I går: Parameteroverføring

- **Verdi-overføring: inn-parametre**
  Parametre av enkle typer – **integer, real, character, boolean** – overføres som verdier, med mindre man angir **name** -overføring.
- **Navne-overføring: inn/ut-parametre**
  Navne-overføring må spesifiseres eksplisitt.
- **Referanse-overføring: adresse-overføring**
  Parametre av typene **array, text,** m.m. (sammensatte objekter) – overføres som referanser med mindre man angir **name** eller **value** overføring.

I dag: Rekursive prosedyrer.

Eksempel: Bergne n-fakultet, (n!):
n! = n*(n-1)*(n-2) .... 2*1 for n>0

```
integer procedure fak(n); integer n;
    ! Beregner n-fakultet for n > 0.
    begin
        integer produkt, i;
        produkt := 1;
        for  i := 1 step 1 until n do produkt := produkt*i;
        fak := produkt;
    end;
```

Alternativ løsning:

n! = n*(n-1)! når n>1
= 1 når n=1

```
integer procedure fak(n);integer n;
    if n>1 then fak := n*fak(n-1)
        else fak := 1;
...
integer fakultets_verdi;
fakultets_verdi := fak(4);
...
```

Rekursive prosedyrer = prosedyrer som kaller seg selv!

Binært søk

Binært søk: Finn en verdi i en sortert tabell.

Algoritme-idé: Finn ut om verdien er før eller etter midten av tabellen, og gjenta så søket i den halvparten hvor elementet befinner seg, helt til en bare har et element å teste på.

```
integer procedure bin_sok(verdi,tabell,lav,hoy);
    integer array tabell; integer lav,hoy, verdi;
        ! leter etter verdi i tabellen tall_tab.
        ! returnerer indeks hvis funnet, maxint ellers.
begin
    integer midten;
    if lav=hoy then
        if tabell(lav)=verdi then bin_sok:=lav
          else bin_sok := maxint;
    else
        midten := (lav + hoy)//2;        !avrunder nedover
        if verdi <= tabell(midten) then
            bin_sok :=  bin_sok(verdi,tabell,lav,midten)
        else
            bin_sok :=  bin_sok(verdi,tabell,midten+1,hoy);
end;
```

Generell ide:

Del problemet i mindre deler
og løs det for (hver av) delen(e).

Quicksort: algoritme

Quicksort: sortere en tabel.

Algoritme-idé: Del tabellen i to slik at alle elementene i venstre del er mindre enn en valgt verdi, og alle elementene i høyre del er større.
Gjenta så dette på venste og høyre del rekursivt.

```
algoritme del_tabell(tabell,venstre,høyre,delepunkt);
    velg en deleverdi
    løkke
        finn første element fra venstre som er større en
          deleverdi
        finn første element fra høyre som er mindre enn
          deleverdi
        bytt de to elementene som er funnet
    gjenta til tabellen er delt i to
    returner delepunktets indeks;
slutt
```

```
algoritme quicksort(tabell,venstre,høyre)
    del_tabell(tabell,venstre,høyre,delepunkt);
    quicksort(tabell,venstre,delepunkt-1);
    quicksort(tabell,delepunkt+1,høyre);
slutt
```

Valg av deleverdi?

Best: medianen, (men må da finne den - nesten like vanskelig som sortering)

Som oftest like bra: Ta et vilkårlig element fra tabellen, for eksempel det første i den delen som skal deles.

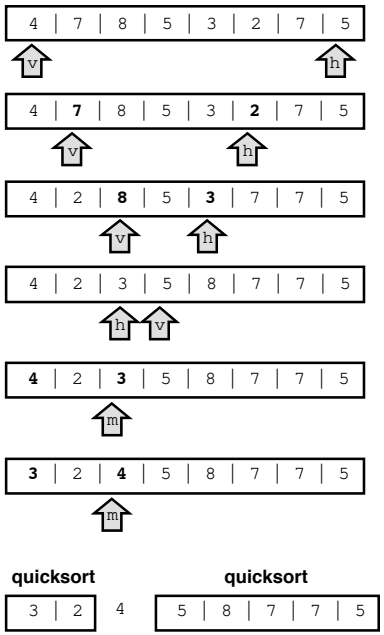
Quicksort

```
procedure bytt_om(x,y);name x,y; integer x,y;
begin
        integer tmp;
        tmp:= x;   x := y;    y:= tmp;
end;
```

```
procedure del_tabell (tab,venstre,hoyre,midten);
        name midten;
        integer array tab; integer lav, hoy;
begin
        integer dele_verdi, v, h;
        dele_verdi := tab(lav);
        v:=venstre; h:= hoyre;
        while v<h do begin
                while (tab(v)<= dele_verdi) and (v<=hoyre) do v:=v+1;
                while (tab(h) > dele_verdi) and (h>=venstre) do h:=h-1;
                if v < h then bytt_om(tab(v),tab(h));
        end;
        midten:= h;
        bytt_om(tab(venstre),tab(midten));
end;
```

```
procedure quicksort(tab,venste,hoyre);
begin
        integer midten;
        if venstre<hoyre then begin
                del_tabell(tab,venstre,hoyre,midten);
                quicksort(tab,venstre,midten-1);
                quicksort(tab,midten+1,hoyre);
        end;
end;
```

Deling av tabell i quicksort



Eksterne prosedyrer

Skrive prosedyrene i egne filer:

Filen: *les_tegn.sim:*

```
character procedure les_tegn(beskjed);
        text beskjed;
begin
        outtext(beskjed);outimage;inimage;
        les_tegn := inchar
end;
```

prog.sim

```
begin
        external character procedure les_tegn;
        character ch;
        ch := lestegn("Et tegn, takk: ");
        ...
end;
```

> sc -c les_tegn
>
> *Kompilerer filen "les_tegn.sim" med eksterne prosedyrer*
>
> sc -C prog les_tegn
>
> *Kompilerer de filene som er nødvendige*

Kan ha flere prosedyrer i samme fil, f.eks i *tegn_bib.sim:*

external character procedure les_tegn = "tegn_bib"