KS i Programmeringsparadigm 2014, del 1: Funktionell Programmering
2014-09-22 15.15-16.00 med efterföljande kamraträttning

Inga hjälpmedel är tillåtna. Skriv svaren direkt på blanketten. Bonuspoäng från Haskelllabbarna hösten 2014 kommer automatiskt att tillgodoräknas på denna KS. 12 poäng (utifrån max 20) krävs för godkänt. Sitt kvar ända till klockan 16.00. Då lämnar alla in samtidigt. Därefter tar kamraträttning vid.

1. (4 p)

   En viktig egenskap vid lat evaluering är terminering.

   a) (1 p) Förklara och visa med ett exempel när det inte terminerar oavsett evalueringsstrategi.

   b) (3 p) Förklara varför och visa med ett exempel när det terminerar endast med lat evaluering! OBS! Förklara också varför det inte terminerar då man använder en annan evalueringsstrategi än den för lat evaluering.

2. (8 p)

Följande kod kan används för att implementera matriser i Haskell.

```
data Matris a = M
        Int -- Antal rader
        Int -- Antal kolumner
        [a] -- matriselementen
        deriving (Show, Eq)
```

a) (2 p) För att skapa 2×2-matrisen $\begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix}$ görs anropet nedan vars resultat lagras i `m1`. Motivera varför resultaten av anropen som lagras i `m2` och `m3` också fungerar, dvs inte ger körningsfel.

```
m1 = M 2 2 [1,2,3,4]
m2 = M 2 2 [[1,2],[1,3],[5,1],[7,8]]
m3 = M 1 1 "qwe"
```

.....

.....

.....

b) (3 p) Med funktionen `matrisOperation` nedan kan man t.ex. addera och subtrahera två matriser. Hur ska anropet se ut för att addera `m1` med sig själv?

```
matrisOperation matris1 matris2 operation = M r c elem
        where (r,c) = msize matris1
              l1 = elemlist matris1
              l2 = elemlist matris2
              elem = [operation el | el <- (elements l1 l2)]

msize (M rader kolumner _) = (rader, kolumner)

elemlist (M _ _ elemlist) = elemlist

elements [] _ = []
elements _ [] = []
elements (el1:list1) (el2:list2) = (el1, el2) : (elements list1 list2)
```

.....

.....

.....

c) (3 p) Skriv om raden `elem = [operation el | el <- (elements l1 l2)]` utan att använda listomfattning och så att funktionens funktionalitet bibehålles.

.....

.....

.....

3. (8 p)

I denna uppgift tittar vi på hur man kan göra fakultetsberäkningar i imperativ och funktionell programmering. (Fakulteten av ett positivt heltal n är produkten av alla heltal från 1 till n.)

För nedanstående paradigm med tillhörande villkor är din uppgift att för anropet `fakultet 4` (beräkna fakulteten av talet 4) tydligt visa och förklara hur i varje anrops/beräkningssteg:

1. anropet/beräkningssteget ser ut och
2. minnesutrymmet används och uppdateras

Du behöver inte svara med kod (men du får naturligtvis skriva sådan för egen del)!

Lös uppgiften med

a) (3 p) det imperativa paradigmet OCH med iteration (= loop).

.....

.....

.....

.....

.....

b) (3 p) det funktionella paradigmet OCH rekursion men INTE med den svansrekursiva idén.

.....

.....

.....

.....

.....

c) (2 p) Förklara med ord vad som skiljer de två paradigmen åt, utgå ifrån uppgiften ovan.

.....

.....

.....

.....