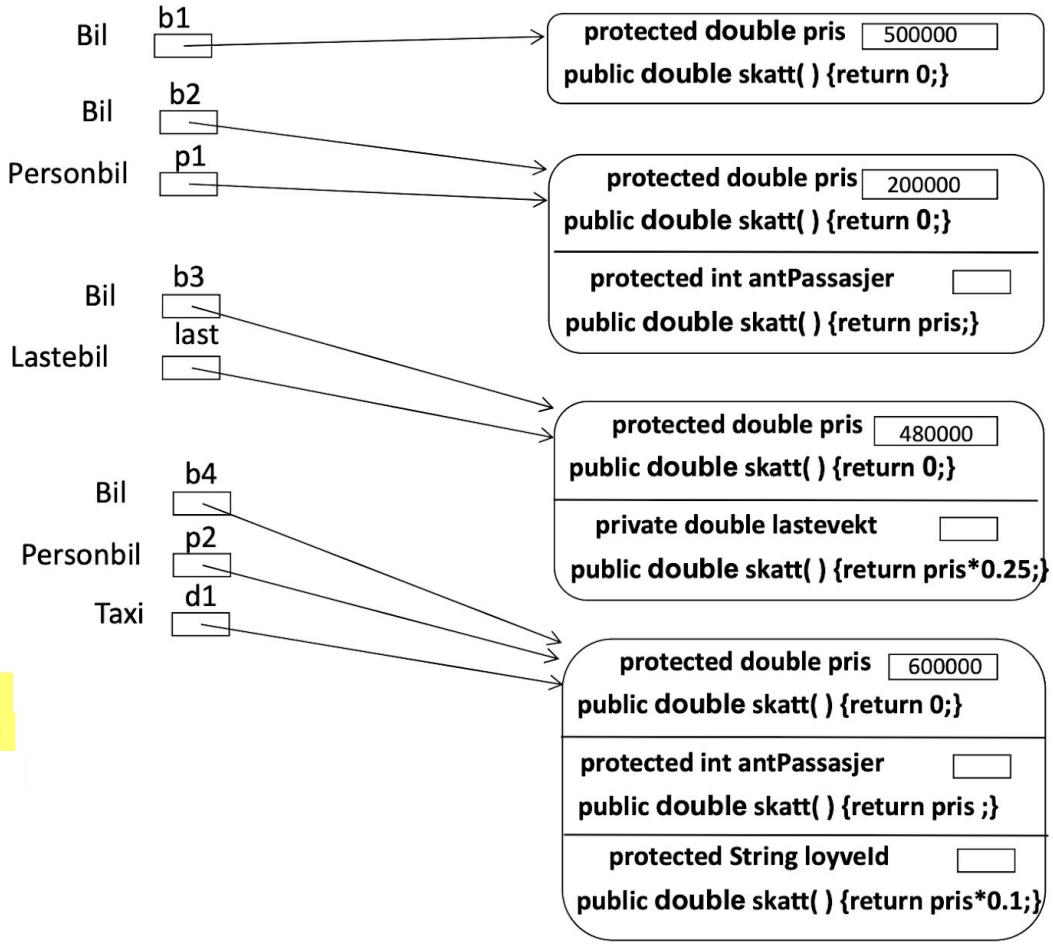


Hva blir:

```
b1.skatt()
b2.skatt()
p1.skatt()
b3.skatt()
last.skatt()
b4.skatt()
p2.skatt()
d1.skatt()
```

```
0
200000
```

