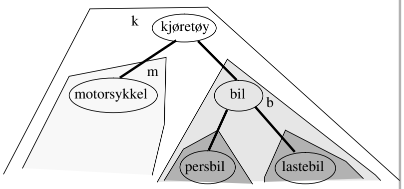
Subklasser

class kjøretøy ;
begin integer regnr; **end**;
kjøretøy **class** motorsykkel ;
begin text type; **end**;
kjøretøy **class** bil ;
begin integer aar; **end**;
bil **class** personbil ;
begin integer pers; **end**;
bil **class** lastebil ;
begin integer vekt; **end**;

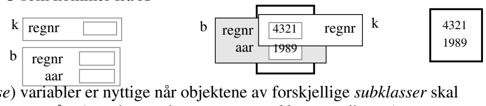


1 definerer et klassehierarki

2 en **ref**(X) variabel kan peke på objekter av alle subklasser av X

3 en **ref**(X) variabel er et "vindu" med struktur tilsvarende dens (**statiske**) klasse X: har vi et objekt O av en *subklasse* av X, vil en slik variabel gi oss adgang bare til de attributtene til O som kommer fra X

ref(kjøretøy) k ;
ref(bil) b ;



4 **ref**(*superklasse*) variabler er nyttige når objektene av forskjellige *subklasser* skal betraktes på samme måte (som bare avhenger av *superklasse*-attributter)

5 dersom en **ref**(X) variabel *k* peker på et objekt av en *ekte subklasse* av X og vi trenger adgang til objektets *subklasse*-attributter, må vi

finne subklassen til objektet	og deretter "åpne" objektet	eller lage et passende vindu
if k **in** motorsykkel **then**	k **qua** motorsykkel.type:–"Honda"	**begin** m:–k; m.type:–"Honda"; **end**
else if k **in** bil **then**	k **qua** bil.aar := 1993	**begin** b:–k; b.aar:=1993; **end**
else	<k er ikke noe skikkelig kjøretøy> ;	<k er ikke noe skikkelig kjøretøy> ;
inspect k		
when motorsykkel **do**	type :– "Honda"	
when bil **do**	aar := 1993	
otherwise	<k er ikke noe skikkelig kjøretøy> ;	

ref(*superklasse*) variabler

gir mulighet til å se bort fra forskjeller mellom de aktuelle objektene (av forskjellige *subklasser*).

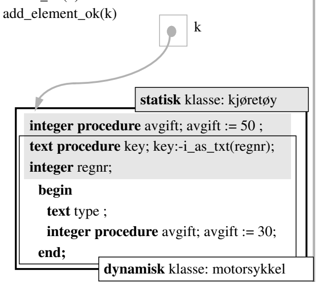
element **class** kjøretøy(regnr) ; **integer** regnr;
begin text procedure key; key:– int_as_text(regnr); **end**;
kjøretøy **class** bil ; ;
bil **class** personbil ; ;
bil **class** lastebil ; ;
kjøretøy **class** motorsykkel ; ;
class Parkering;
begin
ref(Set) ~~parkerte ;~~
lasteb, motors, personb, andre; ! alle parkerte kjøretøy ;
boolean procedure parker(k); **ref**(kjøretøy) k ;
if <flere parkeringsplasser> **then**
begin ~~parkerte . add_element_ok(p) ;~~
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 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 ;

Avgift (1)

element **class** kjøretøy(regnr) ; **integer** regnr;
begin integer procedure avgift; avgift := 50;
text procedure key; key :– int_as_text(regnr); **end**;
kjøretøy **class** motorsykkel ;
begin text type; **integer procedure** avgift; avgift := 30; **end**;
kjøretøy **class** bil ;
begin integer aar; **end**;
bil **class** personbil ;
begin integer pers; **end**;
bil **class** lastebil ;
begin integer vekt; **integer procedure** avgift; avgift := vekt*5; **end**;
class Parkering;
begin
ref(Set) lasteb, motors, personb, andre;
integer kasse;
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** ;
kasse := kasse + $\boxed{\text{k . avgift ;}}$
end else parker := **false ;**
end Parkering ;
ref(Parkering) pVakt ;
ref(Sequence) kø ;
< kø :– new Sequence >
< pVakt :– new Parkering >
< . . . >
while not kø . is_empty **do**
begin
if pVakt . parker (kø . head) **then** kø . remove_head ;
< . . . >
end ;



