1.

```
(** **** Exercise: 2 stars, standard (ev_sum) *)
Theorem ev_sum : forall n m, ev n -> ev m -> ev (n + m).
Proof.
  intros. induction H as [ | n' H' IH].
  - simpl. apply H0.
  - simpl. apply ev_SS. apply IH.
  Qed.
```

```
(** **** Exercise: 2 stars, standard (ev_sum) *)
Theorem ev_sum : forall n m, ev n -> ev m -> ev (n + m).
Proof.
  intros. induction H as [ | n' H' IH].
  - simpl. apply H0.
  - simpl. apply ev_SS. apply IH.
  Qed.
```

2.

```
(** **** Exercise: 5 stars, standard, optional (le_and_lt_facts) *)
Lemma le_trans : forall m n o, m <= n -> n <= o -> m <= o.
Proof.
  intros. induction H0 as [ | n' H' IH].
  - apply H.
  - apply le_S. apply IH.
Qed.
```

```
Lemma le_trans : forall m n o, m <= n -> n <= o -> m <= o.
Proof.
  intros. induction H0 as [ | n' H' IH].
  - apply H.
  - apply le_S. apply IH.
Qed.
```

3.

```
Theorem App_ : forall n m o p : nat, [n;m;o;p] = [n;m]++[o;p].
Proof. intros. reflexivity. Qed.

Example reg_exp_ex3 : [0;1;0;1] =~ Star (App (Char 0) (Char 1)).
Proof. rewrite -> App_.
apply MStarApp.
* apply (MApp [0] _ [1]).
  + apply MChar.
  + apply MChar.
* rewrite <- (app_nil_r _ [0;1]).
  apply MStarApp.
  + apply (MApp [0] _ [1]).
    ++ apply MChar.
    ++ apply MChar.
  + apply MStar0.
Qed.
```


```
Theorem App_ : forall n m o p : nat, [n;m;o;p] = [n;m]++[o;p].
Proof. intros. reflexivity. Qed.

Example reg_exp_ex3 : [0;1;0;1] =~ Star (App (Char 0) (Char 1)).
Proof. rewrite -> App_.
apply MStarApp.
* apply (MApp [0] _ [1]).
  + apply MChar.
  + apply MChar.
* rewrite <- (app_nil_r _ [0;1]).
  apply MStarApp.
  + apply (MApp [0] _ [1]).
    ++ apply MChar.
    ++ apply MChar.
  + apply MStar0.
Qed.
```


4.

```
Lemma MUnion' : forall T (s : list T) (re1 re2 : reg_exp T),
  s =~ re1 \/ s =~ re2 ->
  s =~ Union re1 re2.
Proof.
  intros. destruct H.
  - apply MUnionL. apply H.
  - apply MUnionR. apply H.
Qed.
```


```
Lemma MUnion' : forall T (s : list T) (re1 re2 : reg_exp T),
  s =~ re1 \/ s =~ re2 ->
  s =~ Union re1 re2.
Proof.
  intros. destruct H.
  - apply MUnionL. apply H.
  - apply MUnionR. apply H.
Qed.
```