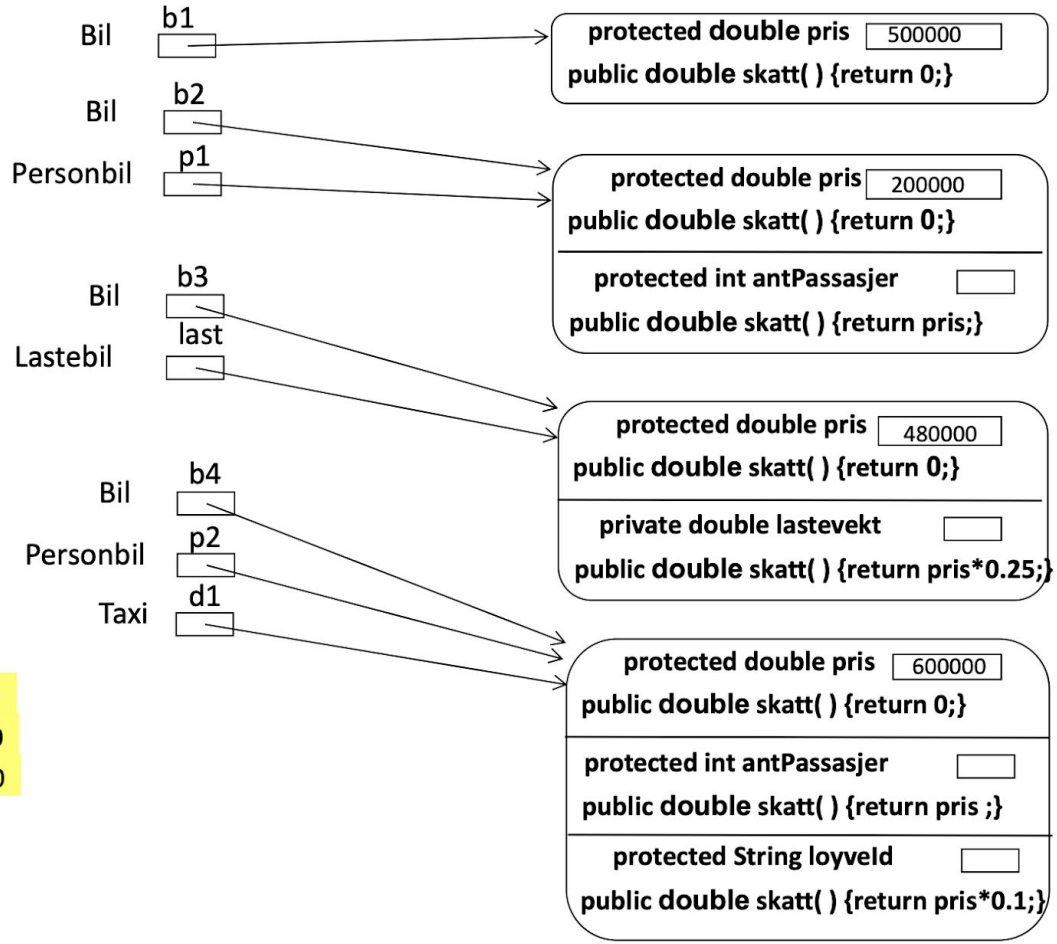
Polymorfi: eksempel



Hva blir:

b1.skatt()	0
b2.skatt()	200000
p1.skatt()	200000
b3.skatt()	
last.skatt()	
b4.skatt()	
p2.skatt()	
d1.skatt()	

