Q1 Proof Trees
25 Points

Q1.1
5 Points

2*(3+4)

1. 2 * (3 + 4)

```
              -------Ty-Int   -------Ty-Int
               {}⊢ 3:ℤ       {}⊢ 4:ℤ
 -------Ty-Int    --------------Ty-Add
 {}⊢ 2:ℤ          {}⊢ (3 + 4):ℤ
 -------------------------------Ty-Mul
 {}⊢ 2 * (3 + 4) :ℤ
```

No files uploaded

Q1.2
5 Points

if (0>1) 2 3

2. if (0 > 1) 2 3

```
 ------Ty-Int  ------Ty-Int
 {}⊢0:ℤ       {}⊢1:ℤ
 --------Ty-GT   -----Ty-Int  ----Ty-Int
 {}⊢(0>1):𝔹     {}⊢2:ℤ       {}⊢3:ℤ
 --------------------Ty-If
 {}⊢if (0 > 1) 2 3 :ℤ
```

No files uploaded

Q1.3
5 Points

(((λx:ℤ.λy:ℤ.x+y) 8) 9)

3. (((λx:ℤ.λy:ℤ.x + y) 8) 9)

```
------------------Ty-Var  ------------------Ty-Var
{(x,ℤ),(y,ℤ)}⊢x:ℤ         {(x,ℤ),(y,ℤ)}⊢y:ℤ
-------------------------------------------Ty-Add
{(x,ℤ),(y,ℤ)}⊢(x+y):ℤ
-------------------------Ty-λ
{(x,ℤ)}⊢(λy:ℤ.x+y):ℤ→ℤ
--------------------Ty-λ   -------Ty-Int
{}⊢(λx:ℤ. λy:ℤ.x+y):ℤ→ℤ→ℤ   {}⊢8:ℤ
-------------------------------Ty-App  -------Ty-Int
{}⊢((λx:ℤ. λy:ℤ.x+y) 5):ℤ→ℤ            {}⊢9:ℤ
-----------------------------------------------Ty-App
{}⊢(((λx:ℤ. λy:ℤ.x+y) 8) 9) : ℤ
```

No files uploaded

Q1.4
5 Points

((λx:ℤ.x+1) 5)

4. ((λx:ℤ.x+1) 5)

```
-------------Ty-Var  -------------Ty-Int
{(x,ℤ)}⊢x:ℤ          {(x,ℤ)}⊢1:ℤ
-----------------Ty-Add
{(x,ℤ)}⊢(x+1): ℤ
------------------Ty-λ   -----------Ty-Int
{}⊢(λx:ℤ.x+1): ℤ→ℤ       {}⊢5:ℤ
---------------------Ty-App
{}⊢((λx:ℤ.x+1) 5) :ℤ
```

No files uploaded

Q1.5
5 Points

(head (nil ⟦B⟧))

5. (head (nil ⟦𝔹⟧))

----------Ty-Nil
{}⊢(nil ⟦𝔹⟧):⟦𝔹⟧
----------Ty-Head
{}⊢(head (nil ⟦𝔹⟧)):𝔹

No files uploaded

Q2 Type Errors
10 Points

Q2.1
5 Points

cons true (nil ⟦ℤ⟧)

1. cons true (nil ⟦ℤ⟧)

----------Ty-True ----------Ty-Nil
{}⊢true:𝔹 (nil ⟦ℤ⟧):⟦ℤ⟧
----------Ty-Cons
{}⊢cons true (nil ⟦ℤ⟧):<Failure> This use of Ty-Cons expects a ℤ in its left subterm, but we got a 𝔹.

No files uploaded

Q2.2
5 Points

((λx:ℤ.x>2) true)

2. ((λx:ℤ.x>2) true)

----------Ty-λ ----------Ty-true
{}⊢(λx:ℤ.x>2):ℤ→𝔹 {}⊢true:𝔹
----------Ty-App
{}⊢((λx:ℤ.x>2) true):<Failure> This use of Ty-App expects ℤ as its input, but we got a 𝔹

No files uploaded

Q3 Language Extensions
20 Points

Q3.1
8 Points

Typing rules for Maybe extension.

