Introduktion til programmering, ugeseddel 4

Version 1.0

21. september 2014

Den fjerde undervisningsuge handler om *brugerdefinerede datatyper*, *abstraktion* og *kombinatorisk søgning*. Derudover skal vi arbejde videre med lister, brug af listekombinatorer og brug af funktioner som argumenter og returværdier.

Omrokering:

Vi bytter rundt på fredagsforelæsningen og mandagens repetitionstime. Fredagen her i uge 4 vil derfor blive brugt på repetition og begge timer mandag morgen i uge 5 vil dække nyt stof.

En anden ændring er at vi har fordelt tirsdagsøvelserne mellem tirsdag og fredag, så den første time begge dage bruges på at gennemgå øvelser fra ugesedlen og den anden time bruges på at arbejde på afleveringsopgaverne.

1 Plan for ugen

Mandag

Simple indbyggede datatyper og brugerdefinerede datatyper

Pensum: HR: 7.1-7.5

Tirsdag

Abstraktion, brugerdefinerede undtagelser og kombinatorisk søgning

Pensum: IP-2: 9 → HR: 7.6-7.7

Fredag

Repetition af ugens pensum. Se ovenfor.

2 Mandagsopgaver

Emner: listekombinatorer, anonyme funktioner og `option`-datatypen

4M1 Definer en anonym funktion der sætter paranteser om en `string`. Anvend den anonyme funktion på alle elementerne i listen `["a", "b", "c", "d"]`.

4M2 Brug `ListPair.zip` og `map` til at skrive en funktion

```
differences : int list -> int list
```

Sådan at kaldet:

```
differences [x1, x2, x3,...,xn]
```

returnerer en liste der indeholder differencer mellem par af elementer i `xs`.

```
[x2-x1, x3-x2, x4-x3,...,xn-xn-1]
```

4M3 Her er `lookup`-funktionen fra mandags-forelæsningerne i Uge 3:

```
(* lookup : (''a * 'b) list -> ''a -> 'b *)
fun lookup []          key = raise Fail "Not in list"
  | lookup ((a,b)::xs) key =
      if key = a then b
      else lookup xs key
```

Omskriv `lookup` så dens returtype er `'b option`

4M4 HR 7.2

4M5 Brug `foldr` til at skrive funktionen

```
somes : ('a option) list -> 'a list
```

der tager en liste af `option`'s og returnerer listen af alle `SOME`-værdierne.

4M6 Brug `foldl` til at finde den største fællesdivisor for en liste af tal (Benyt at 0 er neutral-element for `gcd`)

4M7 Vi har set at funktioner af to argumenter kan skrives på to måder:

Brug af tupler `'a * 'b -> c`

Sekventialiseret `'a -> 'b -> c`

Skriv funktionerne `curry` og `uncurry`, der konverterer mellem de to:

```
curry   : ('a * 'b -> c) -> ('a -> 'b -> c)
uncurry : ('a -> 'b -> c) -> ('a * 'b -> c)
```

(En anden forklaring af opgaven findes som opgave 9.5 i HR)

3 Tirsdagsopgaver

Emner: Sortering, `order`-datatypen og brugerdefinerede datatyper.

Det forventes, at du inden øvelserne tirsdag har forberedt dig på opgaverne ved at løse så mange som muligt på egen hånd.

4T1 Definer en datatype

```
datatype solution = ...
```

der kan repræsentere de forskellige muligheder når der findes rødder i en andengradsligning: Ingen reelle rødder, én reel rod eller to reelle rødder.

Løs opgave 1G1 sådan at `solve2` får typen

```
solve2 : real * real * real -> solution
```

Afprøv funktionen med samme kald som i 1G2 og 1G3.

4T2 Datatypen `shape` i HR kapitel 7 kan udvides så alle shapes også har en placering i et todimensionelt koordinatsystem:

```
datatype point = Point of real * real
datatype shape = Circle   of point * real
               | Square   of point * real
               | Triangle of point * real * real * real
```

Skriv en funktion `translate (x,y) shape` sådan at koordinaterne for cirklen, kvadratet og trekanten flyttes med vektoren `(x,y)`. F.eks. skal funktionskaldet:

```
translate (-2.0,3.0) (Circle((0.0,1.0)), 5.0)
```

returnere `Circle((-2.0,4.0), 5.0)`.

