1. Since the [optimize_0plus] transformation doesn't change the value of [aexp]s, we should be able to apply it to all the [aexp]s that appear in a [bexp] without changing the [bexp]'s value. Write a function that performs this transformation on [bexp]s and prove it is sound. Use the tacticals we've just seen to make the proof as short and elegant as possible.

代码

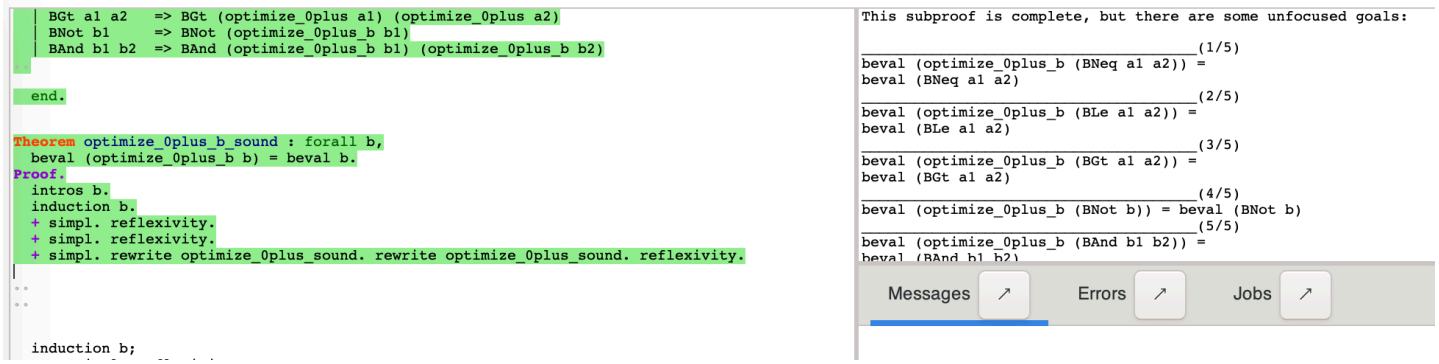
```
Fixpoint optimize_0plus_b (b : bexp) : bexp :=
  match b with
  | BTrue       => b
  | BFalse      => b
  | BEq a1 a2   => BEq (optimize_0plus a1) (optimize_0plus a2)
  | BNeq a1 a2 => BNeq (optimize_0plus a1) (optimize_0plus a2)
  | BLe a1 a2   => BLe (optimize_0plus a1) (optimize_0plus a2)
  | BGt a1 a2   => BGt (optimize_0plus a1) (optimize_0plus a2)
  | BNot b1     => BNot (optimize_0plus_b b1)
  | BAnd b1 b2  => BAnd (optimize_0plus_b b1) (optimize_0plus_b b2)
  end.

Theorem optimize_0plus_b_sound : forall b,
  beval (optimize_0plus_b b) = beval b.
Proof.
  intros b.
  induction b;
  try (simpl; reflexivity);
  try (simpl; repeat rewrite optimize_0plus_sound; reflexivity).
  + simpl. rewrite IHb. reflexivity.
  + simpl. rewrite IHb1. rewrite IHb2. reflexivity.
Qed.
```

分析

提示的意思就是，由于optimize_0plus转换不会改变aexps的值，因此应该能够将其应用于bexp中出现的所有aexps，而无需更改bexp的值。编写一个在bexps上执行此转换的函数，并证明它是正确的。使用我们刚刚看到的tacitics，使证明尽可能简洁优雅。

```
Inductive bexp : Type :=
  | BTrue
  | BFalse
  | BEq (a1 a2 : aexp)
  | BNeq (a1 a2 : aexp)
  | BLe (a1 a2 : aexp)
  | BGt (a1 a2 : aexp)
  | BNot (b : bexp)
  | BAnd (b1 b2 : bexp).
```

对于`optimize_0plus_b`，按照bexp的类型定义“依葫芦画瓢“就可以了。



对于`optimize_0plus_b_sound`，`intros`和`induction`之后，根据观察可以把需要证明的8个分支分为3类。1-2直接`simpl. reflexivity.`即可，第二种，根据试验，只需要`simpl.`之后不同`rewrite optimize_0plus_sound.`，最后再`reflexivity`。最后的`BNot`和`BAnd`则需要用一些假设去`rewrite`。因为要求证明要尽可能的"elegant"，所以我们使用`try`和`repeat`。具体如上面的代码。

运行结果

```
Fixpoint optimize_0plus_b (b : bexp) : bexp :=
  match b with
  | BTrue       => b
  | BFalse      => b
  | BEq a1 a2   => BEq (optimize_0plus a1) (optimize_0plus a2)
  | BNeq a1 a2 => BNeq (optimize_0plus a1) (optimize_0plus a2)
  | BLe a1 a2   => BLe (optimize_0plus a1) (optimize_0plus a2)
  | BGt a1 a2   => BGt (optimize_0plus a1) (optimize_0plus a2)
  | BNot b1     => BNot (optimize_0plus_b b1)
  | BAnd b1 b2  => BAnd (optimize_0plus_b b1) (optimize_0plus_b b2)
  end.

Theorem optimize_0plus_b_sound : forall b,
  beval (optimize_0plus_b b) = beval b.
Proof.
  intros b.
  induction b;
  try (simpl; reflexivity);
  try (simpl; repeat rewrite optimize_0plus_sound; reflexivity).
  + simpl. rewrite IHb. reflexivity.
  + simpl. rewrite IHb1. rewrite IHb2. reflexivity.
Qed.
```