Imperative Core in Functional Languages

Natela Archvadze

e-mail: natela.archvadze@tsu.ge

Department of Computer Sciences
Faculty of Exact and Natural Sciences
Iv. Javakhishvili Tbilisi State University
University str., 13, Georgia

Functions in Haskell are pure functions that take all their input as arguments and produce all their output as results. However, many programs require some form of side effect that would appear to be at odds with purity, such as reading input from the keyboard, or writing output to the screen, while the program is running. Haskell provides a uniform framework for handling effects without compromising the purity of functions, based upon the mathematical notion of a monad (for Haskell) and Computational Workflows (for F#).

Keywords: Pure Functions, Monadic effects, Lazy evaluation, Computational Workflows.

References

[1.] Philip Wadler. Monads for Functional Programming. In Manfred Broy, editor, *Proceedings of the Marktoberdorf Summer School on Program Design Calculi*. Springer–Verlag, August 1992.

[2.] Simon Peyton Jones. Tackling the Awkward Squad: Monadic Input/Output, Concurrency, Exceptions, and Foreign-Language Calls in Haskell. In Tony Hoare, Manfred Broy, and Ralf Steinbruggen, editors, *Engineering Theories of Software Construction*, pages 47–96. IOS Press, 2001. Presented at the 2000 Marktoberdorf Summer School.

[3.] Saunders MacLane. *Categories for the Working Mathematician*. Number 5 in Graduate Texts in Mathematics. Springer-Verlag, 1971.