4T3 De rationelle tal, er de tal vi kan skrive som en brøk af to heltal:

```
datatype rational = Ratio of int * int
```

For eksempel kan brøkken $\frac{1}{3}$ repræsenteres som `Ratio (1,3)`.

Definer funktionen:

```
rationalCompare : rational * rational -> order
```

sådan at kaldet `rationalCompare(a,b)` sammenligner to rationelle tal og returnerer `LESS`, `EQUAL` eller `GREATER` afhængigt af om `a` er mindre, lig eller større end `b`.

Start med at forlænge brøkkerne, så de får samme nævner. Brug `lcm` til at finde det mindste fælles multiplum, der skal bruges til at forlænge med.

```
(* Least common multiple *)
fun lcm (n, m) = (n * m) div gcd(n, m)
```

4T4 Benyt rationalCompare og Listsort.sort til at sortere en liste af rationelle tal.

4 Fredagsopgaver

Emner: Funktioner som værdier

Det forventes, at du inden øvelserne fredag har forberedt dig på opgaverne ved at løse så mange som muligt på egen hånd.

4F1 Funktionen `quicksort` fra IP-2 afsnit 8.5 kan kun sortere lister af `real`'s. Omskriv `quicksort` til en polymorf funktion:

```
qsort : ('a * 'a -> order) -> 'a list -> 'a list
```

Sådan at `qsort` sorterer en liste ved brug af en sammenligningsfunktion, der er givet som parameter (f.eks. `Int.compare` eller `rationalCompare`)

Quicksort fra IP-2, afsnit 8.5 til kopiere ind og arbejde videre på:

```
fun partition (_,[]) = ([],[])
  | partition (pivot : real, x :: xr)
    = let val (xv, xh) = partition (pivot, xr)
      in if x <= pivot then (x :: xv,     xh)
                       else (     xv,x :: xh)
      end
fun quicksort [] = []
  | quicksort (x :: xr)
    = let val (xv, xh) = partition (x, xr)
      in quicksort xv @ x :: quicksort xh end
```

4F2 Definer funktionen:

```
splitWhen : ('a -> bool) -> 'a list -> 'a list * 'a list
```

sådan at et kald `splitWhen p xs` opdeler `xs` i to lister `(ys, zs)` ved det første `x` i `xs` hvor `p(x) = true`. Det skal gælde at `xs = ys @ zs` og `x` skal være det første element i `zs`.

4F3 Benyt `splitWhen` til at definere funktionen:

```
fields : string -> string list
```

sådan at `fields str` opdeler teksten `str` i felter ved hvert komma. F.eks. skal kaldet `fields ",foo,bar,baz,"` returnere værdien `["","foo","bar","baz",""]` og kaldet `fields "test"` returnere værdien `["test"]`.

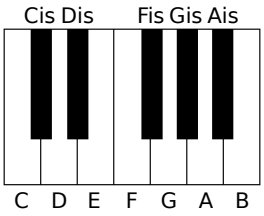
5 Opgavetema: klavernoder

I det vestlige tonesystem benyttes 12 toner:

C, Cis, D, Dis, E, F, Fis, G, Gis, A, Ais og B[1].

Vi kan repræsentere de tolv toner med denne datatype (enumeration):

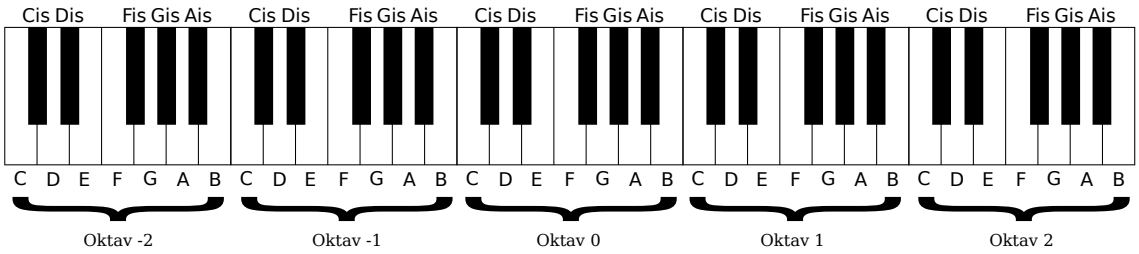
```
datatype pitchclass = C | Cis | D | Dis | E | F | Fis
                    | G | Gis | A | Ais | B
```



Figur 1: En oktav på et klaver

På et klaver gentages disse tolv toner igen og igen i et antal oktaver, hvor de oktaver der ligger til venstre for midten giver dybe toner og de oktaver der ligger til højre for midten giver lyse (eller „høje“) toner.