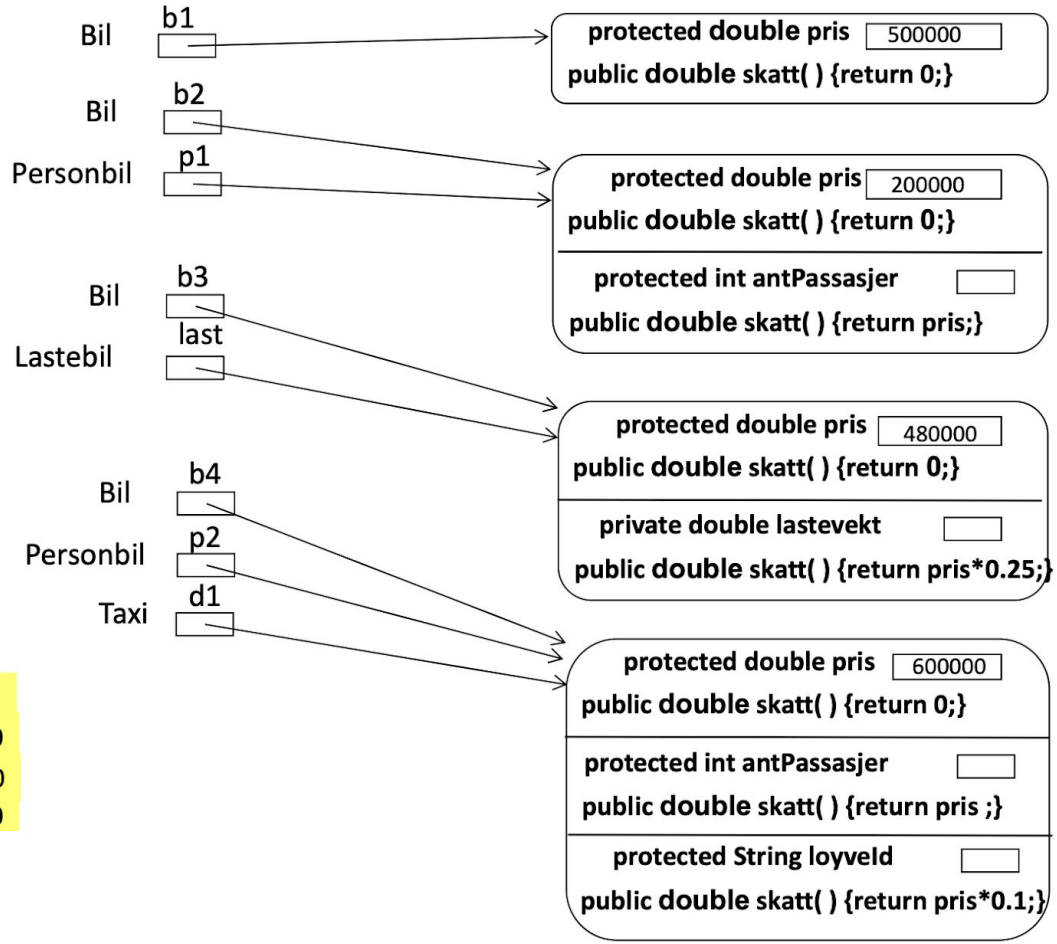
Polymorfi: eksempel



Hva blir:

b1.skatt()	0
b2.skatt()	200000
p1.skatt()	200000
b3.skatt()	120000
last.skatt()	
b4.skatt()	
p2.skatt()	
d1.skatt()	

