CIS 500 — Software Foundations

Midterm I

February 15, 2012

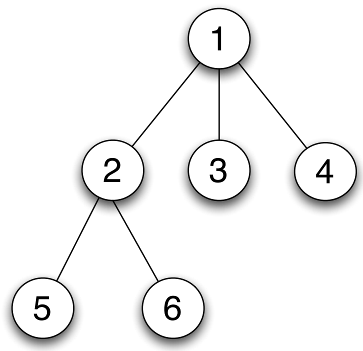
Name: ______________________________

Pennkey: ______________________________

Scores:

1	
2	
3	
4	
5	
6	
7	
8	
9	
Total (80 max)	

1. (8 points) A *2-3 tree* is a tree data structure in which (1) every node is labeled with a value (drawn from some set `X`), and (2) every node has zero, two, or three children. For example, here is a 2-3 tree of numbers:



(a) Complete the following inductive definition of 2-3 trees:

```
Inductive ttree {X : Type} : Type :=
```

(b) Write down a term of type `ttree nat` representing the tree shown above.

2. (6 points) Briefly explain the behavior of the `apply` and `apply... with...` tactics in Coq.

3. (6 points) For each of the following types, define a function (using `Definition` or `Fixpoint`) with the given type.

(a) `nat -> list (list nat)`

(b) `forall X Y : Type, list X -> (X -> Y) -> list Y`

4. (8 points) Write down the type of each of the following expressions. (For example, for the expression

```
fun (x y : nat) => beq_nat (x+y) 10
```

you'd write `nat -> nat -> bool`.) If an expression is not typeable, write "ill typed."

(a) `fun (x : nat) => x :: []`

(b) `(2 :: 3 :: []) :: []`

(c)
```
fun (X : Type) (l : list X) =>
  match l with
    [] => []
  | h :: t => h
  end
```

(d)
```
fun (X Y Z : Type) (f : X->Y) (g : Y->Z) (a : X) =>
  g (f a)
```

5. (12 points) In this question, we'll consider two different implementations of the same transformation on lists — one as an inductively defined relation and one as a `Fixpoint`.

(a) The relation `rdrop` is a three-place relation that holds between a number `n`, a list `xs`, and a list `xs'` if and only if `xs'` is the list obtained by dropping the first `n` elements of `xs`. For example, the following are all provable instances of `rdrop`.

```
rdrop 3 [1,2,3,4,5] [4,5]
rdrop 2 [5,4,3,2,1] [3,2,1]
rdrop 5 [1,2,3] []
```

Complete the following definition of `rdrop`.

```
Inductive rdrop {X : Type} : nat -> list X -> list X -> Prop :=
```

(b) Similarly, `fdrop` is a *function* that takes a number `n` and a list `xs` and returns the list consisting of all except the first `n` the elements of `xs`. For example:

```
fdrop 3 [1,2,3,4,5] = [4,5].
fdrop 2 [5,4,3,2,1] = [3,2,1].
fdrop 5 [1,2,3] = []
```

Complete the following `Fixpoint` definition of `fdrop`.

```
Fixpoint fdrop {X : Type} (n : nat) (xs : list X) : list X :=
```

6. (20 points) Recall the definition of `beq_nat`:

```
Fixpoint beq_nat (n m : nat) : bool :=
  match n with
  | O => match m with
         | O => true
         | S m' => false
         end
  | S n' => match m with
            | O => false
            | S m' => beq_nat n' m'
            end
  end.
```

Write out a careful informal proof of the following theorem, using the pedantic “template” style discussed in the notes. Make sure to state the induction hypothesis explicitly.

Theorem: For all natural numbers `n` and `m`, if `beq_nat n m = true` then `n = m`.

Proof:

7. (10 points) Recall the inductive definitions of logical conjunction and the property `beautiful`:

```
Inductive and (P Q : Prop) : Prop :=
  conj : P -> Q -> (and P Q).

Notation "P /\ Q" := (and P Q) : type_scope.

Inductive beautiful : nat -> Prop :=
  b_0   : beautiful 0
| b_3   : beautiful 3
| b_5   : beautiful 5
| b_sum : forall n m, beautiful n -> beautiful m -> beautiful (n+m).
```

Suppose we have already proved the following theorem:

```
Theorem b1000: beautiful 1000.
```

Give a proof object for the following proposition. Show all parts of the proof object explicitly (i.e., do not use _ anywhere).

```
Definition b_facts : forall x,
                       beautiful x ->
                       (beautiful (1000 + x) /\ beautiful 3) :=
```

8. (2 points) How many different proof objects are there for the proposition in the previous question?

9. (8 points) Recall the definition of existential quantification:

```
Inductive ex (X:Type) (P : X->Prop) : Prop :=
  ex_intro : forall (witness:X), P witness -> ex X P.
```

(a) Write a proposition capturing the claim “there is some number whose successor is beautiful.”

(b) Give a proof object for this proposition.