Practicumopdracht 2: fpCal

De eerste practicumopdracht bestond uit het implementeren van een aantal eenvoudige operaties op elektronische agenda's. Die agenda's werden in Haskell gerepresenteerd met behulp van tupels en lijsten. In deze tweede opdracht zullen we opnieuw met elektronische agenda's werken, maar ditmaal gebruiken we zogenaamde algebraïsche datatypes om de agenda's te representeren. (Algebraïsche datatypes zijn types die gedefinieerd zijn met behulp van het sleutelwoord **data**.) Bovendien beperken we ons dit keer niet tot het schrijven van losse operaties op agenda's, maar implementeren we een heuse interactieve applicatie: fpCal.

De opdracht wordt uitgevoerd in groepjes van maximaal twee studenten.

Startbestanden

Op de wikipagina van het practicum tref je een zipbestand aan met daarin de startcode voor de opdracht. (`http://www.cs.uu.nl/wiki/FP/Practicum`) Deze startcode is verdeeld over twee Haskell-bestanden, een `.fpc`-bestand en een HTML-bestand.

Het bestand `Parsing.hs` bevat bibliotheekje met ontleedcombinatoren. De werking en het gebruik van deze combinatoren worden uitgelegd in hoofdstuk 8 van het dictaat (`http://www.cs.uu.nl/wiki/FP/CourseLiterature`), maar zijn verder niet van belang voor het maken van deze opdracht. Het is niet de bedoeling dat je wijzigingen aan dit bestand aanbrengt. Het tweede Haskell-bestand, `FpCal.hs`, bevat startcode voor de te implementeren applicatie en bestaat – vooralsnog – uit een hoofdprogramma, een serie typedefinities voor elektronische agenda's, een ontleedprogramma en een programma dat deelagenda's berekent. Tijdens het practicum zul je het hoofdprogramma moeten uitbreiden. Daarnaast moet je programma's toevoegen.

Het bestand `FP.fpc` bevat agendagegevens voor een gedeelte van het rooster voor het vak Functioneel programmeren. In het bestand `FP.html` vind je een HTML-weergave van diezelfde gegevens.

Inleveren

Vermeld aan het begin van het bestand `FpCal.hs` – in commentaarregels – je naam, login en studentnummer. (Als je de opdracht met zijn tweeën doet, vermeld dan de gegevens van beide studenten.) De opdracht bestaat uit drie opgaven. Zorg dat de namen en types van je functies overeenkomen met wat in de opgaven gevraagd wordt. Dit voorkomt misverstanden bij het nakijken. Als je klaar bent met de opdracht, lever je je uitbreiding van het bestand `FpCal.hs` in via Submit (`http://www.cs.uu.nl/docs/submit`). De deadline is op **maandag 27 maart 2006** om **23:59 uur**. Zorg ervoor dat de ingeleverde code werkt met de laatste versie van de GHC (versie 6.4.1).

Agenda's representeren

Voor de representatie van elektronische agenda's in Haskell gebruiken we de volgende algebraïsche datatypen en typesynoniemen:

```
data Agenda       = Agenda AgendaNaam [Afspraak]

data Afspraak     = Afspraak Tijdslot Omschrijving Locatie [Deelnemer]
data Tijdslot     = Tijdslot Tijdstip Tijdstip
data Tijdstip     = Tijdstip Jaar Maand Dag Uur Minuut Seconde
data Deelnemer    = Deelnemer Naam EMailAdres

type AgendaNaam = String
type Omschrijving = String
type Locatie      = String
type Naam         = String
type EMailAdres   = String

type Jaar         = Int
type Maand        = Int
type Dag          = Int
type Uur          = Int
type Minuut       = Int
type Seconde      = Int.
```

