Inductive seq : nat -> Set :=

| niln : seq 0

| consn : forall n : nat, nat -> seq n -> seq (S n).

Fixpoint length (n : nat) (s : seq n) {struct s} : nat :=

match s with

| niln => 0

| consn i \_ s' => S (length i s')

end.

Theorem length\_corr : forall (n : nat) (s : seq n), length n s = n.

Proof.

intros n s.

(\* reasoning by induction over s. Then, we have two new goals

corresponding on the case analysis about s (either it is

niln or some consn \*)

induction s.

(\* We are in the case where s is void. We can reduce the

term: length 0 niln \*)

simpl.

(\* We obtain the goal 0 = 0. \*)

trivial.

(\* now, we treat the case s = consn n e s with induction

hypothesis IHs \*)

simpl.

(\* The induction hypothesis has type length n s = n.

So we can use it to perform some rewriting in the goal: \*)

rewrite IHs.

(\* Now the goal is the trivial equality: S n = S n \*)

trivial.

(\* Now all sub cases are closed, we perform the ultimate

step: typing the term built using tactics and save it as

a witness of the theorem. \*)

Qed.