Programmering og Problemløsning
Datalogisk Institut, Københavns Universitet
Uge(r)seddel 3 - individuel opgave

Jon Sporring

19. - 25. september.
Afleveringsfrist: onsdag d. 28. september kl. 22:00

I denne periode skal I arbejde individuelt. Regler for individuelle afleveringsopgaver er beskrevet i „Noter, links, software m.m.“→„Generel information om opgaver“. Formålet er at arbejde med:

- bindinger af værdier og funktioner
- muterbare variable
- kodedokumentation
- betingelser og løkker
- rekursive funktioner

Opgaverne for denne uge er delt i øve- og afleveringsopgaver.

Øve-opgaverne er:

3ø.0 Indtast følgende program i en tekstfil, og oversæt og kør programmet

```
let a = 3
let b = 4
let x = 5
printfn "%A * %A + %A = %A" a x b (a * x + b)
```

Listing 1: Expression for a line

Forklar hvad parentesen i kaldet af `printfn` funktionen gør godt for. Tilføj en linje i programmet, som udregner udtrykket $ax + b$ og binder resultatet til `y`, og modificer kaldet til `printfn` så det benytter denne nye binding. Er det stadig nødvendigt at bruge parentes?

3ø.1 Listing 1 benytter F#'s letvægtssyntaks (Lightweight syntax). Omskriv programmet (enten med eller uden `y` bindingen), så det benytter regulær syntaks.

3ø.2 Følgende program,

```
let firstName = "Jon"
let lastName = "Sporring" in let name = firstName + " " +
   lastName;;
printfn "Hello %A!" name;;
```

skulle skrive „Hello Jon Sporring!“ ud på skærmen, men det indeholder desværre fejl og vil ikke oversætte. Ret fejlen(e). Omskriv programmet til en linje (uden brug af semikolonner). Overvej hvor mange forskellige måder, dette program kan skrives på, hvor det stadig gør brug af bindingerne `firstName` `lastName` `name` og `printfn` funktionen.

3ø.3 Tilføj en funktion

```
f : a:int -> b:int -> x:int -> int
```

til Listing 1, hvor `a`, `b` og `x` er argumenter til udtrykket $ax + b$, og modificer kaldet til `printfn` så det benytter funktionen istedet for udtrykket `(a * x + b)`.

3ø.4 Brug funktionen udviklet i Opgave 3ø.3, således at du udskriver værdien af funktionen for $a = 3$, $b = 4$ og $x = 0 \ldots 5$ ved brug af 6 `printfn` kommandoer. Modificer nu dette program vha. af en `for` løkke og kun en `printfn` kommando. Gentag omskrivningen men nu med en `while` løkke.

3ø.5 Lav et program, som udskriver 10-tabellen på skærmen, således at der er 10 søjler og 10 rækker formateret som

	1	2	...	10
1	1	2	...	10
2	2	4	...	20
⋮				
10	10	20	...	100

hvor venstre søjle og første række angiver de tal som er ganget sammen. Du skal benytte to `for` løkker, og feltbredden for alle tallene skal være den samme.

3ø.6 Fakultetsfunktionen kan skrives som,

$$n! = \prod_{i=1}^{n} i = 1 \cdot 2 \cdot \ldots \cdot n \tag{1}$$

(a) Skriv en funktion

```
fac : n:int -> int
```

som benytter en `while` løkke og en lokal variable til at beregne fakultetsfunktionen.

(b) Lav en variant

```
recFac : n:int -> int
```

som benytter rekursion og ingen variable til at beregne fakultetsfunktionen.

(c) Afprøv begge funktioner ved at lave et program, som laver en tabel med 3 kolonner `n`, `fac n` og `recFac n`, og sikr dig at de 2 funktioner regner rigtigt.

(d) Hvad er det største n, som disse funktioner kan beregne fakultetsfunktionen for, og hvad er begrænsningen?

Afleveringsopgaven er:

3i.0 Betragt følgende sum af heltal,

$$\sum_{i=1}^{n} i. \tag{2}$$

Man kan ved induktion vise, at $\sum_{i=1}^{n} i = \frac{n(n+1)}{2}$, $n \geq 0$. Opgaven har følgende delafleveringer:

(a) Skriv en funktion

```
sum : n:int -> int
```

som benytter en lokal variabel `s` og en `while` løkke til at udregne summen $1 + 2 + \cdots + n$.

(b) Lav en funktion

```
recSum : n:int -> int
```

som benytter rekursion og uden brug af variable til at udregne summen $1 + 2 + \cdots + n$. Hint: $\sum_{i=1}^{n} i = n + \sum_{i=1}^{n-1} i$.

(c) Lav en funktion

```
simpleSum : n:int -> int
```

som i stedet benytter formlen $\frac{n(n+1)}{2}$.

(d) Lav et program, som skriver en tabel ud på skærmen med 4 kolonner: `n`, `sum n`, `recSum n` og `simpleSum n`, og verificer at de 3 funktioner kommer til samme resultat.

3i.1 Som en variant af Opgave 3ø.5, skal der arbejdes med funktionen

```
mulTable : n:int -> string
```

som tager 1 argument og returnerer en streng indeholdende de første $1 \leq n \leq 10$ linjer i multiplikationstabellen inklusiv ny-linje tegn, således at hele tabellen kan udskrives med et enkelt `printf "%s"` statement. F.eks. skal kald til `mulTable 3` returnere

```
printf "%s" (mulTab 3);;
        1    2    3    4    5    6    7    8    9   10
   1    1    2    3    4    5    6    7    8    9   10
   2    2    4    6    8   10   12   14   16   18   20
   3    3    6    9   12   15   18   21   24   27   30
```

Listing 2: Example of use and output from `mulTab`

hvor alle indgange i tabellen har samme bredde. Opgaven har følgende delafleveringer

(a) Lav

```
mulTable : n:int -> string
```

så den som lokal værdibinding benytter en og kun en streng, der indholder tabellen for $n = 10$, og benyt streng-indicering til at udtrække dele af tabellen for $n < 10$. Afprøv `mulTable n` for $n = 1, 2, 3, 10$.

(b) Lav

```
loopMulTable : n:int -> string
```

så den benytter en lokal streng-variabel, som bliver opbygget dynamisk vha. 2 `for` løkker og `sprintf`. Afprøv `loopMulTable n` for $n = 1, 2, 3, 10$.

(c) Lav

```
recMulTable : n:int -> string
```

som benytter rekursion og uden brug af variable opbygger strengen. Afprøv `recMulTable n` for $n = 1, 2, 3, 10$.

(d) Lav et program, som benytter sammenligningsoperatoren for strenge `=`, og som skriver en tabel ud på skærmen med 3 kolonner: `n`, og resultatet af sammenligningen af `mulTable n` med hhv. `loopMulTable n` og `recMulTable n` som `true` eller `false`.

(e) Forklar forskellen mellem at benytte `printf "%s"` og `printf "%A"` til at printe resultatet af `multTab`.

Afleveringsopgaven skal afleveres som et antal fsx tekstfiler navngivet efter opgaven, som f.eks. `3i1a.fsx` for delopgave a af første opgave. Tekstfilerne skal kunne oversættes med fsharpc, og resultatet skal kunne køres med mono. Funktioner skal dokumenteres ifølge dokumentationsstandarden, og ud over selve programteksten skal besvarelserne indtastes som kommentarer i de fsx-filer, de hører til. Det hele skal samles i en zip fil og uploades på Absalon.

God fornøjelse.