Een agenda bestaat dus uit een tekenreeks, namelijk de naam van de agenda, en een lijst met afspraken. Van elke afspraak houden we een tijdslot, een omschrijving, een locatie en een lijst met deelnemers bij, waarbij omschrijvingen en locaties tekenreeksen zijn. Een tijdslot wordt gekenmerkt door een begin- en een eindtijdstip, die elk opgebouwd zijn uit zes numerieke componenten: een jaartal, een maandnummer, een dagnummer, een uuraanduiding, een minuutaanduiding en een secondeaanduiding. (Let op: in vergelijking met het type dat in de eerste opdracht gebruikt werd voor de representatie van tijdstippen, verschilt het type Tijdstip zowel in het *aantal* velden als de *volgorde* van die velden.) Bijvoorbeeld: de deadline voor deze opdracht wordt gerepresenteerd door de waarde *Tijdstip* 2006 3 27 23 59 0. Deelnemers, tenslotte, worden gerepresenteerd met behulp van twee tekenreeksen: één voor hun naam en één voor hun e-mailadres.

De startcode bevat een functie

```
voorInterval :: Agenda → Interval → [Afspraak],
```

die voor een gegeven agenda en interval de bijbehorende deelagenda oplevert, waarbij een interval gedefinieerd is in termen van een begin- en een eindtijd:

```
type Interval = (Tijdstip, Tijdstip).
```

Het fpCal-formaat

Bekijk nu het bestand `FP.fpc` eens. Het bevat de agendagegevens voor een gedeelte van het rooster voor het vak Functioneel programmeren. De gegevens zijn gerangschikt in een formaat dat we het fpCal-formaat zullen noemen. Het fpCal-formaat is een deelverzameling van de zogenaamde iCalendar-standaard. Deze standaard wordt gebruikt om agendagegevens zó op te slaan

dat ze gemakkelijk gedeeld kunnen worden door verschillende applicaties. Voorbeelden van applicaties die deze standaard (gedeeltelijk) ondersteunen zijn Microsoft Outlook en iCal (het agendaprogramma voor het Mac OS). Dat het fpCal-formaat een deelverzameling is van het iCalendar-formaat wil zeggen dat elk fpCal-bestand een correct iCalendar-bestand is, maar dat het omgekeerde niet noodzakelijk waar is.

Een voorbeeld van een agenda in fpCal-formaat is

```
BEGIN:VCALENDAR
SUBSET:fpCal
BEGIN:VEVENT
DTSTART:20060310T130000
DTEND:20060310T150000
SUMMARY:Eerste toets FP
LOCATION:EDUC-alfa
ATTENDEE;CN="Jeroen Fokker":MAILTO:jeroen@cs.uu.nl
ATTENDEE;CN="Stefan Holdermans":MAILTO:stefan@cs.uu.nl
END:VEVENT
END:VCALENDAR
```

Deze agenda bevat één afspraak: de eerste toets voor het vak Functioneel programmeren. (Maar de agenda zou dus ook uit meerdere afspraken kunnen bestaan; zie `FP.fpc`. Ook zouden er helemaal geen afspraken in kunnen staan.) De regels `BEGIN:VCALENDAR` en `END:VCALENDAR` markeren het begin en het einde van de agenda, net als `BEGIN:VEVENT` en `END:VEVENT` het begin en einde van een afspraak markeren. De regel `SUBSET:fpCal` geeft aan dat het hier om een fpCal-bestand gaat (in tegenstelling tot een 'gewoon' iCalendar-bestand). Alle velden in een fpCal-bestand moeten in de hierboven weergegeven volgorde staan. `ATTENDEE`-velden zijn optioneel: er kunnen immers ook afspraken zonder deelnemers zijn.

In de startcode is een functie

laadAgenda :: AgendaNaam → IO (Maybe Agenda)

opgenomen die, gegeven een agendanaam *nm*, eerst het bestand met de bestandsnaam *nm* ++ `".fpc"` probeert te openen om vervolgens een poging te doen om de inhoud van het bestand als agenda in het fpCal-formaat te ontleden. De agenda in het bestand `FP.fpc` zou je dus bijvoorbeeld kunnen inladen met behulp van de expressie

laadAgenda `"FP"`.

