



# Programmering og Problemløsning

## Aflevering 2i

Adam Frederik Ingwersen Linnemann,  
GQR701  
Hold 4

Datalogisk Institut  
Københavns Universitet

September 23, 2016



## 2i.0

Denne delopgave beskæftiger sig med metasyntaksen 'Extended Backus-Naur Form' (EBNF). Vi betragter her en EBNF der indeholder 4 tokens.

1. `charLiteral` kan antage enhver unicode værdi
2. `stringLiteral` kan antage enhver unicode værdi i citationstegn
3. `operator` kan udelukkende antage værdien `'+'`
4. `expression` kan enten antage værdi i form af `stringLiteral` eller `stringLiteral` efterfulgt af `operator` efterfulgt af `expression`

Expressions kan udtrykkes udelukkende af kombinationer af tokenener - som i denne sammenhæng ikke har noget eksplicit indhold:

1.  $\text{Exp1} \leftarrow \text{stringLiteral}$
2.  $\text{Exp2} \leftarrow \text{stringLiteral}, \text{operator}, \text{stringLiteral}, \text{operator}, \text{Exp1}$
3.  $\text{Exp3} \leftarrow \text{stringliteral}, \text{operator}, \text{Exp2}$   
 $= \text{stringliteral}, \text{operator}, \text{stringLiteral}, \text{operator}, \text{stringLiteral}, \text{operator}, \text{stringLiteral}$

Hver enkelt token er defineret ved en eller flere terminaler - som er antager værdier som f.eks. unicode karakterer. Terminaler kan sammensættes inden for reglementet defineret i hvert token. Givet betragtede EBNF har vi, at mulige kombinationer kan være et arbitrært antal adderede strenge. Af mulige kombinationer, er 3 anført i listen nedenfor:

1. `"Chika chika" → "Chika chika"`
2. `"What?"+"..."+"My name is..."+"Who?..."+[Expression 1] → "What?...My name is...Who?..Chika chika"`
3. `"Hi.. My name is.. "+[Expression 2] → "Hi.. My name is.. What?...My name is...Who?..Chika chika"`

EBNF'en er ikke i stand til at arbejde med værdier, som ikke er defineret i et token. Eksempler på situationer, hvor sekvenser er ikke-gyldige er angivet nedenfor:

1. `stringLiteral, stringLiteral, operator, operator`
2. `"Hej"*"Hej"`
3. `amamdmaudjwjadjaw`

## 2i.1

Decimal	Binær	Heximal	Oktal
10	01010	A	12
21	10101	15	25
63	00111111	3f	77
63	00111111	3f	77

Fra Decimal til Binær: Helt generelt, kan vi udlede den binære værdi for et decimal-tal ved at dividere decimal-tallet med 2. Herfra skriver vi fra venstre mod højre; når divisionen resulterer i et heltal, angives 0, ellers 1.  $10/2 \rightarrow 0, 5/2 \rightarrow 1, 2/2 \rightarrow 0, 1/2 \rightarrow 1, 0/2 \rightarrow 0$

Fra Decimal til Hex: Heximal kan antage 16 unikke værdier, fra 0 til 15 - den 10. i rækken er A.

Fra Decimal til Octal: Octal kan antage 8 unikke værdier, fra 0-7. Vi kan dividere decimalen med 8 - herefter kan vi multiplicere med den komma-denoterende rest:  $10/8 \rightarrow 1, rest : 0, 25 \rightarrow 0, 25 \cdot 8 \rightarrow 2$

Fra Binær til Decimal: En binær række følger sumfølgen  $n^2$  fra højre mod venstre, således at den 4. binære værdi i rækken (fra højre mod venstre) er  $2^4 = 16$  hvis 1, 0 hvis 0. Vi kan således udlede værdierne for hver enkelt position og tage summen til følge:  $10101 : 1 \cdot 2^4 + 0 \cdot 2^3 + 1 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0 = 16 + 4 + 1 = 21$

Fra Binær til Hex: Her vil vi udnytte, at vi kender decimalværdien:  $21/16 \rightarrow 1, rest : 0, 3125 \rightarrow 0, 3125 \cdot 16 = 5 \rightarrow 15$

Fra Binær til Octal: Her udnytter vi igen at decimalværdien kendes:  $21/8 \rightarrow 2, rest : 0, 625 \rightarrow 0, 625 \cdot 8 = 5 \rightarrow 25$

Fra Hex til Decimal:  $3 \cdot 16^1 + F(= 15) \cdot 16^0 = 63$

Fra Hex til Binær: Vi udnytter at vi kender decimalværdien:  $63/2 \rightarrow 1, 31/2 \rightarrow 1, 15/2 \rightarrow 1, 7/2 \rightarrow 1, 3/2 \rightarrow 1, 1/2 \rightarrow 1 = 111111$  Vi har nu 6 1-taller - vi bør derfor angive to 0'er foran idet hexadecimal-til-binær antager 4 værdier. Disse resultater er ækvivalente

Fra Hex til Octal: Vi kender decimalværdien:  $63/8 \rightarrow 7, rest : 0, 875 \rightarrow 0, 875 \cdot 8 = 7 \rightarrow 77$

Den sidste række i tabellen: Vi ser, at tallene er ens - og et dobbeltcheck sikrer, at værdierne i række 3 korrekt udregnet. Jeg vil derfor opskrive udregningerne for række 4.

## 2i.2

To mulige løsninger på problemet er angivet nedenfor:

```
///    Mulighed 1)
let streng = "hello world"
printfn "%A" (streng.[0..4] + streng.[6..10])

///    Mulighed 2)
let subStreng1 = streng.[0..4]
```

```
let subStreng2 = streng.[6..10]  
printfn "%A" subStreng1  
printfn "%A" subStreng2
```