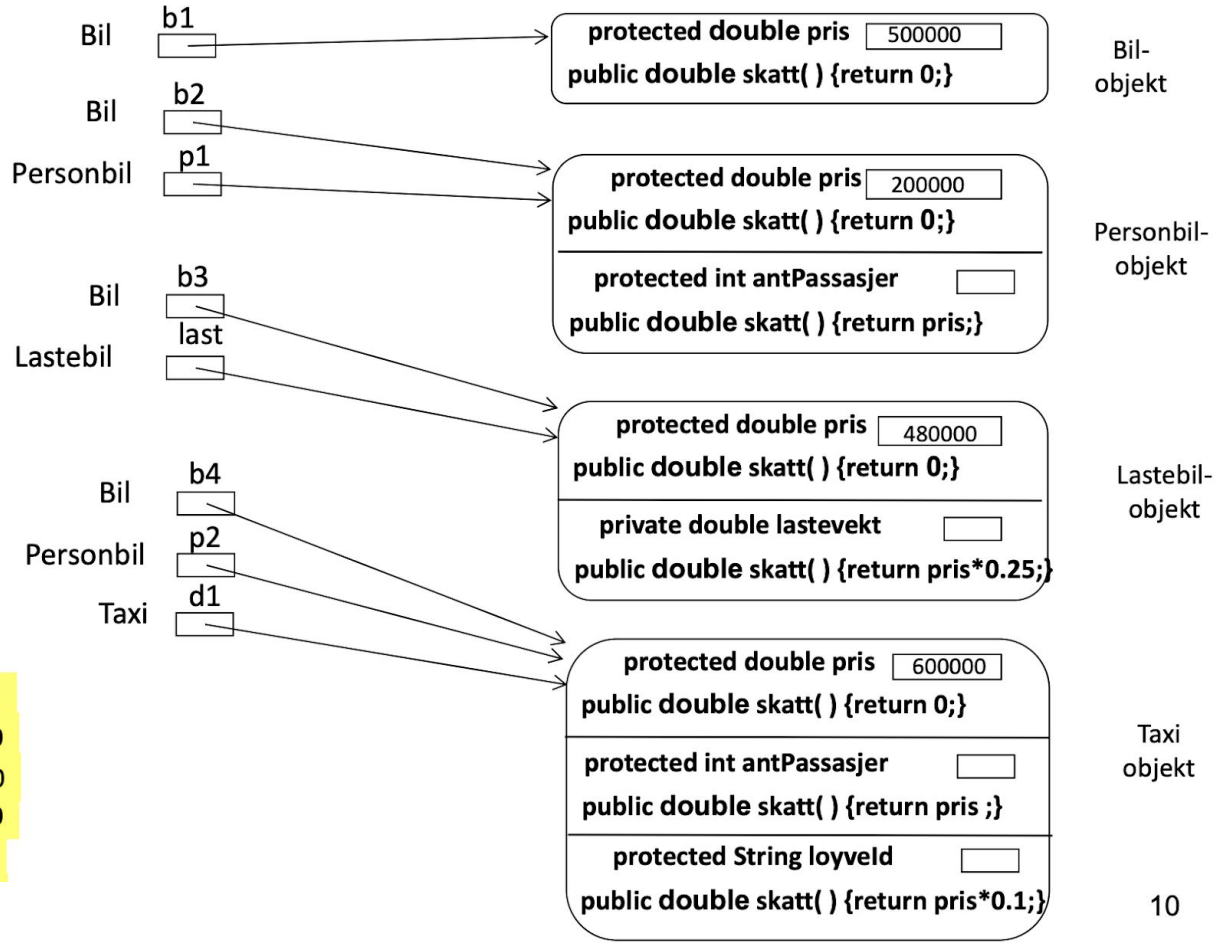
Polymorfi: eksempel



Hva blir:

b1.skatt()	0
b2.skatt()	200000
p1.skatt()	200000
b3.skatt()	120000
last.skatt()	120000
b4.skatt()	
p2.skatt()	
d1.skatt()	

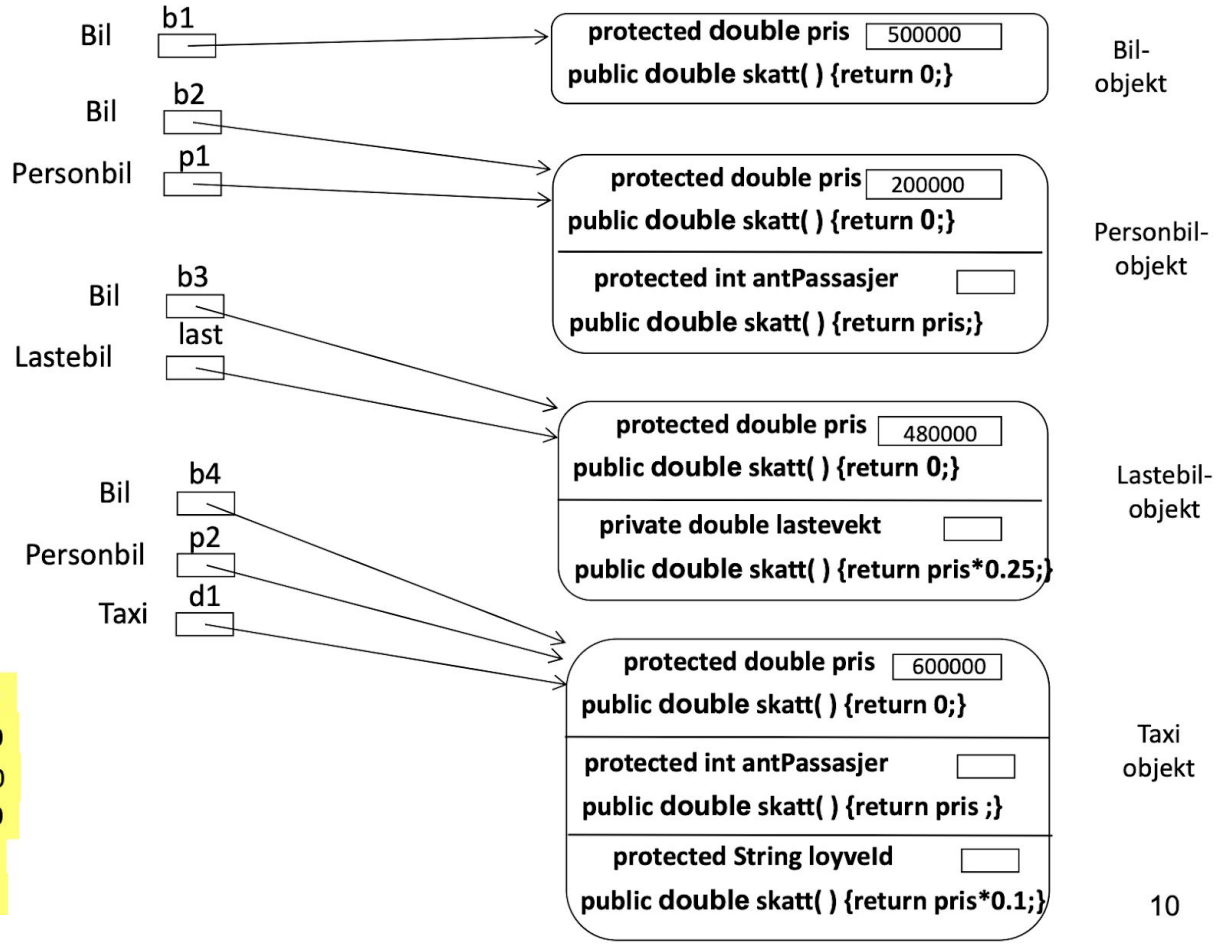
Polymorfi: eksempel



Hva blir:

