Introduction to Haskell

Haskell is a computer programming language. In particular, it is a [**polymorphically**](https://wiki.haskell.org/Polymorphism)[**statically typed**](https://wiki.haskell.org/Typing), [lazy](https://wiki.haskell.org/Lazy_evaluation), [purely functional](https://wiki.haskell.org/Functional_programming) language, quite different from most other programming languages. The language is named for [Haskell Brooks Curry](https://wiki.haskell.org/Haskell_Brooks_Curry), whose work in mathematical logic serves as a foundation for functional languages. Haskell is based on [lambda calculus](https://wiki.haskell.org/Lambda_calculus).

Haskell offers:

* Substantially increased programmer productivity (Ericsson measured an improvement factor of between 9 and 25 using Erlang, a functional programming language similar to Haskell, in one set of experiments on telephony software).
* Shorter, clearer, and more maintainable code.
* Fewer errors, higher reliability.
* A smaller "semantic gap" between the programmer and the language.
* Shorter lead times.

Haskell is a wide-spectrum language, suitable for a variety of applications. It is particularly suitable for programs that need to be highly modifiable and maintainable.

Haskell is **lazy**. That means that unless specifically told otherwise, Haskell won't execute functions and calculate things until it's really forced to show you a result.

Haskell is **statically typed**. When you compile your program, the compiler knows which piece of code is a number, which is a string and so on. That means that a lot of possible errors are caught at compile time.