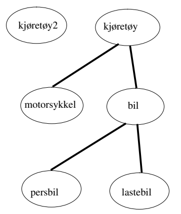
Klasser og subklasser

klassehierarki



en bil **er et** kjøretøy / en lastebil **er en** bil
bil er en **umiddelbar subklasse** av kjoretøy
lastebil er en **subklasse** av kjøretøy
hver klasse er **sin egen subklasse**

klassedeklarasjoner

class kjøretøy;
  begin integer regnr; **end**;

kjøretøy **class** bil;
  begin integer aar; **end**;

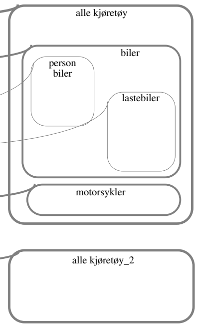
bil **class** persbil;
  begin integer pers; **end**;

bil **class** lastebil;
  begin integer vekt; **end**;

kjøretøy **class** motorsykkel;
  begin text type; **end**;

class kjøretøy_2;
  begin integer regnr; **end**;

objekter av klassen



ref(...) variabler

class kjøretøy;
  begin integer regnr; **end**;

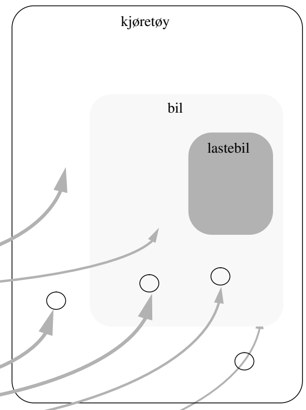
kjøretøy **class** bil;
  begin integer aar; **end**;

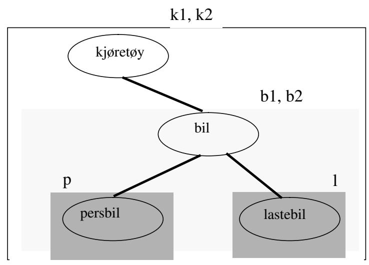
bil **class** persbil;
  begin integer pers; **end**;

bil **class** lastebil;
  begin integer vekt; **end**;

ref(kjøretøy) k1, k2;
ref(bil) b1, b2;
ref(persbil) p;
ref(lastebil) l;

k1 :– **new** kjøretøy ;
k2 :– **new** bil ;
b1 :– **new** bil ;
b2 :– **new** kjøretøy; !! NB ;
b2 :– k1 ; !! NB ;





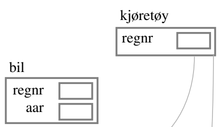
en ref(X) variabel kan peke kun på objektene av *subklasser* til X

en ref(X) variabel er et “vindu”

(tilsv. klassedeklarasjon av X) mot X-objektene

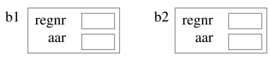
class kjøretøy; **begin integer** regnr; **end;**

kjøretøy **class** bil; **begin integer** aar; **end;**



ref(kjøretøy)   k1, k2, k3;

ref(bil)   b1, b2;



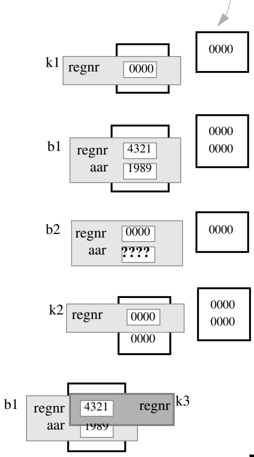
k1 :– new kjøretøy ;

b1 :– new bil ;
b1.regnr:=1234; b1.aar:=1989;

b2 :– new kjøretøy ; !! NB ;

k2 :– new bil ;
k2 . aar := 2000 ; !! NB ;

k3 :– b1 ;



ref_1 :– ref_2

gir ikke feil kun hviss*) ref_1 “vinduet”
ikke ***er*** “større enn” objektet som ref_2 ***peker på***

ref(kjøretøy)   k **;**
ref(bil)   b ;
ref(persbil)   p ;

b :– **new** kjøretøy ;

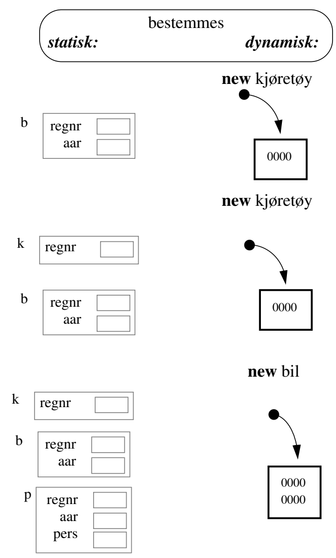
k :– **new** kjøretøy ;

b :– k ; !! Feil ;

k :– **new** bil ;

b :– k ;

p :– k ; !! Feil ;

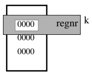


jo mer jeg kan se desto mindre kan jeg peke på
(jo mindre innhold destostørre omfang)

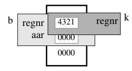
*) ‘hviss’ betyr ‘hvis og bare hvis’

Synlig er kun det som synes gjennom ref-”vinduet” – og litt til . . .

