Polymorphic `list` can be thought of as what?

How are polymorphic types written?

What are the types of the *polymorphic constructors* `nil` and `cons`?

How should `forall X` be read in the type of a polymorphic constructor?

With respect to polymorphic inductive types, Coq will automatically infer what?

What won't Coq automatically infer?

When no inference has been requested, where do you write type argument in a constructor invocation?

How about in a constructor that is a case of a match on a polymorphic datatype?

Underscores can replace what kinds of type information?

How does Coq determine what to fill in for the implicit argument?

How can type argument inference be requested by a user?

How can type argument inference be requested at definition site?

This does not work on ...

What is the limit of implicit arguments?

What's the difference between `(x,y)` and `(X*Y)`?

What is `type_scope`?

Matching on multiple arguments is actually what?

Which way does the type arrow associate?

What is the partial application syntax?

What are the types of the currying and uncurrying a function?

Give the syntax of anonymous functions.

What is the type of the `override` function?

Describe the `unfold` and `fold` tactics.

What does it mean to say that constructors in Coq are *disjoint*?

What does it mean to say that constructors in Coq are *injective*?

Describe the `inversion` tactic.

`inversion` can act on what?

How can tactics be applied to hypothesis instead of the goal?

Describe the difference between `apply` and `apply...in`.

Explain the difference between *forward* and *backward* reasoning.

What happens if you `destruct` on an expression instead of a value?

Explain the `remember` tactic.

What is a common use-case for `remember`?

Explain the `apply ... with ...` tactic.