Where does Coq define booleans and numbers?

What's a "type" in Coq?

Name three ways to check that a function works.

How might you create "unit tests"?

Apply negation to the boolean `true` and evaluate.

How do you fill in a hole in a `Definition`/`Fixpoint`?

In the proof of a `Theorem`?

How does Coq write the type of a boolean conjunction function?

What does the `Check` command do?

How will we use the module system?

What is an enumerated type?

When we use `Inductive` to define a type, we should see it as what?

What "magic" does Coq provide for natural numbers?

What is the fundamental difference between a data constructor and functions?

Name some keywords that can introduce a function.

What kind of recursion does Coq allow?

What notational convenience does Coq provide for multiple parameters of the same type?

How does one match on *multiple* expressions?

What is an underscore in the context of `match` expressions?

How is "language support" introduced for some definitions?

Name three kinds of language support available.

How can one choose between multiple notation interpretations for an expression.

Name a few notation scopes.

What two tactics simplify the goal by performing computation?

The `reflexivity` tactic implicitly does what?

What's the difference between the simplification of `simpl` and that of `reflexivity`?

Why doesn't `simpl`'s implicit simplification unfold definitions?

What keywords behave identically to `Theorem`?

What is the "context"?

What does the `intros` tactic do?

What is the syntax of `intros`?

Some of the simple `intros` examples don't actually require `intros`. Give two reasons you might actually need it in a proof.

Describe the `rewrite` tactic.

What does rewriting *left-to-write* mean?

How are are propositions with multiple hypothesis written?

Why can't simple calculation prove every theorem?

Describe the `destruct` tactic.

Why don't we say `destruct b as [true | false].`?

`destruct` is used to prove a theorem about an enumerated type for each possible ...

What do `Case`/`SCase`/etc do?

Don't confuse `Case` with ...

What is the syntax of the `induction` tactic?

What do context items like `IHn'` stand for?

Name some fundamental facts concerning our definition of `plus` comes up over and over?

How can you create sub-theorems without creating a new top-level name?

What is a common non-stylistic reason for using `assert`?

Describe the `replace` tactic.

When is it often used?