Rekursion

Imperativ Programmering og Datastrukturer

René Rydhof Hansen

October 2007

Skemaændringer

- Forelæsning og øvelser torsdag den 29. november: *flyttes*
- Forelæsning og øvelser fredag den 30. november: *flyttes*
- Forelæsning og øvelser fredag den 7. december: *ny dato*
- Forelæsning og øvelser *mandag* den 10. december: *ny dato*

Mål

- At kunne forklare/definere hvad rekursion er
- At kunne implementere givne rekursive funktioner
- At kunne forklare hvordan rekursion kan bruges som problemløsningsstrategi

For at forstå rekursion... skal man først forstå rekursion

Rekursion

- Rekursion er en problemløsningsstrategi baseret på løsning af delproblemer *af samme slags* som hovedproblemet.
- Der skal være et *basistilfælde* der ikke involverer rekursion for at sikre terminering.

Example

En folder kan indeholde

- Filer
-

Rekursion vs. Iteration

- Løkker kan altid formuleres som rekursion
- Rekursion kan, med noget besvær, formuleres som iteration

For at forstå rekursion... skal man først forstå rekursion

Rekursion

- Rekursion er en problemløsningsstrategi baseret på løsning af delproblemer *af samme slags* som hovedproblemet.
- Der skal være et *basistilfælde* der ikke involverer rekursion for at sikre terminering.

Example

En folder kan indeholde

- Filer
-

Rekursion vs. Iteration

- Løkker kan altid formuleres som rekursion
- Rekursion kan, med noget besvær, formuleres som iteration

For at forstå rekursion... skal man først forstå rekursion

Rekursion

- Rekursion er en problemløsningsstrategi baseret på løsning af delproblemer *af samme slags* som hovedproblemet.
- Der skal være et *basistilfælde* der ikke involverer rekursion for at sikre terminering.

Example

En folder kan indeholde

- Filer
- ... og *andre foldere*

Rekursion vs. Iteration

- Løkker kan altid formuleres som rekursion
- Rekursion kan, med noget besvær, formuleres som iteration

Tilbage til øen...

Example

Fakultetsfunktionen (!):

$$n! = n \cdot (n-1) \cdot \dots \cdot 2 \cdot 1 = \begin{cases} 1 & n = 0 \\ n \cdot (n-1)! & \text{ellers} \end{cases}$$

```
Function Fact(i As Integer)
  if i = 0 then
    Fact = 1
  Else
    Fact = i * Fact(i-1)
  End If
End Function
```

Endnu et eksempel: Fibonacci-tallene

Ballade på Kaninbjerget

Placér et kaninpar i et bur. Under antagelse af, at et kaninpar først kan producere et nyt kaninpar efter to måneder og derefter producerer et kaninpar hver måned, hvor mange kaninpar er der produceret efter et år?

Svar

$$1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144\ldots$$

Definition

$$\mathrm{fib}(n) = \begin{cases} 0 & \text{hvis } n = 0 \\ 1 & \text{hvis } n = 1 \\ \mathrm{fib}(n-1) + \mathrm{fib}(n-2) & \text{ellers} \end{cases}$$

Endnu et eksempel: Fibonacci-tallene

Ballade på Kaninbjerget

Placér et kaninpar i et bur. Under antagelse af, at et kaninpar først kan producere et nyt kaninpar efter to måneder og derefter producerer et kaninpar hver måned, hvor mange kaninpar er der produceret efter et år?

Svar

$$1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144 \ldots$$

Definition

$$\mathrm{fib}(n) = \begin{cases} 0 & \text{hvis } n = 0 \\ 1 & \text{hvis } n = 1 \\ \mathrm{fib}(n-1) + \mathrm{fib}(n-2) & \text{ellers} \end{cases}$$

Endnu et eksempel: Fibonacci-tallene

Ballade på Kaninbjerget

Placér et kaninpar i et bur. Under antagelse af, at et kaninpar først kan producere et nyt kaninpar efter to måneder og derefter producerer et kaninpar hver måned, hvor mange kaninpar er der produceret efter et år?

Svar

$$1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, \textcolor{red}{144} \ldots$$

Definition

$$\mathrm{fib}(n) = \begin{cases} 0 & \text{hvis } n = 0 \\ 1 & \text{hvis } n = 1 \\ \mathrm{fib}(n-1) + \mathrm{fib}(n-2) & \text{ellers} \end{cases}$$

Fibonacci-tallene i Visual BASIC

```
Function Fib(n As Integer)
  If n = 0 Then
    Fib = 0
  Else If n = 1 Then
    Fib = 1
  Else
    Fib = Fib(n-1) + Fib(n-2)
  End If
End Function
```

Rekursion som generel strategi: Hanoi's Tårne

En gang for længe siden...