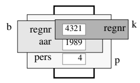
```
ref(kjøretøy)   k ;
ref(bil)   b ;
ref(persbil)   p ;

k :– new persbil;

k . regnr := 4321 ;
k . aar := 1989; !  Feil ;
```



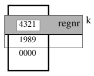

```
b :– k ;
b . aar := 1989
k . pers := 4; !  Feil ;
b . pers := 4; !  Feil ;
```



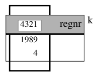

```
p :– k ;
p . pers := 4 ;
```




```
ref(kjøretøy)   k ;

k :– new persbil;

k . regnr := 4321 ;

k qua bil . aar := 1989 ;
```

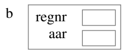



```
k qua persbil . pers := 4 ;
```

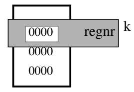


Peker *k* på et objekt av “riktig” type ?
is / in / inspect . . . when

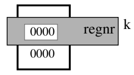
```
ref(kjøretøy)   k ;    ref(personbil)   p ;    ref(motorsykkel)   m ;
ref(bil)   b ;
```



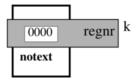

```
k :– new personbil;
b :– k ;
```



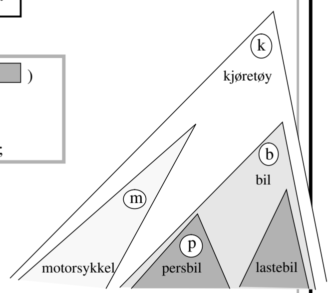

```
k :– new bil;
b :– k ;
```




```
k :– new motorsykkel ;
b:– k;  !! Feil ;
```




```
if k is bil then b :– k ;     (men ikke     )
if k is personbil then p :– k ;
if k is motorsykkel then
   k qua motorsykkel . type :– “Honda” ;
```




```
if k in motorsykkel then      else
if k in bil then      else
if k in personbil then
else <feil> ;
```

```
inspect k
  when motorsykkel do type :– “Honda”;
  when bil do . . .
  otherwise <feil> ;
```


Konklusjonen er . . .

· ***ikke*** at en ref(X) variabler *ikke skal brukes* til å peke på objekter av X sine ekte subklasser,

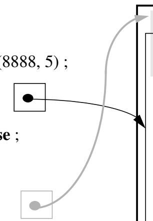
· ***men bare*** at man alltid må tenke på hva som kan være k’s *aktuelle klasse*.

```
element class kjøretøy ; ;
kjøretøy class bil ; ;
bil class personbil ; ;
bil class lastebil ; ;
kjøretøy class motorsykkel ; ;

class Parkering;
 begin                                     ! alle parkerte kjøretøy ;
   ref(Set) lasteb, motors, personb, andre;
   boolean procedure parker(k); ref(kjøretøy) k ;
     if <flere parkeringsplasser> then
       begin
         if k in personbil then personb . add_element_ok(k)
         else if k in lastebil then lasteb . add_element_ok(k)
         else if k in motorsykkel then motors . add_element_ok(k)
         else andre . add_element_ok(k) ;
         parker := true ;
       end else parker := false ;
 end Parkering ;

ref(Parkering)   pVakt ;
ref(Sequence)   kø ;           ! alle mulige kjøretøy som venter for en parkeringsplass;
< kø :– new Sequence >
< pVakt :– new Parkering >
< . . . >
while not kø . is_empty do
 begin
   if pVakt . parker ( kø . head ) then kø . remove_head ;
   < . . . >
 end ;
```


Litt om arve-mekanismen

Objekter av en *subklasse* **arver alt** fra *superklassen* slik at:
attributter / parametre / imperativer deklarert i *superklassen*
kommer før de som er deklarert i *subklassen*.
Deklarasjoner i *subklassen* **overskygger** deklarasjoner fra *superklassen*
(cf. indre blokker) **med mindre** man bruker en **ref**(*superklasse*) variabel.

```
class bil (regnr); integer regnr;
  begin
     integer aar ;
     boolean procedure sjekk_nr; sjekk_nr := true ;
     aar := inint ;
  end;

bil class personbil (pers); integer pers;
  begin
     text eier;
     boolean procedure sjekk_nr;
        sjekk_nr := if regnr > 6000 then false else true ;
     eier :– intext(30) . strip ;
  end;

ref(bil)   b ;
ref(personbil)   p ;

p :– new personbil(8888, 5) ;

! p . sjekk_nr = false ;

b:– p ;

! b. sjekk_nr = true ;
```



```
boolean procedure sjekk_nr; sjekk_nr := true ;
integer regnr, aar ;
regnr := 8888 ;
aar := inint ;
  begin
    integer pers ;
    text eier ;
    boolean procedure sjekk_nr;
      sjekk_nr := if regnr > 6000 then false else true;
    pers := 5 ;
    eier :– intext(30) . strip ;
  end;
```

Oppsummering

klasse deklarasjon **class** X innfører

ny type variabler	ny type objekter
ref(X) k ;	**new** X ;
1. <u>kan peke</u> kun på objekter av typen X	oppretter et objekt av typen X
2. <u>gir adgang</u> til attributtene til objektet den peker på	

subklasse deklarasjon X **class** Y *innfører*

ny (*sub*)type variabler	ny (*sub*)type objekter *et objekt av type Y har også type X*
ref(X) k ;	**new** X ; **new** Y ;
1. <u>kan peke</u> kun på objekter av typen X (*og X's subtyper*)	
2. <u>gir adgang</u> til attributtene til objektet den peker på (*men kun X-attributtene*)	

statisk *vs.* *dynamisk*

*type (av en **ref**(X)-variabel / et objekt) : hva som*

<u>er synlig</u>	*<u>finnes i objektet</u>*
bestemmes av	
variabledeklarasjon (X)	**new** ? ;