Programmering og Problemløsning
Datalogisk Institut, Københavns Universitet
Arbejdsseddel 8.5 - ingen aflevering

Jon Sporring

21. november

Denne arbejdsseddel indeholder et antal lekstra øveopgaver omhandlende højereordens funktioner, undtagelser og optiontyper.

Øveopgaver

5ø.0 Denne opgave omhandler polynomier. Et polynomium af grad n skrives som $f(x) = a_0 + a_1 x + a_2 x^2 + \ldots + a_n x^n = \sum_{i=0}^{n} a_i x^i$.

(a) Skriv en funktion `poly: a:float list -> x:float -> float`, som tager en liste af coefficienter med `a.[i]` $= a_i$ og en x-værdi og returnerer polynomiets værdi. Afprøv funktionen ved at lave tabeller for et lille antal polynomier af forskellig grad, med forskellige koefficienter og forskellige værdier for x og validér den beregnede værdi.

(b) Afled en ny funktion `line` fra `poly` således at `line : a0:float -> a1:float -> x:float -> float` beregner værdien for et 1. grads polynomium hvor `a0` $= a_0$ og `a1` $= a_1$. Afprøv funktionen ved at tabellere værdier for `line` med det samme sæt af coefficienter $a_0 \neq 0$ og $a_1 \neq 0$ og et passende antal værdier for x.

(c) Benyt Currying af `line` til at lave en funktion `theLine : x:float -> float`, hvor parametrene `a0` og `a1` er sat til det samme som brugt i Item 5ø.0b. Afprøv `theLine` som Item 5ø.0b.

(d) Lav en funktion `lineA0 : a0:float -> float` ved brug af `line`, men hvor `a1` og `x` holdes fast. Diskutér om dette kan laves ved Currying uden brug af hjælpefunktioner? Hvis ikke, foreslå en hjælpefunktion, som vil gøre en definition vha. Currying muligt.

5ø.1 Denne opgave omhandler integration. Integralet af næsten alle integrable funktioner kan approximeres som $\int_a^b f(x)\,dx \simeq \sum_{i=0}^{n-1} f(x_i)\Delta x$, hvor $x_i = a + i\Delta x$ og $\Delta x = \frac{b-a}{n}$.

(a) Skriv en funktion `integrate : n:int -> a:float -> b:float -> (f : float -> float) -> float`, hvis argumenter `n`, `a`, `b`, er som i ligningerne, og `f` er en integrabel 1 dimensionel funktion. Afprøv `integrate` på `theLine` fra Item 5ø.0c og på `cos` med $a = 0$ og $b = \pi$. Udregn integralerne analytisk og sammenlign med resultatet af `integrate`.

(b) Funktionen `integrate` er en approximation, og præcisionen afhænger af n. Undersøg afhængigheden ved at udregne fejlen, dvs. forskellen mellem det analytiske resultat og approximationen for værdier af n. Dertil skal du lave to funktioner `IntegrateLine : n:int -> float` og `integrateCos : n:int -> float` vha. `integrate`, `theLine` og `cos`, hvor værdierne for a og b og f er fastlåste. Afprøv disse funktioner for $n = 1, 10, 100, 1000$. Overvej om der er en tendens i fejlen, og hvad den kan skyldes.

5ø.2 Denne opgave omhandler untagelser (exceptions).

(a) Implementer fakultetsfunktionen $n! = \prod_{i=1}^{n} i$, $n > 0$ som `fac : n:int -> int` og kast en `System.ArgumentException` undtagelse, hvis funktionen bliver kaldt med $n < 1$. Kald `fac` med værdierne $n = -4, 0, 1, 4$, og fang evt. untagelser.

(b) Tilføj en ny og selfdefineret undtagelse `ArgumentTooBig of string` til `fac`, og kast den med argumentet `"calculation would result in an overflow"`, når n er for stor til `int` typen. Fang untagelsen og udskriv beskeden sendt med undtagelsen på skærmen.

(c) Lav en ny fakultetsfunktion `facFailwith : n:int -> int`, som `fac`, men hvor de 2 undtagelser bliver erstattet med `failwith` med hhv. argument `"argument must be greater than 0"` og `"calculation would result in an overflow"`. Kald `facFailWith` med $n = -4, 0, 1, 4$, fang evt. untagelser vha. `Failure` mønsteret, og udskriv beskeden sendt med `failwith` undtagelsen.

5ø.3 Denne opgave omhandler option typer og Stirlings formel. Stirlings formel er en approximation til fakultetsfunktionen via $\ln n! \simeq n \ln n - n$.

(a) Omskriv fakultetsfunktionen i Opgave 5ø.2b, som `facOption : n:int -> int option`, således at den returnerer `Some m`, hvis resultatet kan beregnes og `None` ellers. Kald `fac` med værdierne $n = -4, 0, 1, 4$, og skriv resultatet ud vha. en af `printf` funktionerne.

(b) Skriv en funktion `logIntOption : n:int -> float option`, som udregner logaritmen af `n`, hvis `n` > 0 og `None` ellers. Afprøv `logIntOption` for værdierne `-10`, `0`, `1`, `10`.

(c) Skriv en ny funktion `logFac : int -> float option` vha. `Option.bind` 1 eller flere gange til at sammensætte `logIntOption` og `facOption`, og sammenlign `logFac` med Stirlings approximation `n * (log n) - n` for værdierne $n = 1, 2, 4, 8$.

(d) Funktionen `logFac : int -> float option` kan defineres som en enkelt sammensætning af funktionerne `Some` og `Option.bind` en eller flere gange og med `logIntOption` og `facOption` som argument til `Option.bind`. Opskriv 3 udtryk, der bruger hhv. `|>` eller `>>` operatorerne eller ingen af dem.