- Skiverne skal flyttes fra pæl 0 til pæl 2
- En skive kan flyttes til en tom pæl
- En skive kan flyttes over på en *større* skive

```
  x
 xxx
xxxxx
  0     1     2
```

Rekursiv Løsning

- Flyt øverste 2 skiver fra 0 til 1 via 2
- Flyt sidste skive direkte fra 0 til 2
- Flyt øverste 2 skiver fra 1 til 2 via 0

Rekursion som generel strategi: Hanoi's Tårne

En gang for længe siden...

- Skiverne skal flyttes fra pæl 0 til pæl 2
- En skive kan flyttes til en tom pæl
- En skive kan flyttes over på en *større* skive

```
 xxx
xxxxx         x
  0      1    2
```

Rekursiv Løsning

- Flyt øverste 2 skiver fra 0 til 1 via 2
- Flyt sidste skive direkte fra 0 til 2
- Flyt øverste 2 skiver fra 1 til 2 via 0

Rekursion som generel strategi: Hanoi's Tårne

En gang for længe siden...

- Skiverne skal flyttes fra pæl 0 til pæl 2
- En skive kan flyttes til en tom pæl
- En skive kan flyttes over på en *større* skive

```
xxxxx   xxx     x
  0      1      2
```

Rekursiv Løsning

- Flyt øverste 2 skiver fra 0 til 1 via 2
- Flyt sidste skive direkte fra 0 til 2
- Flyt øverste 2 skiver fra 1 til 2 via 0

Rekursion som generel strategi: Hanoi's Tårne

En gang for længe siden...

- Skiverne skal flyttes fra pæl 0 til pæl 2
- En skive kan flyttes til en tom pæl
- En skive kan flyttes over på en *større* skive

```
             x
  xxxxx     xxx
    0        1        2
```

Rekursiv Løsning

- Flyt øverste 2 skiver fra 0 til 1 via 2
- Flyt sidste skive direkte fra 0 til 2
- Flyt øverste 2 skiver fra 1 til 2 via 0

Rekursion som generel strategi: Hanoi's Tårne

En gang for længe siden...

- Skiverne skal flyttes fra pæl 0 til pæl 2
- En skive kan flyttes til en tom pæl
- En skive kan flyttes over på en *større* skive

```
          x
         xxx    xxxxx
   0      1       2
```

Rekursiv Løsning

- Flyt øverste 2 skiver fra 0 til 1 via 2
- Flyt sidste skive direkte fra 0 til 2
- Flyt øverste 2 skiver fra 1 til 2 via 0

Rekursion som generel strategi: Hanoi's Tårne

En gang for længe siden...

- Skiverne skal flyttes fra pæl 0 til pæl 2
- En skive kan flyttes til en tom pæl
- En skive kan flyttes over på en *større* skive

x	xxx	xxxxx
0	1	2

Rekursiv Løsning

- Flyt øverste 2 skiver fra 0 til 1 via 2
- Flyt sidste skive direkte fra 0 til 2
- Flyt øverste 2 skiver fra 1 til 2 via 0

Rekursion som generel strategi: Hanoi's Tårne

En gang for længe siden...

- Skiverne skal flyttes fra pæl 0 til pæl 2
- En skive kan flyttes til en tom pæl
- En skive kan flyttes over på en *større* skive

```
                  xxx
  x              xxxxx
  0       1        2
```

Rekursiv Løsning

- Flyt øverste 2 skiver fra 0 til 1 via 2
- Flyt sidste skive direkte fra 0 til 2
- Flyt øverste 2 skiver fra 1 til 2 via 0

Rekursion som generel strategi: Hanoi's Tårne

En gang for længe siden...

- Skiverne skal flyttes fra pæl 0 til pæl 2
- En skive kan flyttes til en tom pæl
- En skive kan flyttes over på en *større* skive

```
                  x
                 xxx
                xxxxx
  0       1       2
```

Rekursiv Løsning

- Flyt øverste 2 skiver fra 0 til 1 via 2
- Flyt sidste skive direkte fra 0 til 2
- Flyt øverste 2 skiver fra 1 til 2 via 0

Rekursion som generel strategi: Hanoi's Tårne

En gang for længe siden...

- Skiverne skal flyttes fra pæl 0 til pæl 2
- En skive kan flyttes til en tom pæl
- En skive kan flyttes over på en *større* skive

```
                 x
                xxx
               xxxxx
  0      1       2
```

Rekursiv Løsning

- Flyt øverste 2 skiver fra 0 til 1 via 2
- Flyt sidste skive direkte fra 0 til 2
- Flyt øverste 2 skiver fra 1 til 2 via 0

Næste uge

Opsamling og Repetition

- Opsamling på programmeringsdel
- Indledning til datastrukturer
- Repetitionsemner (sendes til `rrh@cs.aau.dk`)