Tonehøjden kan angives ved et tal der indikerer oktaven, vi vil bruge 0 til at indikere den midterste oktav og positive og negative tal til at angive antallet af oktaver væk fra midteroktaven. Dette er illustreret på følgende figur:



Figur 2: Et klaver med 5 oktaver

Vi kan nu definere:

```
type octave = int
type pitch = pitchclass * octave
```

Den længste node man kan notere er *helnoden*. Alle andre noder og pausers varighed angives *relativt* i forhold til varigheden af en *helnode*. På den måde kan tempoet af en sang justeres ved at angive længden af en helnode og så følger længden af alle andre toner med.

[1]Tonerne har forskellige navne i forskellige skalaer, f.eks. har tonen Fis også navnet Ges, men det skal vi ikke bekymre os om her. Hvis der er musikere blandt de studerende, så kan I se det som om vi bygger et system der kun kan bruges til at skrive melodier i den kromatiske C-skala. Anse det som en forenkling foretaget af pædagogiske grunde

En tone der spiller i en fjerdedel af en helnode har længden 1/4, vi nøjes dog med at repræsentere varigheden som et heltal d, der er nævneren i brøkken $1/d$. Vi repræsenterer altså varigheden af en fjerdedelsnode med heltallet 4 og varigheden af en helnode med heltallet 1.

```
type duration = int
```

En melodi kan nu beskrives ved angivelse af en liste af toner og pauser, hvor der for hver tone er angivet længden tonen skal holdes og en tonehøjde (oktav), og der for hver pause er angivet en længde af pausen:

```
datatype music = Note of duration * pitch
               | Rest of duration

type melody = music list
```

6 Gruppeaflevering

Gruppeafleveringen obligatorisk. Alle delspørgsmål skal besvares. Opgaven afleveres i Absalon. Der afleveres en fil pr. gruppe, men den skal angive alle deltageres fulde navne i kommentarlinjer øverst i filen. Filens navn skal være af formen `4G-`*`initialer`*`.sml`, hvor initialer er erstattet af gruppemedlemmernes initialer. Hvis f.eks. Bill Gates, Linus Torvalds, Steve Jobs og Gabe Logan Newell afleverer en opgave sammen, skal filen hedde `4G-BG-LT-SJ-GLN.sml`. Brug gruppeafleveringsfunktionen i Absalon.

Gruppeopgaven giver op til 2 point, som tæller til de 20 point, der kræves for eksamensdeltagelse. Genaflevering kan hæve pointtallet fra første aflevering med højest 1 point, så sørg for at gøre jeres bedste allerede i første aflevering.

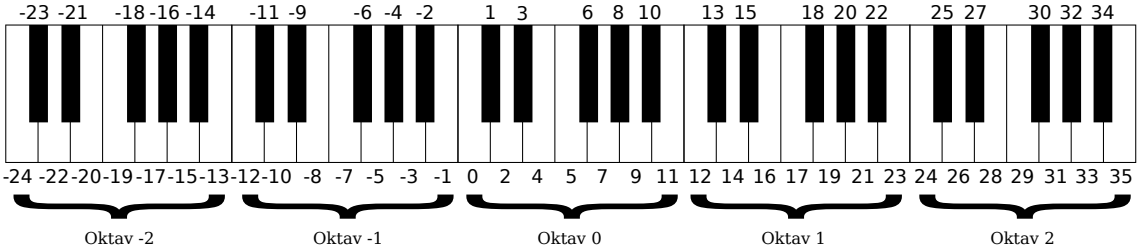
4G1 Skriv en funktion `musicToString : music -> string` der der konverterer en tone eller en pause til følgende format, der både er læsbart for mennesker og computere:

- Pauser printes som „`r`“ („r“ for „rest“) efterfulgt af længden af pausen. En pause af en fjerdedels længde printes `r4`. En pause af en toogtredivtedels længde printes `r32`.
- Toner i midteroktaven (oktav 0) printes som tonensnavn (med lowercase) efterfulgt af længden af tonen. F.eks. printes `Note (4, (Cis, 0))` som `cis4` og `Note (32, (B, 0))` som `b32`.

