Types

What are types? Types are how you describe the data you will work with. They allow you to classify parameters of functions. This should help you when designing functions: whenever you want to make a function always think about what its type should be. Before you give the definition of a function you normally write the type signature using the :: notation. For example,

```
lion :: BigCat
```

is like the function introducing itself and saying, "I am lion and I have type BigCat".

The Haskell compiler is very passionate about its types. Knowledge and understanding of types should be integral in your implementation process, helping you get to solutions quicker, and allowing you to get on with the compiler well. BEWARE running blindly into smashing out a definition without considering types will lead to a compiler strop, and who can blame it? Seriously though, considering types allows you to improve your development practice. The typed nature of Haskell also means that most errors are found at compile time as opposed to run time, meaning that software written in Haskell is usually more reliable than that of other languages.

Types also have a really cool relationship to algebra and this short article explains it really well.