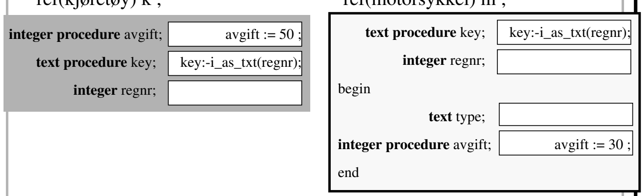
ref(X)-variabel er et vindu med *alle* attributtene til den *statiske klassen* X

element **class** kjøretøy(regnr) ; **integer** regnr;
begin integer procedure avgift; avgift := 50;
text procedure key; key :– int_as_text(regnr); **end**;

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

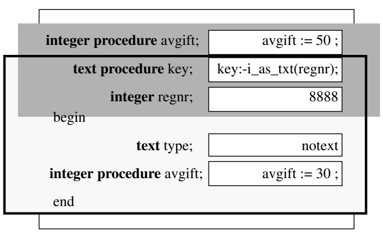
kjøretøy **class** motorsykkel ;
begin text type; **integer procedure** avgift; avgift := 30; **end**;

ref(kjøretøy) k ; ref(motorsykkel) m ;



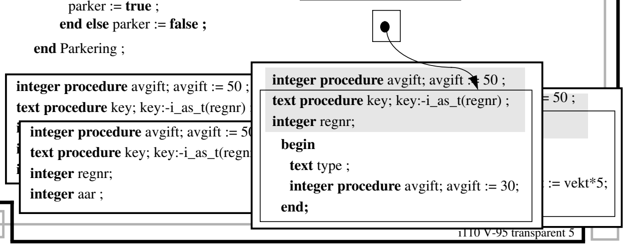
m :- new motorsykkel(8888) ;

k :- m ;



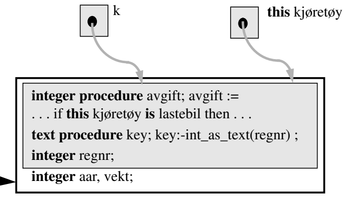
Avgift: *inspect* objekter (2)

```
element class kjøretøy(regnr) ; integer regnr;
   begin  text procedure key; key :– int_as_text(regnr) ;
          integer procedure avgift; avgift := 50;   end;
kjøretøy class motorsykkel ;
   begin  text type;
          integer procedure avgift; avgift := 30; end;
kjøretøy class bil ;
   begin  integer aar; end;
bil class personbil ;
   begin  integer pers; end;
bil class lastebil ;
   begin  integer vekt;
          integer procedure avgift; avgift := vekt*5; end;
class Parkering;
begin
   ref(Set) lasteb, motors, personb, andre;
   integer kasse;
   boolean procedure parker(k); ref(kjøretøy) k ;
     if <flere parkeringsplasser> then
       inspect k
         when personbil do
           begin personb.add_element_ok(k);    kasse := kasse + avgift;   end
         when lastebil do
           begin   lasteb.add_element_ok(k);   kasse := kasse + avgift;   end
         when motorsykkel do
           begin  motors.add_element_ok(k);    kasse := kasse + avgift;   end
         otherwise
           begin  andre . add_element_ok(k);   kasse := kasse + avgift;   end ;
         parker := true ;
       end else parker := false ;
 end Parkering ;
```