Hoofdprogramma

Het hoofdprogramma kun je starten door het Haskell-bestand `FpCal.hs` te laden in de Hint-interpreter of in GHCi en vervolgens de functie *main* aan te roepen:

```
*FpCal> main
Welkom bij fpCal!
```

```
Tot ziens!
**FpCal>
```

Uitbreidingen

Hoewel het gegeven hoofdprogramma uitermate vriendelijk is – de gebruiker wordt netjes begroet en al even netjes weer gedag gezegd – is het nog vrij beperkt. Deze practicumopdracht bestaat dan ook uit het implementeren van een aantal uitbreidingen op het programma.

Zoals al eerder gemeld werd, bestaat de opdracht uit drie opgaven. In de eerste twee opgaven draait het om het schrijven van eenvoudige deelprogramma's; in de laatste opgave worden de deelprogramma's verwerkt in het hoofdprogramma. Het is niet vereist dat je uiteindelijke programma's exact werken als in de hieronder weergeven interactieve sessies. Deze sessies dienen slechts als illustratie van de vereiste functionaliteit. Documenteer je programma's, zodat duidelijk is wat het gewenste gedrag is. Je zou bijvoorbeeld voorbeelden van runs kunnen opnemen als commentaar bij je code. (Programma's waarvan onduidelijk is wat ze zouden moeten doen, zullen worden 'fout' gerekend.)

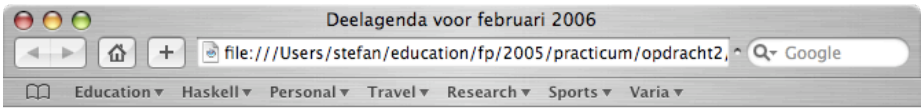
Bij het schrijven van de programma's mag je ervan uitgaan dat de gebruiker nooit zal vragen om een agenda uit een niet-bestaand bestand te lezen (als dat wel gebeurt, dan 'crasht' de functie *laadAgenda*). Wel moet je rekening houden met de mogelijkheid dat een bestand gegevens bevat die niet in het juiste formaat staan (*laadAgenda* levert dan *Nothing* op).

Opgave 1. Schrijf een programma *webuitvoer* :: IO () dat een HTML-weergave van een maand-agenda genereert (zie figuur 1). Het is de bedoeling dat het programma de gebruiker eerst om de naam van een agenda vraagt en deze vervolgens laadt uit een bijbehorend `.fpc`-bestand. Daarna wordt de gebruiker gevraagd om een maand te specificeren. Vervolgens produceert het programma HTML-code voor een tabel met daarin de afspraken voor de betreffende maand en schrijft deze code weg naar een bestand met de naam $nm +\!\!+$ `".html"` waar *nm* de naam van de agenda is. Het startbestand `FP.html` bevat een voorbeeld van de te genereren HTML-code. Een voorbeeld van een run van het programma *webuitvoer* is:

```
*FpCal> webuitvoer
Agendanaam?
FP

Maand?
2

Jaar?
2006

Uitvoer geschreven naar FP.html.
*FpCal>
```

In geval van incorrecte invoer (bijvoorbeeld een incorrect fpCal-bestand, een onjuist maandnummer of niet-numerieke invoer voor jaar- en maandnummers)

