Comp Sci 538, Written Assignment 2

1 Types, Terms, and Contexts

1.1 Fill in the Types

(a) $\Gamma = \{x : Bool, y : Bool\}$; $t = Bool$
(b) $\Gamma = \{x : Bool \to Bool, y : Bool\}$; $t = Bool$
(c) $\Gamma = \cdot$; $t = Bool \to Bool$
(d) $\Gamma = \{y : (Bool \to Bool) \to Bool \to Bool\}$; $t = (Bool \to Bool) \to Bool$
(e) $\Gamma = \{y : Bool \to Bool\}$; $t = (Bool \to Bool) \to Bool$
(f) $\Gamma = \{y : Bool\}$; $t = Bool \to Bool$
(g) $\Gamma = \cdot; t = Bool \to Bool \to Bool \to Bool$

1.2 Fill in the Terms

(a) $e = \lambda x : Bool.\ x$
(b) $e = \lambda x : Bool.\ \lambda y : Bool.\ x$
(c) $e = \lambda f : Bool \to Bool.\ true$
(d) $e = \lambda f : Bool \to Bool \to Bool.\ true$
(e) $e = y\ x$

2 Inhabitants of a Type

2.1 How Many Programs?

(a) $true/false/$; 2 in total
(b) $\lambda x : Bool.\ true/\ \lambda x : Bool.\ false/\lambda x : Bool.\ if\ x\ then\ true\ else\ false\ /\ \lambda x : Bool.\ if\ x\ then\ false\ else\ true$; 4 in total

2.2 Program as Proofs

(a) $e = \lambda x : P.\ x$
(b) $e = \lambda x : P \times Q.\ fst(e)$
(c) $e = \lambda x : P \times Q.\ (snd(x), fst(x))$
(d) $e = \lambda x : P.\ \ left(x)$
(e) $e = \lambda x : P \times (P \to Q).\ \ snd(x)\ fst(x)$
(f) $e = \lambda x : P.\ \lambda y : Q.\ \ fst(x, y)$
(g) $e = \lambda x : (P \to Q) \times (Q \to R).\ \ snd(x)\ .\ fst(x)$

2.3 Are there programs of the following type?

(a) No. We can't get a value of type Q from type P in a closed program.
(b) No. For a sum type, it is not guaranteed that the type we get is of type P instead of Q.
(c) No. Given a function that goes from P to Q, we still need to pass in an argument with value of type P to get a value of type Q.

3 Adding Triples

3.1 Projections

(a) $e ::= \ldots | (e_1, e_2, e_3) | \ fst(e) \mid snd(e) \mid thd(e)$
(b)

$$\frac{\Gamma \vdash e_1 : t_1,\ \Gamma \vdash e_2 : t_2,\ \Gamma \vdash e_3 : t_3}{\Gamma \vdash (e_1, e_2, e_3) : (t_1, t_2, t_3)} \qquad \frac{\Gamma \vdash e : (t_1, t_2, t_3)}{fst(e) : t_1}$$

$$\frac{\Gamma \vdash e : (t_1, t_2, t_3)}{snd(e) : t_2} \qquad \frac{\Gamma \vdash e : (t_1, t_2, t_3)}{thd(e) : t_3}$$

(c)

$$\frac{e \to e'}{fst(e) \to fst(e')} \qquad \frac{}{fst((v_1, v_2, v_3)) \to v_1}$$

$$\frac{e \to e'}{snd(e) \to fst(e')} \qquad \frac{}{snd((v_1, v_2, v_3)) \to v_2}$$

$$\frac{e \to e'}{thd(e) \to fst(e')} \qquad \frac{}{thd((v_1, v_2, v_3)) \to v_3}$$

3.2 Pattern matching

(a) case e of $(x, y, z) \to e_1$
(b)

$$\frac{\Gamma, x : t_1, y : t_2, z : t_3, \vdash e_1 : t \quad \Gamma \vdash e : (t_1.t_2, t_3)}{\Gamma \vdash \text{case } e \text{ of } (x, y, z) \to e_1 : t}$$

(c)

$$\frac{e \to e'}{(\text{case } e \text{ of } (x, y, z) \to e_1) \to \text{case } e' \text{ of } (x, y, z) \to e_1}$$

$$\frac{}{(\text{case } (a, b, c) \text{ of } (x, y, z) \to e_1) \to e_1[x \vdash a,\ y \vdash b,\ z \vdash c]}$$

3.3 Matching deeper

Pattern Matching:
case e of $(a, b, (a', b', c')) \to e_1$
Normal Eliminator:
For the parts that are not nested, say the first and second elements in the example, we can simply project them out with *fst* and *snd* functions as we defined above. For the nested parts, we can gain access to them by functions like $thdFst = fst\ .\ thd$, $thdSnd = snd\ .\ thd$, and $thdThd = thd\ .\ thd$.