```
(*  apl-2013-06-19.v
 *
 * Programmer: Mayer Goldgerg, 2013
 *)
Require Import Setoid.
Axiom PNP: ∀ p : Prop, p ∨ ¬p.
Lemma L1:
  ∀ (p q : Prop), p → (p ∨ q).
Proof.
  intro p.
  intro q.
  intro H.
  left.
  exact H.
Qed.
Lemma L2:
  ∀ (p q : Prop), q → (p ∨ q).
Proof.
  intro p.
  intro q.
  intro H.
  right.
  exact H.
Qed.
Lemma L3:
  ∀ (p q : Prop), p → q → p.
Proof.
  intro p.
  intro q.
  intro H.
  intro Q.
  exact H.
Qed.
Lemma L4_too_long:
  ∀ (p q : Prop), (p ∨ q) ↔ (q ∨ p).
Proof.
  intro p.
  intro q.
  split.
  (*  part 1: p \/ q -> q \/ p *)
  intro H.
  destruct H as [H1 | H2].
  right.
  exact H1.
  left.
  exact H2.
  (*  part 2: q \/ p -> p \/ q *)
  intro Q.
  destruct Q as [Q1 | Q2].
  right.
  exact Q1.
  left.
  exact Q2.
Qed.
Lemma L4_one_way:
  ∀ (p q : Prop), (p ∨ q) → (q ∨ p).
Proof.
  intro p.
  intro q.
  intro H.
  destruct H as [H1 | H2].
  right.
  exact H1.
  left.
  exact H2.
Qed.
Lemma L4:
  ∀ (p q : Prop), (p ∨ q) ↔ (q ∨ p).
Proof.
  intro p.
  intro q.
  split.
  apply L4_one_way.
  apply L4_one_way.
Qed.
Lemma L5:
  ∀ (p q : Prop), (p ∧ q) ↔ (q ∧ p).
Proof.
  admit.
Qed.
Lemma L7:
  ∀ (p : Prop), p ↔ (p ∨ p).
```

```
Proof.
  admit.
Qed.

Lemma L8:
  ∀ (p : Prop), p ↔ (p ∧ p).
Proof.
  admit.
Qed.

Lemma L9:
  ∀ a b c : Prop, ((a ∧ b) → c) ↔ (a → b → c).
Proof.
  admit.
Qed.

Lemma L10:
  ∀ p : Prop, p ↔ ¬ ¬p.
Proof.
  intro p.
  split.

  (*  part 1: p -> ~ ~ p *)
  intro H.
  unfold not.
  intro Q.
  apply (Q H).

  (*  part 2: ~ ~ p -> p *)
  unfold not.
  intro H.
  destruct (PNP p) as [H1 | H2] in H.
  exact H1.
  unfold not in H2.
  apply H in H2.
  contradiction.
Qed.
```