What's the difference between declarations and typing rules?

Declarations may give the type of an identifier, but typing rules establish how expressions of various types are composed.

Functions in Coq are as functions in ...

Define them.

... computer science, not set theory.

They are algorithms mapping values of one type to values of another type.

What is computation to Coq?

Reducing terms until they cannot be further reduced.

Why isn't Coq Turing-complete?

The *strong normalization* property is too strong. Any system that can describe all computable functions must also admit functions that don't halt. Since the strong normalization property guarantees all Coq functions halt, the ability to compute some computable functions is necessarily left out.

In Coq types are ... hence they have ...

... terms ... a type.

What's the type of a proposition?

How might one make a predicate?

`Prop`

Using an arrow type ending in `Prop`, with the earlier types being the types of the variables in the proposition.

What is the relationship between propositions and programs in Coq?

Propositions can refer to programs (e.g., the `Prop` type), and programs can refer to propositions (*dependent types*).

What is the comments syntax?

As in OCaml:

```
(* comment *)
```