Figuur 1 HTML-weergave van de agenda `"FP"`



Deelagenda voor februari 2006

15-02-2006 13:00:00	15-02-2006 15:00:00	Werkcollege groep 1	BBL-416	Stefan Holdermans Marilou Dubbeld
15-02-2006 13:00:00	15-02-2006 15:00:00	Werkcollege groep 2	BBL-420	Stefan Holdermans Marien Krouwel
15-02-2006 13:00:00	15-02-2006 15:00:00	Werkcollege groep 3	BBL-426	Jeroen de Knijf Remco Ruijsenaars
15-02-2006 13:00:00	15-02-2006 15:00:00	Werkcollege groep 4	BBL-430	Jeroen de Knijf Marcel Sondaar
15-02-2006 15:00:00	15-02-2006 17:00:00	Hoorcollege	WENT-groen	Doaitse Swierstra
17-02-2006 09:00:00	17-02-2006 11:00:00	Practicum	BBL-461	Stefan Holdermans Marien Krouwel Remco Ruijsenaars Marcel Sondaar
17-02-2006 11:00:00	17-02-2006 13:00:00	Werkcollege groep 1	BBL-416	Stefan Holdermans Marilou Dubbeld
17-02-2006 11:00:00	17-02-2006 13:00:00	Werkcollege groep 2	BBL-420	Stefan Holdermans Marien Krouwel
17-02-2006 11:00:00	17-02-2006 13:00:00	Werkcollege groep 3	BBL-426	Jeroen de Knijf Remco Ruijsenaars
17-02-2006 11:00:00	17-02-2006 13:00:00	Werkcollege groep 4	BBL-430	Jeroen de Knijf Marcel Sondaar
17-02-2006 13:00:00	17-02-2006 15:00:00	Hoorcollege	AARD-groot	Doaitse Swierstra

moet het programma een passende foutmelding geven. Bijvoorbeeld:

```
*FpCal> webuitvoer
Agendanaam?
FP

Maand?
bla

Fout: ongeldig maandnummer!
*FpCal>
```

(Hint: maak gebruik van de aangeleverde functies *laadAgenda* en *voorInterval*.) □

Opgave 2. Schrijf een programma *toevoegen* :: IO () dat de gebruiker in staat stelt om interactief afspraken aan een agenda toe te voegen. Het is de bedoe-

ling dat het programma de gebruiker eerst om de naam van een agenda vraagt en deze vervolgens laadt uit een bijbehorend `.fpc`-bestand. Daarna wordt de gebruiker gevraagd om – stap voor stap – de gegevens van de toe te voegen afspraak te verstrekken. Het programma produceert dan vervolgens fpCal-code voor de nieuwe agenda (dat wil zeggen: een agenda met daarin zowel de afspraken uit de oorspronkelijke agenda als de toegevoegde afspraak) en schrijft deze code weg naar een bestand met de naam $nm +\!\!+ \texttt{".fpc"}$ waar *nm* de naam van de agenda is. Het ingelezen bestand wordt dan dus overschreven! Een voorbeeld van een run van het programma *toevoegen* is:

```
*FpCal> toevoegen
Agendanaam?
FP

Begintijd?
17 2 2006 9 00 00

Eindtijd?
17 2 2006 11 00 00

Omschrijving?
Werkcollege groep 5

Locatie?
BBL-462

Deelnemer toevoegen (j/n)?
j

Naam?
Stefan Holdermans

E-mailadres?
stefan@cs.uu.nl

Deelnemer toevoegen (j/n)?
j

Naam?
Marilou Dubbeld

E-mailadres?
mdubbeld@students.cs.uu.nl

Deelnemer toevoegen (j/n)?
n

Uitvoer geschreven naar FP.fpc.
*FpCal>
```

In geval van incorrecte invoer moet een passende foutmelding gegeven wor-

den. □

Opgave 3. Pas het hoofdprogramma *main* :: IO () zodanig aan dat het de gebruiker in staat stelt om te kiezen uit een van de programma's *webuitvoer* en *toevoegen* uit opgave 1 en 2. Nadat de gebruiker een keuze gemaakt heeft, wordt het gekozen programma uitgevoerd. Bijvoorbeeld:

```
*FpCal> main
Welkom bij fpCal!

Programma (w/t)?
w

Je hebt gekozen voor webuitvoer.

Agendanaam?
```

Enzovoort. Als het gekozen programma in zijn geheel doorlopen is, wordt het hoofdprogramma afgesloten:

```
Tot ziens!
*FpCal>
```

In geval van incorrecte invoer moet een passende foutmelding gegeven worden. □