The purpose of this book is to teach logic and mathematical reasoning in practice, and to connect logical reasoning with computer programming. The programming language that will be our tool for this is Haskell, a member of the Lisp family. Haskell emerged in the last decade as a standard for lazy functional programming, a programming style where arguments are evaluated only when the value is actually needed. Functional programming is a form of descriptive programming, very different from the style of programming that you find in prescriptive languages like C or Java. Haskell is based on a logical theory of computable functions called the lambda calculus. Lambda calculus is a formal language capable of expressing arbitrary computable functions. In combination with types it forms a compact way to denote on the one hand functional programs and on the other hand mathematical proofs. Haskell can be viewed as a particularly elegant implementation of the lambda calculus. It is a marvelous demonstration tool for logic and maths because its functional character allows implementations to remain very close to the concepts that get implemented, while the laziness permits smooth handling of infinite data structures.