AGDA

LINGUAGEM COM TIPOS DEPENDENTES

- Extensão da teoria dos tipos (Intuicionista) de Martin-Löf.
  - Um programa e do tipo T sempre vai terminar com um resultado do tipo T
  - Garantia de que não tem Runtime Error.
  - Todo programa escrito termina (a não ser que seja especificado pelo programador).
- Consegue representar proposições lógicas usando tipos (Curry-Howard).
- Por ser fortemente tipada (isto é, toda variável tem um tipo específico) e ter tipos dependentes, Agda consegue provar teoremas matemáticos e rodar tais provas como algoritmos.

TIPOS DEPENDENTES?

- São introduzidos ao ter famílias de tipos indexadas por objetos de outro tipo.
  - Por exemplo: definir '*Vec n*' um **vetor** de tamanho **n.** Temos uma família de tipos indexada por objetos em Nat (tipo parametrizado pelos números naturais).
- (a : A) -> (B a)
  - B é uma família de tipos indexada por elementos do tipo A
- Matriz n x n – função identidade que pega um número natural n e produz a identidade da matriz n x n.
  - (n : Nat) -> (Mat n n)
- Multiplicação de Matriz – verificar se o tamanho é correto
  - **∀** {i j k} → (Mat i j) **->** (Mat j k) **->** (Mat i k)

PROVAS MATEMÁTICAS

- Agda serve para provar teoremas matemáticos, baseado no paradigma de proposições como tipos, escritos de forma funcional.

- module Example where
- open import Data.Nat
- open import Relation.Binary.PropositionalEquality
- prf : ∀ n m → (n + m) + n ≡ m + (n + n)
- prf n m = ?