CS_336
INTERACTIVE THEOREM PROVING
(*Attempt* **2** *questions out of 3*)

Question 1.

(a) Define the notion of a *reduction system*. What does it mean for a reduction system to be *weakly normalising*? What does it mean for it to be *confluent*?

[6 marks]

(b) Determine a reduction system which is not weakly-normalising and non-confluent, and which uses as few terms and as few reductions as possible. Explain why it is non-confluent and not weakly-normalising.

[6 marks]

(c) Derive, using the rules for introducing terms of the simply typed λ-calculus, that

$$(\lambda f : \mathrm{o} \to \mathrm{o}.\lambda x : \mathrm{o}.f\,(f\,x)) : (\mathrm{o} \to \mathrm{o}) \to \mathrm{o} \to \mathrm{o}$$

[6 marks]

(d) Show that it is not possible to assign a simple type to $\lambda x.x\,x$, i.e. that there exist no types σ, τ such that $(\lambda x : \sigma.x\,x) : \tau$.

[7 marks]

Question 2.

(a) Describe briefly two examples of how dependent types could be used in general programming.

[4 marks]

(b) There are two ways of introducing the product of two sets `A::Set` and `B::Set` in Agda. Introduce both of them.

[6 marks]

(c) Let `AB` be one of the products of the sets `A` and `B`, as introduced in (b). Solve the following goal in Agda:

```
f  (c_a::C → A)
   (c_b::C → B)
   (c::C)
   ::AB
   = {!  !}
```

[6 marks]

(d) Assuming `X,Y,Z::Set`, introduce in Agda a function `f` such that

```
f::((X → Z) → Z) → (X → Y) → (Y → Z) → Z
```

Hint: When we apply intro and then introduce one let-expression of type `X → Z`, we obtain the following goals:

```
f  ::((X → Z) → Z) → (X → Y) → (Y → Z) → Z
   = λ(x_z_z::(X → Z) → Z)→
     λ(x_y::X → Y) →
     λ(y_z::Y → Z) →
     let x_z::X → Z
         = λ(x::X) → {!  !}
     in {!  !}
```

Solve the goals in this expression.

[9 marks]

Question 3.

(a) Introduce in Agda the sets `Bool::Set` of Booleans, the empty set `False::Set`, the set `True::Set` containing one element, and the operation `atom::Bool → Set`, which maps a Boolean value to the formula corresponding to its truth-value.

[5 marks]

(b) Introduce in Agda the disjoint union `X + Y::Set` of two sets `X,Y::Set`.

[2 marks]

In (c) - (f) assume

```
A::Set
B::Set
IdABool::A → A → Bool
IdBBool::B → B → Bool
```

The intended meaning is that `IdABool` is a Boolean valued equality on `A` and `IdBBool` is a Boolean valued equality on `B`.

(c) Define in Agda the set theoretic equality

```
IdA::A → A → Set
```

corresponding to `IdABool`.

[2 marks]

(d) Introduce in Agda a set expressing "`IdA` is reflexive".

[3 marks]

(e) Assume that the equality `IdB` on `B` is defined as `IdA`. Introduce in Agda an equality set `IdAB` on `A + B`.

[6 marks]

(f) Show in Agda that `IdAB` it is reflexive, provided that `IdA` and `IdB` are reflexive (where the reflexivity of `IdB` is defined as for `IdA`).

[7 marks]