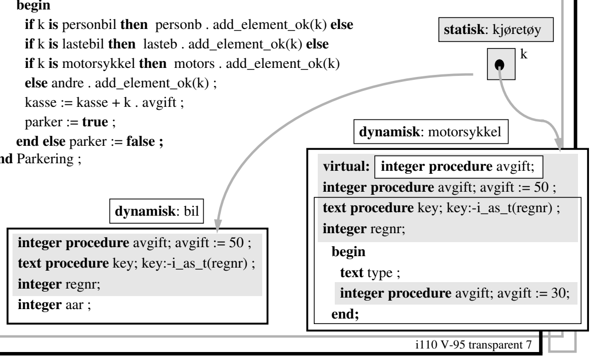
Avgift: en prosedyre i superklassen (3)

```
element class kjøretøy(regnr) ; integer regnr;
   begin  text procedure key; key :– int_as_text(regnr) ;
          integer procedure avgift; avgift :=
             if this kjøretøy is motorsykkel then 30 else
             if this kjøretøy is lastebil then  vekt*5   this kjøretøy qua lastebil . vekt * 5
             else 50;
   end;
kjøretøy class motorsykkel ;
   begin  text type; end;
kjøretøy class bil ;
   begin  integer aar; end;
bil class personbil ;
   begin  integer pers; end;
bil class lastebil ;
   begin  integer vekt; end;
class Parkering;
begin
   ref(Set) lasteb, motors, personb, andre;
   integer kasse;
   boolean procedure parker(k); ref(kjøretøy) k ;
     if <flere parkeringsplasser> then
       begin
         if k is personbil then  personb . add_element_ok(k) else
         if k is lastebil then  lasteb . add_element_ok(k) else
         if k is motorsykkel then  motors . add_element_ok(k)
         else andre . add_element_ok(k) ;
         kasse := kasse + k . avgift ;
         parker := true ;
       end else parker := false ;
 end Parkering ;
```



"*virtual*" : navnet/bruksmåten er felles, men verdien avhenger av den aktuelle (sub)klassen

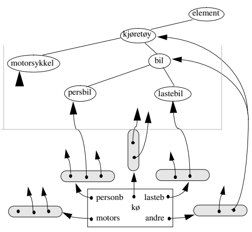
```
element class kjøretøy(regnr) ; integer regnr;
  virtual : integer procedure avgift ;
   begin  text procedure key; key :– int_as_text(regnr) ;
          integer procedure avgift; avgift := 50 ;          ! "default" for alle kjøretøy ;
   end;
kjøretøy class motorsykkel ;
   begin  text type;
          integer procedure avgift; avgift := 30 ;
   end;
kjøretøy class bil ;
   begin  integer aar; end;
bil class personbil ;
   begin  integer pers; end;
bil class lastebil ;
   begin  integer vekt;
          integer procedure avgift; avgift := vekt * 5;
   end;
class Parkering;
begin
   ref(Set) lasteb, motors, personb, andre;
   integer kasse;
   boolean procedure parker(k); ref(kjøretøy) k ;
     if <flere parkeringsplasser> then
       begin
         if k is personbil then  personb . add_element_ok(k) else
         if k is lastebil then  lasteb . add_element_ok(k) else
         if k is motorsykkel then  motors . add_element_ok(k)
         else andre . add_element_ok(k) ;
         kasse := kasse + k . avgift ;
         parker := true ;
       end else parker := false ;
 end Parkering ;
```



