Programming Paradigms 2023
Session 12 : Monadic parsing

Preparing for the session

Hans Hüttel

22 November 2022

Where nothing else is mentioned, chapters and page numbers refer to *Programming in Haskell.*

The video podcast

You can watch the podcast on YouTube via the course page on Moodle.

Tuesday 28 November 2023 – Monadic parsing

Please read

- Chapter 13 of *Programming in Haskell.*

Learning goals for the session

- To be able to explain the central ideas underlying the Parser monad
- To be able to explain how the Parser monad can be used for building parsers
- To be able to build a parser using monadic parsing

How you should prepare before we meet on Tuesday

Before we meet, watch the podcast and read the text. You can do this in any order you like. Also see if you can solve the following two small discussion problems. We will talk about them in class.

Please note! *For this session, we will need the modules mentioned in chapter 13 of the textbook and the many functions defined in that chapter. From the entry for this session on the Moodle course page you can get the file Parsing.hs that contains all of this. You need to compile and import it as a module. To compile Parsing.hs, use the ghc compiler for Haskell. In a terminal, write*

```
ghc -I. --make Parsing.hs
```

This will generate files called `Parsing.hi` *and* `Parsing.o`*. To simplify things in the following, make sure that these files are in the same directory as your program that uses them.*

1. In Session 10, we saw how one can declare an algebraic datatype for onions. Here let us consider integer onions:

   ```
   data Onion = Core Integer | Layer Onion deriving Show
   ```

   Use the Parser monad to define a parser theonion that can parse strings. Whenever a string is of the required form, the parser must then give us the corresponding value of type Onion.

   As an example, if we give the parser the string LLLLL7, we should get (Layer (Layer (Layer (Layer (Layer (Core 7))))))

2. The language $L = \{a^n b^n \mid n \geq 0\}$ is a well-known example of a context-language that is not also a regular language. For instance, $abba \in L$ but $aab \notin L$. L can be defined using the context-free grammar

   $$S \to aSb \mid \varepsilon$$

   Use the Parser monad to define a parser ab that recognizes the language L.

What happens on Tuesday?

When we meet, students that have been contacted by me who will present the solutions to the small discussion problems above.

Problems for Tuesday

For the plenary session we will solve and discuss a collection of problems that can be found on a separate page, available on the day of the session.