UMEÅ UNIVERSITET
Institutionen för Datavetenskap
Laborationsrapport

February 11, 2020

5DV086 - LAB 1

Programspråk 7.5 hp
VT 2020

Parkeringshus

Namn Emil Söderlind (id15esd@cs.umu.se)

Kursansvarig
Jan Erik Moström

Contents

1 Information från specifikation

Följande information tillhandahölls från kursen gällande laborationens utformning och krav.

1.1 Introduktion

Denna laboration består i att lösa den givna uppgiften, och skriva en laborationsrapport. Fokus ligger på att förstå och applicera grunderna i Haskell och den funktionella paradigmen, som rekursion, namngivna värden, tupler och listor. även labrapportens utformning, stavning och språk är relevant, speciellt för bonuspoäng.

Laborationen ska utföras och lämnas in individuellt, men det är såklart tillåtet (till och med uppmuntrat) att prata om, och fråga om hjälp från dina kurskamrater gällande labben. Tumregeln är att du ska ha skrivit koden och rapporten själv, och ska förstå den till den grad att du själv kan förklara vad du gjort och varför.

1.2 Bakgrund

Ett P-hus har infört automatisk registrering av inkommande och lämnande bilar, detta för att få en automatisk debitering av de som använder P-huset. Din uppgift är att bearbeta registreringslistan till en mer användbar form. Registreringslistan består av en tupel för varje ankommande bil och för varje lämnande bil. Tupeln innehåller information om registreringsnummer, huruvida det är en ankommande bil och tiden då registreringen sker. Representerad som ett värde i Haskell får registreringslistan typen

```
[(String, Bool, (Integer, Integer))]
```

där tupelns värden står för

```
(registreringsnummer, ankommande?, (timmar, minuter))
```

1.3 Uppgift

Resultatet från din bearbetning ska vara ett par. Det första elementet är registreringsnumret på den bil som varit längst i p-huset under dygnet. Det andra elementet är en lista som redovisar hur länge varje bil har varit i P-huset under dagen. Listan skall innehålla par av 1) ett registreringsnummer och 2) ett par av timmar och minuter. Har en bil varit i parkeringshuset flera gånger under en dag skall tiderna summeras ihop (dvs ett registreringsnummer skall bara förekomma en gång i resultatlistan). Listan behöver inte ha någon specifik ordning.

Din funktion ska alltså ha signaturen:

```
phus :: [(String, Bool, (Integer, Integer))]
    -> (String, [(String, (Integer, Integer))])
```

Tänk efter hur de två delarna av resultatet beror av varann, hur problemet kan delas upp i mindre delproblem, och i vilken ordning delresultat bör beräknas.

1.4 Indata

Registreringslistan uppfyller vissa krav:

Listan beskriver en dag. P-huset är tomt från början och i slutet. Varje bil som ankommer lämnar också P-huset. En bil kan besöka huset flera gånger under samma dag. Listan är ordnad efter den registrerade tiden Det kommer alltid finnas exakt en bil som stått i parkeringhuset längst tid under en dag. Exempel:

```
Prelude> phus [("CZX537",True,(8,10)),("ABC123",True,(9,20)),
  ("CZX537",False,(10,30)),("DEF456",True,(11,0)),
```

```
        ("ABC123",False,(14,10)),("DEF456",False,(15,10))]

    ("ABC123", [("CZX537",(2,20)),("ABC123",(4,50)), ("DEF456",(4,10))])}
```

2 Användarhandledning

2.1 Inlämningens innehåll

Inlämningen består av Phus.hs samt PDF:filen du just nu läser, ps_ou1_id15esd.pdf. Phus.hs innehåller samtlig Haskell-kod för laborationen och PDF:en innehåller information om programmet och lösningen.

2.2 Kompilering och körning av programmet

För att köra programmet ska vederbörande utföra nedan steg:

1. Se till att ghci är installerat på din maskin
2. Befinna dig i samma katalog som Phus.hs med valfritt shell
3. Starta ghci genom att köra: "ghci"
4. Ladda Phus.hs genom att köra: ":load Phus.hs"
5. Med en korrekt registreringslista X (beskriven i 1.2) köra: "phus X"
6. Programmet kommer då köra och skriva ut resultatet i enlighet med beskrivning under 1.4

3 Systembeskrivning

Programmet består av ett antal Haskell-funktioner som i X steg går igenom indatan, processar den och skriver ut resultatet. Programmet är byggt utifrån nedan X steg:

1. Summerar tiden (minuter) i parkeringshuset för ett registreringsnummer
2. Gör ovan (1.) för samtliga registreringsnummer
3. Summera flera besök i parkeringshuset för ett registreringsnummer
4. Gör ovan (4.) för samtliga registeringsnummer
5. Hitta bilen som befunnit sig längst i parkeringshuset
6. Konvertera tiden för varje registreringsnummer till timmar och minuter
7. Skriv ut bilen som varit längst i parkeringshuset (5.) + samtliga bilars summerade tid (4.)

4 Testkörningar

För att säkerställa att programmet är korrekt har tester utförts. Utifrån den givna Registers.hs-filen, där indata med motsvarande korrekt utdata åtefanns, har tester utifrån dessa utförts. Genom att köra phus-funktionen med den givna indatan och manuellt jämfört utdatan mot den korrekta utdatan given i Registers.hs-filen.