NTNU

Kunnskap for en bedre verden

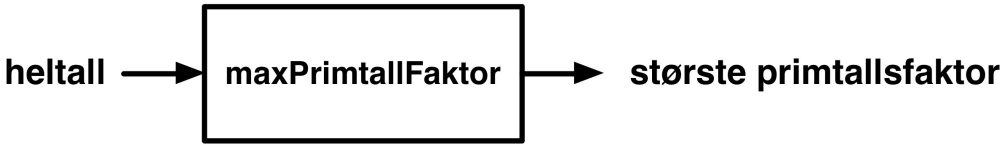
TDT4110 Informasjonsteknologi grunnkurs:

Uke 40:
Gjør ferdig problemløsning (faktorisering)
Vektorisering

Amanuensis Terje Rydland
Kontor: ITV-021 i IT-bygget vest (Gløshaugen)
Epost: terjery@idi.ntnu.no
Tlf: 735 91845

Største primtallsfaktor i tall

- Alle positive heltall kan faktoriseres i primtallsfaktorer
  - 6 = 3 * 2
  - 99 = 11 * 3 * 3
- Vi skal lage en funksjon maxPrimtallFaktor som finner den største primtallsfaktoren til et heltalltall
- Ide: Fjerner de mindre faktorene til vi står igjen med en faktor, som er den største



NTNU

3

Pseudokode

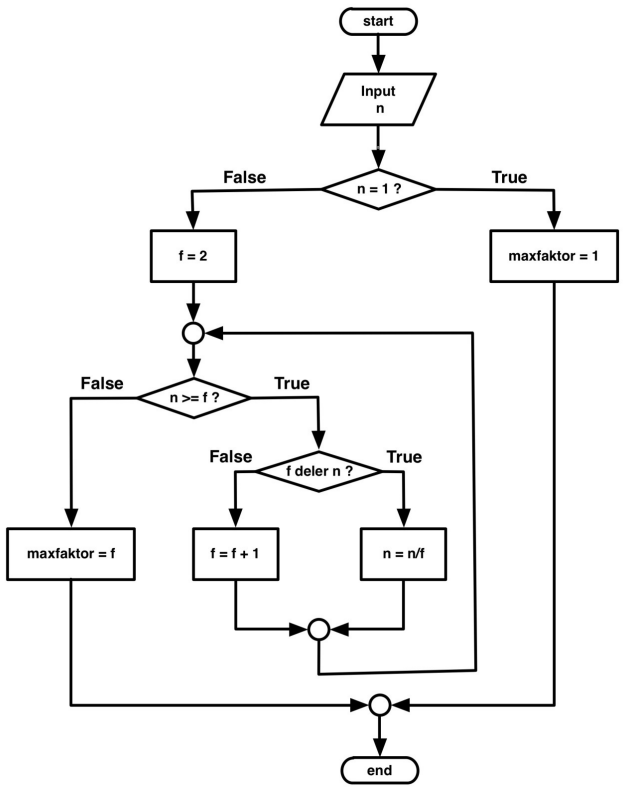
Input: n

hvis n = 1
  maxfaktor = 1
ellers
  faktor = 2
  gjenta så lenge n >= faktor
    hvis faktor deler n
      % fjerner faktor
      n = n/faktor
    ellers
      % må prøve neste tall
      faktor = faktor + 1
    slutthvis
  sluttgjenta
  maxfaktor = faktor

NTNU

4

Flytskjema



NTNU

maxPrimtallFaktor.m

```
function f = maxPrimtallFaktor(n)
% Finner det storste primtallet som er en faktor i n

if n < 1
    % avslutter med feilmelding
    error('Feil i maxPrimtallFaktor: n < 1!')
end %if

if n == 1
    f = 1;
else
    f = 2;
    while n >= f
        if mod(n, f) == 0
            % f er faktor
            n = n/f;
        else
            % f er ikke en faktor (lengre)
            f = f + 1;
        end %if
    end % while
end % if

end % function
```

```
>> maxPrimtallFaktor(99)

ans =

    11

>> maxPrimtallFaktor(333333)

ans =

    37

>> maxPrimtallFaktor(1)

ans =

     1

>>
```


Programkjøringer

```
function f = maxPrimtallFaktor(n)
% Finner det storste primtallet som er en faktor i n

if n < 1
    % avslutter med feilmelding
    error('Feil i maxPrimtallFaktor: n < 1!')
end %if

if n == 1
    f = 1;
else
    f = 2;
    while n >= f
        if mod(n, f) == 0
            % f er faktor
            n = n/f;
        else
            % f er ikke en faktor (lengre)
            f = f + 1;
        end %if
    end % while
end % if

end % function
```

maxPrimtallFaktor(99)

n settes lik 99
f settes lik 2 (de to if-klausulene feiler)

