Sketch of a new course plan for AFP (20100205)

- Credit distribution: New "official" division of work (closer to reality):
  - Exam 3.0 (was 7.5)
  - Lab 4.5 (was 0.0)
- Examiner: will be Patrik Jansson (is formally Koen Claessen now)
- Prerequisites:
  - Functional Programming
  - Programming languages / programming paradigms {- new -}
  - Notions: Abstract syntax tree, semantics, interpreter, compiler. Algorithms, complexity, divide-and-conquer. Induction proofs and simple logic, equality reasoning. {- new -}
- Aim: "The aim of the course is to explore the powerful mechanisms that functional programming languages offer to solve real problems and structure larger programs. The focus lies on library design and the concept of embedded languages." {- Move this: "The programming language used in the course is Haskell." It is true, but not an aim. -}

Learning outcome

After completion of this course, the student should be able to

- design an embedded domain specific language (EDSL)
  - (abstract) syntax, semantics
  - implement EDSL in Haskell (as a combinator library)
- read, understand and extend Haskell programs which use advanced type system features
  - type classes
  - (generalized) algebraic datatypes
  - functors, monads and monad transformers
- use specification based development techniques
  - formulate and test properties about the program
  - reason about correctness of functional programs
  - transform programs on the basis of such reasoning

- explain and discuss the above topics

Drop from learning outcomes:

- "GUI programming techniques" (there is no GUI (= user interface) focus - only graphics as an example app.)
- "Parser and pretty printing combinators" - again, only example. app.
- "Examples of concrete real-life applications of the above are also part of the course." - belongs to Contents.

Contents:

The big advantage with functional languages is that language constructions can be given names and thereby reused, using higher order functions. Functional programs can therefore often be constructed by composing constructions from a library. This method enables a way to construct programs quickly and with a high degree of correctness. This is the central idea in this course.

Remove: "We can learn a lot from studying the standard library of list functions such as map, fold and so on. These functions can be generalized to operate on other datatypes." True, but not really tought in the course.

Realistic functional programs must handle changes in state, exceptions, backtracking and other "non-functional" behaviors. We will look at how these can be modeled in a purely functional manner. The concept of "monads" will help us here.

Armed with this knowledge we will construct domain specific libraries, designed to construct programs in a certain application domain. This type of library can be said to define a domain specific language, since the constructions the programmer uses to construct larger programs mainly consists of library functions. We will study libraries for parsing, pretty printing, graphics, pseudo-parallel programming and web applications. The course will also present some recent research which can make the contents of the course vary to some degree.

Examination

- Possible new idea of examination (change to lab sequence): do a review of an existing library (from hackage, say), identifying certain patterns etc. + suggest improvements.
- Add optional weekly exercises with bonus points for the exam
- Reduce the lab work-load to compensate for the first two additions