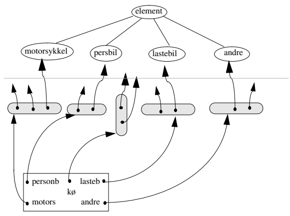
Virtuelle prosedyrer

```
element class kjøretøy(regnr) ; integer regnr;
  virtual : integer procedure avgift ;
          procedure har_parkert is boolean procedure har_parkert(PP); ref(Parkering) PP; ;
  begin  text procedure key; key :– int_as_text(regnr) ;
         integer procedure avgift; avgift := 50 ;
         boolean procedure har_parkert(PP); ref(Parkering) PP;
            har_parkert := <er plass> and then PP . andre . add_element_ok(this kjøretøy) ;
         procedure still_i_kø(PP);  ref(Parkering) PP; PP . kø . append(this kjøreetøy) ;
  end;
kjøretøy class motorsykkel ;
  begin  text type;
         integer procedure avgift; avgift := 30 ;
         boolean procedure har_parkert(PP); ref(Parkering) PP;
            har_parkert := <er plass> and then PP . motors . add_element_ok(this motor-
sykkel) ;
  end;
kjøretøy class bil ;
   begin  integer aar; end;
bil class personbil ;
   begin  integer pers;
         boolean procedure har_parkert(PP); ref(Parkering) PP;
            har_parkert := <er plass> and then PP . personb . add_element_ok(this personbil) ;
  end;
bil class lastebil ;
  begin  integer vekt;
         integer procedure avgift; avgift := vekt * 5;
         boolean procedure har_parkert(PP); ref(Parkering) PP;
            har_parkert := <er plass> and then PP . lasteb . add_element_ok(this lastebil) ;
  end;
class Parkering;
 begin
  ref(Set) lasteb, motors, personb, andre;
  ref(Sequence) kø ;
  integer kasse;
  procedure parker ;
   begin ref(kjøretøy) k ; k :– kø . head ;
   if k =/= none and then k . har_parkert(this Parkering) then
     begin kasse := kasse + k . avgift ;  kø . remove_head; end ;
   end ;
 end Parkering ;
```

Subklasser + ref(*superklasse*) variabler + virtuelle prosedyrer
–> forenklet og mer effektiv programmering



procedure parker (k) ; **ref**(kjøretøy) k ;
  if k =/= none **and then**
    k . har_parkert(**this** Parkering) **then**
  kasse := kasse + k . avgift ;



procedure parker (k) ; **ref**(kjøretøy) k ;
inspect k
  when personbil **do**
    if har_parkert(**this** Parkering) **then**
      kasse := kasse + avgift;
  when lastebil **do**
    if har_parkert(**this** Parkering) **then**
      kasse := kasse + avgift;
  when motorsykkel **do**
    . . .
otherwise . . .

class *element* og *element* **class . . .**

class element ;
  virtual : text procedure key ;
    procedure display;
    . . .
  begin
    text procedure key; key :– **notext** ;
    procedure display; outtext(key) ;
    . . .
  end element ;

element **class** set ;
  begin
    ref(element) head ;   ! data ;
    ref(set) tail ;   ! neste ;

    boolean procedure add_element_ok(el); **ref**(element) el ;
      if head == **none then**
        begin head :– el ; add_element_ok := **true**; **end**
      else if head . key = el . key **then** add_element_ok := **false**
      else begin
        if tail == **none then** tail :– **new** set ;
        tail . add_element_ok ;
      end ;

    ref(element) **procedure** find_element(key); **text** key ;
      find_element :–
        if head == **none then none**
        else if head . key = key **then** head
        else if tail == **none then none**
        else tail.find_element(key) ;
    . . .
  end set ;

virtual : navnet er *“felles”*
innholdet er *“default”*

element **class** ansatt ;
  virtual : procedure ansett;
  begin   ansett_f
    procedure ~~ansett~~ ; bedrift . add_element_ok(**this** ansatt);
    procedure ansett ; ansett_f ;
    . . .
  end ansatt;

ansatt **class** ansatt_i_avd;
  begin
    integer avd;
    procedure ansett ;
    begin   **integer** i;
      ansett_f ;
      i := be_om_int(“I hvilken avdeling? “);
      avdeling(i) . add_element_ok(**this** ansatt_i_avd);
    end;
    . . .
  end ansatt_i_avd;

ref(Ordered_Set) bedrift ;
ref(Ordered_Set) **array** avdeling(1:10) ;

ref(ansatt) A; **ref**(ansatt_i_avd) Aa;

! <Initialiser > ;

A . ansett ;
Aa . ansett ;

! *A :- Aa & A . ansett   hjelper ikke* ;