99 >= 2 så vi går inn i while-sløyfa
mod(99,2) er ikke lik 0 så vi hopper til else
f økes med en og blir 3

så opp til while og sjekk
99 >= 3 så vi fortsetter i while-sløyfa
mod(99,3) er lik 0, så vi utfører if
n blir da 99/3, altså 33

så opp til while og sjekk
33 >= 3 så vi fortsetter i while-sløyfa
mod(33,3) er lik 0, så vi utfører if
n blir da 33/3, altså 11

så opp til while og sjekk
11 >= 3 så vi fortsetter i while-sløyfa
mod(11,3) er ikke lik 0, så vi hopper til else
f økes med 1 og blir 4

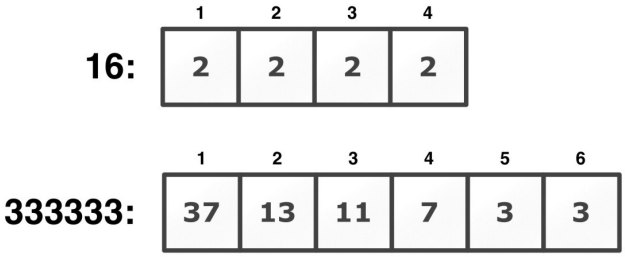
så opp til while og sjekk
11 >= 4 så vi fortsetter i while-sløyfa
mod(11,4) er ikke lik 0, så vi hopper til else
f økes med 1 og blir 5

osv…

Primtallsfaktorisering

- 33 = 11 * 3
- 16 = 2 * 2 * 2 * 2

- Kan bruke maxPrimtallFaktor som byggekloss siden den finnes den største primtallfaktoren
  - Utfør denne flere ganger til det bare står et primtall igjen
  - Lagre de enkelte faktorene i en vektor




```
function v = primtallFaktorisering(n)
% finner primtallsfaktorene i n

   nesteFaktorNr = 1;

if n == 1
    v(nesteFaktorNr) = 1;
else

   restAvN = n;

   while restAvN > 1
       % tar vare paa den storste (gjenvarende faktoren)
       v(nesteFaktorNr) = maxPrimtallFaktor(restAvN);

       % oppdaterer det som staar igjen av N
       restAvN = restAvN/v(nesteFaktorNr);

       % oppdaterer nr for neste faktor
       nesteFaktorNr = nesteFaktorNr + 1;

   end % while

end % if

end %function
```

primtallFaktorisering(99)

n settes lik 99
nesteFaktorNr settes lik 1
n ~=1 så if-klausulen feiler

restAvN settes lik 99

99 > 1 så vi går inn i while-sløyfa
v(1) settes lik 11 (maxPrimtallFaktor(99))
restAvN settes lik 9 (99/11)
nesteFaktorNr settes lik 2

så opp til while og sjekk
9 >= 1 så vi fortsetter i while-sløyfa
v(2) settes lik 3 (maxPrimtallFaktor(9))
restAvN settes lik 3 (9/3)
nesteFaktorNr settes lik 3

så opp til while og sjekk
3 >= 1 så vi fortsetter i while-sløyfa
v(3) settes lik 3 (maxPrimtallFaktor(3))
restAvN settes lik 1 (3/3)
nesteFaktorNr settes lik 4

så opp til while og sjekk
1 = 1 så while-sløyfa er slutt

v er nå lik [11 3 3]

9

Eksempelkjøringer

```
>> primtallFaktorisering(1)

ans =

     1

>> primtallFaktorisering(34)

ans =

    17     2

>> primtallFaktorisering(333333)

ans =

    37    13    11     7     3     3

>> primtallFaktorisering(1000000)

ans =

     5     5     5     5     5     5     2     2     2     2     2     2

>>
```

NTNU

10

Læringsmål og pensum

- Læringsmål
  - Vektorisering av kode
    - Enkel og effektiv kode i Matlab
    - Utnytte mulighetene i Matlab
      - Bruk av innebygde operatorer på tabeller
      - Pre-allokering av tabell-variabler
- Pensum
  - Matlab-boka kapittel 2 og 5.4

