Coq quick reference

Coq 8.2pl2

Starting the system

`coqtop` or `coqide`
use `coqc` to compile a file

Quitting the system

Quit.

Starting a proof

```
Lemma the_name :  forall x y, x < x + y.
```

Finishing a proof

```
Abort.
Qed.
```

Libraries

Loading libraries
`Require Import ...`
Important libraries
`Arith`, `ZArith`, `List`, `Wellfounded`, `Reals`

Gathering information

```
Locate "_ /\ _''.
SearchAbout name.
Search name.
SearchPattern (_ <= _ * _).
SearchRewrite (_ * ( _ + _)).
```

Defining a datatype

```
Inductive tree (A : Type) : Type :=
 L (a : A) | N (t1 t2 : tree A).
```

Programming

```
Fixpoint ev (n:nat) : bool :=
match n with
| 0 => true
| 1 => false
| S (S p) => ev p
end.

Fixpoint revt A (l1 l2 : list A) :=
match l1 with
| nil => l2
| a::tl => revt A tl (a::l2)
end.

Definition rev' A l := revt A l nil.
```

Computing values

```
Eval vm_compute in ev 3.
```

Basic tactics

Simple goals

`trivial`, `assumption`, `exact H`

Adding intermediate facts

`assert (H : formula)`,
`assert (H := theorem a b)`

universal quantification `forall x:T, A` (conclusion)
implication `A -> B` (conclusion)

`intros a b c`

universal quantification (hypothesis)
implication (hypothesis)

`apply H`

equality `A = B` (conclusion)

`reflexivity`, `trivial`, `ring`

equality, universally quantified equality (hypothesis)

`rewrite H`

equality between constructors

`injection H`, `discriminate`

conjunction, `A /\ B` (conclusion)

`split`

conjunction (hypothesis)

`destruct H as [H1 H2]`

disjunction, `A \/ B` (conclusion)

`left`, `right`

disjunction (hypothesis)

`destruct H as [H1 | H2]`

existential quantification
`exists a:T, P a` (conclusion)

`exists e`

existential quantification (hypothesis)

`destruct H as [x Px]`

negation `~A` (conclusion)

`intros H`

negation (hypothesis)

`case H`

case analysis

`destruct (f x) as [v1 | v2 v3]`
`case_eq (f x)`, `case (f x)`

computation and replacement

`unfold f`,`fold a`, `simpl`, `simpl f`, `simpl (f x)`
`change (f x) with (g x)`, `replace (f x) with (g x)`

Proof by induction

`elim x`, `elim H`, `induction x`,
`functional induction f x`

Automatic proof tactics

`tauto`, `firstorder`, `omega`, `ring`