Advanced Functional Programming

Software Engineering Chair and Programming Systems Lab

Assignment

1. Read *How to make* ad-hoc *polymorphism less* ad hoc by Philip Wadler and Stephen Blott, 16th Symposium on Principles of Programming Languages, 1989.

2. We encourage you to read and discuss the paper with other students. Unlike *Fun with binary heap trees* by Chris Okasaki, this is a research paper that was presented at a top conference. Discuss the differences between the two papers.

3. Summarize the paper *in your own words* on one page. Most of your summary should identify the problem being solved, the solution, advantages, and limitations. After you have done that you can offer also your opinion about the paper, best backed up by some arguments. You can write your summary either in German or English.

4. Put your name and student ID on your summary and drop off a printout at office 326/45 until Friday, November 4th at noon (12am). If the door is closed, slide your printout under the door. No Emails.

Get Your Hands Dirty

Install a Haskell interpreter and solve the following programming exercise. Implement a data type `BigNum` that implements big numbers by utilizing `[Int]`. The new type should be an instance of the type classes `Eq`, `Show`, and `Num`; this implies the functions you have to provide for `BigNum`. You have to use `data` or `newtype` to create a type than can be an instance of a type class.

```
class (Eq a, Show a) => Num a where
    (+), (-), (*)  :: a -> a -> a
    negate         :: a -> a
    abs, signum    :: a -> a
    fromInteger    :: Integer -> a
    fromInt        :: Int -> a
```

Efficiency is not a concern! Use the simplest algorithms and base possible.