exceptions

Jon Sporring

October 15, 2019

1 Lærervejledningn

Emne Untagelser og option typen

Sværhedsgrad Let

2 Introduktion

Denne opgave omhandler undtagelser (exceptions), option typer og Stirlings formel. Stirlings formel er en approximation til fakultetsfunktionen via

$$\ln n! \simeq n \ln n - n.$$

3 Opgave(r)

1. 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.

2. Tilføj en ny og selvdefineret 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 undtagelsen og udskriv beskeden sendt med undtagelsen på skærmen.

3. 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. undtagelser vha. `Failure` mønsteret, og udskriv beskeden sendt med `failwith` undtagelsen.

4. Omskriv fakultetsfunktionen i Opgave 2, 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.

5. 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`.

6. 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 $n = 1, 2, 4, 8$.

7. 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.

8. Der skal laves følgende implementationer af samme funktion

   ```
   safeIndexIf : arr:'a [] -> i:int -> 'a
   safeIndexTry : arr:'a [] -> i:int -> 'a
   safeIndexOption : arr:'a [] -> i:int -> 'a option
   ```

   De skal alle returnere værdien i `arr` på plads `i`, hvis `i` er et gyldigt index, og ellers håndtere fejlsituationen. Fejlsituationerne skal håndteres forskelligt:

   - `safeIndexIf` må ikke gøre brug af `try-with` og må ikke kaste en undtagelse.
   - `safeIndexTry` skal benytte `try-with`, og ved fejltilstand skal `failwith` kaldes.
   - `safeIndexOption` skal returnere `None` i en fejlsituation.

   Der skal laves en kort afprøvning af alle 3 funktioner, der indebefatter at den indicerede værdi udskrives til skærmen som et heltal og ikke som en option type. Afprøvningen skal også afprøve korrekt håndtering af funktionernes evt. kastede undtagelser. Lav en kort beskrivende sammenligning af metodernes evne til at håndtere fejltilstande.