  Toner i oktaver over eller under midteroktaven bruges et antal kommaer eller apostrofer[2] til at angive oktaven. En apostrof går en oktav op og et komma går en oktav ned. `Note (4, (Fis, 2))` udskrives som `fis''4` og `Note (1, (A, ~3))` som `a,,,1`.

Opdel problemet i passende delproblemer og skjul hjælpefunktionerne i en lokal erklæring.

[2] Apostroftegn er `'`, og ligger på tasten til højre for „Ø“ på et dansk tastatur



Figur 3: Et klaver med 5 oktaver, hvor hver tangent er nummereret

4G2 Skriv en funktion `melodyToString : melody -> string` der printer en hel melodi ved at kalde `musicToString` for alle toner og pauser og indsætter et blanktegn mellem hver. Teksten `"a16 b16 c'4 c'8"` bør returneres ved kaldet:

```
melodyToString [Note (16, (A,0)), Note (16, (B,0)),
                Note (4, (C,1)), Note (8, (C,1))]
```

4G3 Nogle beregninger er nemmere at lave med en anden repræsentation, hvor hver en tone er angivet som et tal der angiver afstanden fra det midterste C på klaveret.

```
type abspitch = int
```

Skriv en funktion `absolutePitch : pitch -> abspitch`, der kombinerer pitchclass og oktav i et enkelt tal. F.eks. skal `absolutePitch (A, ~1)` returnere `~3` og kaldet `absolutePitch (G, 1)` skal returnere `19`. Se nummerering for et 5-oktavers klaver på Figur 3.

4G4 Skriv en funktion `pitch : abspitch -> pitch` der foretager den modsatte konvertering: fra et heltal til et par bestående af tone og oktav.

4G5 En melodi kan flyttes op eller ned et antal tonetrin, ved at flytte alle toner det samme antal skridt i samme retning. Dette kaldes at „transponere“. Skriv funktionen

```
transpose : int -> melody -> melody
```

sådan at `transpose n mel` flytter alle tonerne i melodien `mel` op eller ned med `n` skridt (negativt `n` betyder at tonerne skal transponeres ned). Pauser skal bevares som de er.

For eksempel skal kaldes `transpose 3 [Note (32, (A,-1))]` returnere `[Note (32, (C,0))]` fordi tre skridt over A (oktav -1) ligger C (oktav 0). Kaldet `transpose ~12 mel` transponere melodien `mel` ned med én oktav.

4G6 Skriv en funktion `maxPitch : melody -> pitch`. Der finder den højeste tone i en melodi. Hvis listen er tom eller der kun er pauser i en melodi kastes undtagelsen `Domain`.

4G7 Skriv en funktion `duration : melody -> rational`, der udregner længden af en melodi i antal helnoder. Da vi ikke kan være sikker på at der samlet set er et helt antal af helnoder, skal funktionen returnere varigheden som en rationel værdi (se tirsdagsøvelserne, find evt. den vejledende løsning på Absalon).

7 Individuel aflevering

Den individuelle opgave er obligatorisk. Alle delspørgsmål skal besvares. Opgaven afleveres i Absalon som en fil med navnet `4I-`*`navn`*`.sml`, hvor *navn* er erstatttet med dit navn. Hvis du fx hedder Anders A. And, skal filnavnet være `4I-Anders-A-And.sml`. Skriv også dit fulde navn som en kommentar i starten af filen.

Den individielle opgave giver op til 3 point, som tæller til de 20 point, der kræves for eksamensdeltagelse. Genaflevering kan hæve pointtallet fra første aflevering med højest 1 point, så sørg for at gøre dit bedste allerede i første aflevering.