b1.skatt()	0
b2.skatt()	200000
p1.skatt()	200000
b3.skatt()	120000
last.skatt()	120000
b4.skatt()	60000
p2.skatt()	
d1.skatt()	

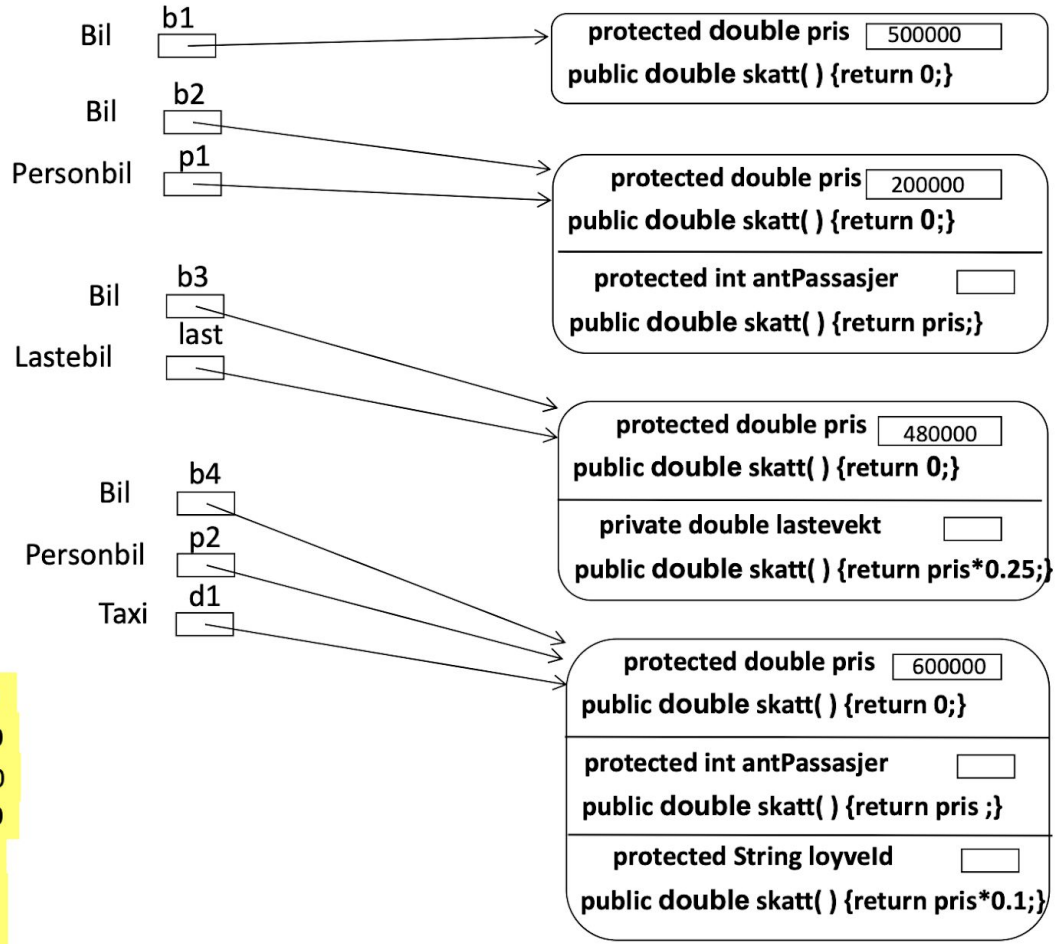
Polymorfi: eksempel



Hva blir:

b1.skatt()	0
b2.skatt()	200000
p1.skatt()	200000
b3.skatt()	120000
last.skatt()	120000
b4.skatt()	60000
p2.skatt()	60000
d1.skatt()	

Polymorfi: eksempel



Hva blir:

b1.skatt()	0
b2.skatt()	200000
p1.skatt()	200000
b3.skatt()	120000
last.skatt()	120000
b4.skatt()	60000
p2.skatt()	60000
d1.skatt()	60000