NTNU

Vektorisering

- Skrive eller skrive om kode slik at den blir enklere og/eller mer effektiv
  - Bruke tabelloperasjoner
    - + - .* .^ ./ …
    - <, >, …
  - Funksjoner med tabeller som input
- Prøver å unngå brukerskrevne løkker
- Pre-allokering av vektorer og matriser
  - Billigere minnehåndtering

vektorisering_0.m

```
clear
clc

a = 1:10
b = 10:-1:1

c = 2*a

for i = 1:10
    d(i) = 2*a(i);
end

d

e = a .* b

for j = 1:10
    f(j) = a(j)*b(j);
end

f
```

```
a =
     1     2     3     4     5     6     7     8     9    10
b =
    10     9     8     7     6     5     4     3     2     1
c =
     2     4     6     8    10    12    14    16    18    20
d =
     2     4     6     8    10    12    14    16    18    20
e =
    10    18    24    28    30    30    28    24    18    10
f =
    10    18    24    28    30    30    28    24    18    10
>>
```

Relasjonsoperatorene

- Fungerer elementvis på hele vektorer/tabeller
- Gir en logisk vektor/tabell som resultat

```
>> m = randi(10, 3, 10)
m =
     2    10     5     7     6     3     7     1    10    10
     6     7     1     9     7     4    10     7     5    10
     4     2     9     6     7     4     1     3     6    10
>> mindreEnn5 = m<5
mindreEnn5 =
     1     0     0     0     0     1     0     1     0     0
     0     0     1     0     0     1     0     0     0     0
     1     1     0     0     0     1     1     1     0     0
>> antall = sum(mindreEnn5)
antall =
     2     1     1     0     0     3     1     2     0     0
>> antall = sum(sum(mindreEnn5))
antall =
    10
>>
```


Noen innebygde funksjoner

- sum
  - sum(<vektor>) -> summen av elementene
  - sum(<matrise>) -> kolonnesummer
- cumsum
  - Kumulative summer
- max (min)
  - max(<vektor>) -> største element
  - max(<matrise>) -> max element i hver kolonne
- find(<betingelse>)
  - I vektor: Indeksene
  - I matrise: Lineær indeks (1. kolonne, 2. kolonne, …)

Eksempler

vektorisering_4.m

```
clear, clc

v = [1 2 3 4 5];
m = [1:4; 5:8; 9:12];

sum(v), sum(m), sum(sum(m))

max(v), max(m), max(max(m))

find(v>3)
find(m==11)
```

```
ans =
    15
ans =
    15    18    21    24
ans =
    78
ans =
     5
ans =
     9    10    11    12
ans =
    12
ans =
     4     5
ans =
     9
>>
```



```
% trekker 4,5 mill inntekter tilfeldig mellom 1 og 2 mill
% finner antall som tjener mindre enn 100-tusen

clear
clc

% trekker inntektene
m = randi(2000000,4500000,1);

% med for-lokke
tic
ant=0;
for i = 1:4500000
    if m(i) < 100000
        ant = ant + 1;
    end
end
toc
disp(ant)

% med vektor-operasoner
tic
ant = sum(m < 100000);
toc
disp(ant)
```

vektorisering_1.m

```
Elapsed time is 0.184061 seconds.
      225137
Elapsed time is 0.015765 seconds.
      225137
>>
```

17

vektorisering_2.m

```
% summen av de forste N kvadrattallene

clear
clc

N = 100000;

% med for-lokke
tic
ksum = 0;
for i = 1:N
    ksum = ksum + i*i;
end
toc
disp(ksum);

% med vektoroperasjoner
tic
i = 1:N;
ksum = sum(i.*i);
toc
disp(ksum)
```

```
Elapsed time is 0.067772 seconds.
   3.3334e+14
Elapsed time is 0.000649 seconds.
   3.3334e+14
>>
```

NTNU

18

Preallokering av plass

- Allokering vil si å avsette plass til en variabel i minnet
- Tabeller som vokser gradvis er svært lite effektivt
  - Finne ny plass
  - Kopiere gamle verdier til ny plass
- Lønner seg å sette av nødvendig plass (pre-allokering)
  - zeros(n), zeros(n,m)
  - ones(n), ones(n,m)
  - true(n), true(n,m) / false(n), false(n,m) – logiske verdier

NTNU

Kumulative summer

```
clear
clc

N = 100000;

% Uten preallokering av tabell
tic
sum1(1) = 1;
for i = 2:N
    sum1(i) = sum1(i-1) + i;
end

toc

% med preallokering av tabell

tic
sum2 = ones(1,N);
for i = 2:N
    sum2(i) = sum2(i-1) + i;
end

toc

% med innebygd funksjon

tic
sum3 = cumsum(1:N);
toc
```

vektorisering_3.m

```
Elapsed time is 0.020361 seconds.
Elapsed time is 0.000974 seconds.
Elapsed time is 0.000986 seconds.
>>
```