4I1 Betragt datatypen:

```
datatype ('a, 'b) either = Left of 'a | Right of 'b
```

Skriv funktionen

```
partEither : (('a,'b) either) list -> 'a list * 'b list
```

sådan at `partEither xs` opdeler listen `xs` i to lister, hvor den første har alle `Left`-værdierne og den anden indeholder alle `Right`-værdierne, i samme rækkefølge som de optræder i den oprindelige liste.

4I2 Betragt datatypen:

```
type username = string
type message = int * username * string
datatype status = Online | Away | Offline
datatype chatuser = User of username * status
datatype chatroom = PrivateChat of username * message list
                  | GroupRoom of username list * message list;
```

En bruger i en Chat-klient har et *brugernavn* og en `status`-angivelse (Online, Away, Offline). Der kan ikke være flere brugere med samme brugernavn.

Chat-klienten understøtter både fler-bruger chatrum og private chatrum. For hvert chatrum er en liste af beskeder. En besked indeholder et brugernavn, en besked og en tidsangivelse repræsenteret som antal sekunder siden 1. Januar 1970 kl. 00:00:00.

```
val userlist = [User ("Sus", Offline), User ("Mike", Offline),
                User ("Markus", Online), User ("Emil", Online)]
val room = [PrivateChat ("Sus", [(793980034, "Sus", "Hello!")]),
            GroupRoom (["Mike", "Markus", "Emil"],
                       [(793983657, "Mike", "Hopla!")])]
```

Skriv funktionen `removeOffline : chatuser list * chatroom list -> chatroom list` der fjerner de chatrum hvor alle brugere er `Offline`.

4I3 Skriv en funktion der finder tidspunktet t for den seneste besked i et chatrum og returnerer some t (hvor t er angivet i antal sekunder siden 1. Januar 1970 00:00:00), og returnerer NONE hvis der endnu ikke er skrevet noget i chatrummet.

```
newestMessage : chatroom -> int option
```

4I4 Skriv en funktion der sorterer en liste af chatrum efter hvornår der sidst har været aktivitet, forrest i listen skal alle chatrum uden aktivitet stå og derefter kommer chatrummene med senest aktivitet.

```
sortRooms : chatroom list -> chatroom list
```

8 Ugens nød

I det følgende repræsenteres et punkt som to heltal:

```
type point = int * int
```

Følgende funktion kan bruges til at tegne ensfarvede *figurer* oven på et InstagraML-billede, hvor en *figur* angives som en funktion af typen `point -> bool` der returnerer `true` ved de pixels der skal farves og `false` ved de farver hvor det underliggende billede skal bevares:

```
(* Draw a figure (f) on top of an image (img) in
 * the specified colour. *)
fun drawFigure colour img f =
  let val (w,h,f) = InstagraML.toFunction img
      fun overlay p = if f p then colour else f p
  in InstagraML.fromFunction (w,h, overlay)
  end
```

For eksempel kan en cirkel eller et rektangel tegnes ved at skrive funktioner der afgører om et punkt er inde eller uden for figuren:

```
(* Create a circle from radius and center point
     circle : point * int -> point -> bool
 *)
fun circle ((a,b), r) = fn (x,y) =>
  let fun square x = x*x
  in square (x-a) + square (y-b) < r end

(* Create a rectangle from its top left corner
   and its bottom right corner
     rectangle : point * point -> point -> bool *)
fun rectangle ((x1,y1),(x2,y2)) = fn (p1,p2) =>
  Int.min (x1,x2) <= p1 andalso p1 <= Int.max(x1,x2) andalso
  Int.min (y1,y2) <= p2 andalso p2 <= Int.max(y1,y2)
```

4N1 Skriv en funktion der afgører om et punkt er inde i et polygon eller udenfor, hvor et polygon er repræsenteret som en liste af punkter: `point list`, sådan at den kan bruges til at tegne polygoner med `drawFigure`.

Benyt at du kan finde ud af om et punkt (p_x, p_y) er på venstre side eller højre side af et linjesegment fra (v_x, v_y) til (u_x, u_y) ved at kigge på fortegnet af determinanten:

$$\begin{vmatrix} u_x - v_x & p_x - v_x \\ u_y - v_y & p_x - v_y \end{vmatrix}$$