Programmering og Problemløsning
Datalogisk Institut, Københavns Universitet
Arbejdsseddel 4 - gruppeopgave

Jon Sporring

30. september - 8. oktober.
Afleveringsfrist: lørdag d. 8. oktober kl. 22:00.

På denne uge skal vi tage et nøjere kig på rekursion. Rekursion er et kraftigt værktøj til løsning af en række problemer, blandt andet ved behandling af lister, og kan kædes direkte sammen med induktionsbeviser til at bevise korrekthed af kode, men det kan tager tid at tilegne sig, og derfor bruger vi denne periode på at nærstudere rekursive funktioner.

Denne arbejdsseddels læringsmål er:

- Forklare forskellen på almindelig rekursion og halerekursion
- Løse et problem ved hjælp af rekursion
- Håndkøre rekursive funktioner

Opgaverne er opdelt i øve- og afleveringsopgaver. I denne periode skal I arbejde i grupper med jeres afleveringsopgaver. Regler for gruppe- og individuelle afleveringsopgaver er beskrevet i "'Noter, links, software m.m."'→"'Generel information om opgaver"'.

Øveopgaver (in English)

4ø0 Write a function `length : 'a list -> int` that calculates the length of the argument list using recursion. The function should make use of pattern matching on lists.

4ø1 The greatest common divisor (gcd) between two integers t and n is the largest integer c that divides both t and n with 0 as remainder. Euclid's algorithm[1] finds the greatest common divisor, using recursion, as follows:

$$\gcd(t,0) = t, \tag{1}$$
$$\gcd(t,n) = \gcd(n, t \,\%\, n), \tag{2}$$

Here % is the remainder operator (as in F#).

[1] https://en.wikipedia.org/wiki/Greatest_common_divisor

(a) Implement Euclid's algorithm by writing a recurive function

```
gcd : t:int -> n:int -> int
```

(b) Write down a tracing-by-hand derivation for the calls `gcd 8 2` and `gcd 2 8`.

4ø2 Write a function `lastFloat : float list -> float` that, using recursion, returns the last element of the argument list if the list is non-empty and returns the float value `NaN` if the argument list is empty.

For example, the call `lastFloat [2.1;4.2]` should return the float value `4.2` and the call `lastFloat []` should return the value `NaN`.

Afleveringsopgaver (in English)

In this assignment, you will work with simple continued fractions[2], henceforth just called continued fractions. Continued fractions are lists of integers which represent real numbers. The list is finite for rational numbers and infinite for irrational numbers.

Continued fractions to decimal numbers A continued fraction is written as $x = [q_0; q_1, q_2, \ldots]$ and the corresponding decimal number is found by the following recursive algorithm:

$$x = q_0 + \frac{1}{q_1 + \frac{1}{q_2 + \ldots}}. \tag{3}$$

The series of fractions continues as long as there are elements in the continued fraction.

For example, $[3;4,12,4] = 3.245$, since:

$$\begin{aligned} x &= 3 + \frac{1}{4 + \frac{1}{12 + \frac{1}{4}}} & (4) \\ &= 3 + \frac{1}{4 + \frac{1}{12.25}} & (5) \\ &= 3 + \frac{1}{4.081632653} & (6) \\ &= 3.245. & (7) \end{aligned}$$

Note that all but the first digit must be larger than 0, e.g., $[1;0]$ is an illigal number, and that every rational number has exactly 2 representations $[q_0; q_1, \ldots, q_n] = [q_0; q_1, \ldots, (q_n - 1), 1]$ where the first is called the canonical representation. E.g., $[2;3] = [2;2,1]$, since

$$2 + \frac{1}{3} = 2 + \frac{1}{2 + \frac{1}{1}}. \tag{8}$$

[2] https://en.wikipedia.org/wiki/Continued_fraction

Decimal numbers to continued fractions For a given number x on decimal form, its continued fraction $[q_0; q_1, q_2, \ldots]$ can be found using the following algorithm:

Let $x_0 = x$ and $i \geq 0$, and calculate

$$q_i = \lfloor x_i \rfloor \tag{9}$$
$$r_i = x_i - q_i \tag{10}$$
$$x_{i+1} = 1/r_i \tag{11}$$
$$\tag{12}$$

recursively until $r_i = 0$. The continued fraction is then the sequences of q_i.

For example, if $x = 3.245$ then

i	x_i	$q_i = \lfloor x_i \rfloor$	$r_i = x_i - q_i$	$x_{i+1} = 1/r_i$
0	3.245	3	0.245	4.081632653...
1	4.081632653...	4	0.081632653	12.25
2	12.25	12	0.25	4
3	4	4	0	-

and hence, the continued fraction is in the third column as $3.245 = [3; 4, 12, 4]$.

4g0 Write a recursive function

```
cfrac2float : lst:int list -> float
```

that takes a list of integers as a continued fraction and returns the corresponding real number.

4g1 Write a function

```
float2cfrac : x:float -> int list
```

that takes a real number and calculates its continued fraction. Recall that floating-point numbers are inaccurate, so you should check that r_i is reasonably close to 0 instead of comparing it for equality to 0.0. For example, `abs ri < 1e-10`.

4g2 Collect the above functions in a library as the interface file `continuedFraction.fsi` and implementation file `continuedFraction.fs`. Make a white- and blackbox test of these functions as the application `continuedFractionTest.fsx`.

Krav til afleveringen

Afleveringen skal bestå af

- en zip-fil, der hedder `4g_<navn>.zip` (f.eks. `4g_jon.zip`)

Zip-filen 4g_<navn>.zip skal indeholde en og kun en mappe som hedder 4g<navn>. I den mappe skal der ligge en src mappe og filen README.txt. I src skal der ligge følgende og kun følgende filer: continuedFraction.fsi, continuedFraction.fs og continuedFractionTest.fsx svarende til de relevante delopgaver. De skal kunne oversættes med fsharpc, og de oversatte filer skal kunne køres med mono. Funktioner skal dokumenteres ifølge dokumentationsstandarden som minimum ved brug af <summary>, <param> og <returns> XML-tagsne. Filen README.txt skal ganske kort beskrive, hvordan koden oversættes og køres.

God fornøjelse.