Partial application

- `Uncurried functions.` 1 function, many arguments.
- `Curried functions.` Many functions, 1 argument apiece.

Typecheck

You can check types that *aren't implemented yet*. Give it an *undefined* to bind the signature.

3 types of type signatures: concrete, constrained polymorphic, and parametrically polymorphic.

Parametricity means that the behavior of a function with respect to the types of its (parametrically polymorphic) arguments is uniform. The behavior cannot change just because it was applied to an argument of a different type.

1 Type inference

Don't have to assert a type for value because Haskell has **type inference** (algorithm); which infer the most generally applicable (polymorphic) type that is still correct.

Monomorphism restriction

Top-level declarations by default will have a concrete type if any can be determined.

```
{-# LANGUAGE NoMonomorphismRestriction #-}

module DetermineTheType where

-- simple example
example = 1
```

2 Definition

1. **Polymorphism**: *parametric* or *ad-hoc polymorphism*. Smaller sets from a large set.
2. **Type inference:** Infer principal types from terms without needing explicit type annotations.
3. **Type variable** is: `a → a`
4. **Typeclass:** a means of expressing faculties or interfaces that multiple datatypes may have in common.
5. **Ad-hoc polymorphism** (*constrained polymorphism*)
6. **Module**: The unit of organization