Mini-Test 1

Name: MANSI SHARMA Email: Mansi.Sharma001@umb.edu

1. (50 points) Check the correct answer.
   (a) (10 points) The type `nat` has a finite number of elements.
   - ○ True
   - ✓ False

   (b) (10 points) Suppose we have assumption `H : P → Q → R → S` and the current goal is `S`. If we do **`apply`** `H`, then the goal will change to `P → Q → R`.
   - ○ True
   - ✓ False

   (c) (10 points) In Coq, the proposition `True` and the boolean `true` are logically equivalent, *i.e.*, one can prove `True ↔ true`.
   - ○ True
   - ✓ False

   (d) (10 points) If `H : x1 :: y1 = x2 :: y2` is a current assumption, then we know that `x1` is equal to `x2`.
   - ✓ True
   - ○ False

   (e) (10 points) All types defined in Coq must be nonempty. In other words, for any type `A`, there is some Coq expression that has type `A`.
   - ○ True
   - ✓ False

2. (10 points) Give the type of each of the following Coq expressions, or write "ill typed" if an expression does not have a type.
   (a) (5 points) **`fun`** `(b : bool)` ⇒ **`if`** `true` **`then`** `56` **`else`** `b`

   ill typed

   (b) (5 points) **`fun`** `(x y : nat)` ⇒ `x + 56 = y`

   nat → nat → Prop

3. (20 points) For each of the following propositions, check "not provable" if it is not provable (in Coq's core logic, without additional axioms), "induction" if it is provable only using induction, or "easy" if it is provable without using induction and without additional lemmas.
   (a) (4 points) **`forall`** `(A:Type) (l l' : list A), length (l++l') = length l + length l'`
   - ○ Easy
   - ✓ Induction
   - ○ Not Provable

   (b) (4 points) **`forall`** `n, n = S n`
   - ○ Easy

○ Induction
✓ ○ Not Provable

(c) (4 points) **forall** {A:**Type**}, length l = 0 → l = []

○ Easy
✓ ○ Induction
○ Not Provable

(d) (4 points) In 3 [1;2;3;4;5]

✓ ○ Easy
○ Induction
○ Not Provable

(e) (4 points) **forall** P : **Prop**, P \/ ~P

○ Easy
○ Induction
✓ ○ Not Provable

4. (20 points) Complete each proof. Your proof cannot use **auto** nor **intuition**.

(a)
```
P, Q : Prop
H : P \/ Q
H0 : ~ Q
_____________________________________(1/1)
P
```

```
destruct H. 2
    apply H.
y
unfold not in H0.
destruct H0.
apply H.
```

(b)
```
P : Prop
H : P
_____________________________________(1/1)
~ ~ P
```

```
unfold not.
Contradiction.
```

(c) **forall** n : nat, True

```
Proof.
        intros.
        apply I.
Qed.
```

(d) **forall** (A:**Type**) (x:A), [x] = [x].

```
Proof.
        intros.
        assumption.
Qed.
```