3.1 Maybe: t ::= … | just t | nothing T | isjust t | unjust t

$$\frac{\Gamma \vdash t : T}{\Gamma \vdash \text{just } t : \text{just } t}$$

$$\frac{}{\Gamma \vdash \text{nothing } T : T}$$

$$\frac{\Gamma \vdash t : \text{Maybe } T}{\Gamma \vdash \text{isjust } t : \mathbb{B}}$$

$$\frac{\Gamma \vdash t : \text{Maybe } T}{\Gamma \vdash \text{unjust } t : T}$$

No files uploaded

Q3.2
12 Points

Typing rules for Either extension.

3.2 Either: t ::= … | left t T | right T t | isleft t | isright t | getleft t | getright t

$$\frac{\Gamma \vdash t : Tb}{\Gamma \vdash \text{left } t\ Td : Tb}$$

$$\frac{\Gamma \vdash t : Tb}{\Gamma \vdash \text{right } Td\ t : Tb}$$

$$\frac{\Gamma \vdash t : \text{Either } T\ T}{\Gamma \vdash \text{isleft } t : \mathbb{B}}$$

$$\frac{\Gamma \vdash t : \text{Either } T\ T}{\Gamma \vdash \text{isright } t : \mathbb{B}}$$

$$\frac{\Gamma \vdash t : \text{Either } T\ T}{\Gamma \vdash \text{getleft } t : T}$$

$$\frac{\Gamma \vdash t : \text{Either } T\ T}{\Gamma \vdash \text{getright } t : T}$$

No files uploaded

Q4 Encodings
20 Points

Q4.1
2 Points

nand =

1. nand = λa:𝔹. λb:𝔹. (a (not b)) not a

No files uploaded

Q4.2
2 Points

thrice =

2. thrice = (λf:ℤ→ℤ. λx:ℤ.f (f (f x)))

No files uploaded

Q4.3
2 Points

cubed =

3. cubed = λx:ℤ. x * x * x

No files uploaded

Q4.4
2 Points

modify =

4. modify = λx:Either ℤ 𝔹. if (isleft x) (left (x+1) 𝔹) (right ℤ (not x))

No files uploaded

Q4.5
2 Points

safeAdd =

5. safeAdd = λx:Maybe ℤ. λy:Maybe ℤ. safeAddHelper x + safeAddHelper y

safeAddHelper = λx:Maybe ℤ. if (isjust x) x 0

No files uploaded

Q4.6
2 Points

all =

```
6. all = fix ( λself:⟦𝔹⟧→𝔹.
                λxs:⟦𝔹⟧.
                   if (isnil xs) true ((head xs) and (self (tail xs)))
          )
```

No files uploaded

Q4.7
2 Points

sumlist =

```
7. sumlist = fix ( λself:⟦ℤ⟧→ℤ.
                    λxs:⟦ℤ⟧.
                       if (isnil xs) 0 ((head xs) + (self (tail xs)))
```

No files uploaded

Q4.8
2 Points

takeN =

```
8. takeN = fix ( λself:⟦ℤ⟧→ℤ→⟦ℤ⟧→ℤ.
                 λxs:⟦ℤ⟧.
                    λn:ℤ.
                       λys:⟦ℤ⟧.
                          if (n=0) ys (self (tail xs) (n-1) (cons ys (head xs)))
          )
```

No files uploaded

Q4.9
2 Points

filter =

```
9. filter = fix ( λself:(ℤ→𝔹)→[[ℤ]]→[[ℤ]].
                (
                   λf:ℤ→ℤ.
                      λxs:[[ℤ]].
                         if (isnil xs) (nil [[ℤ]])
                            if (f (head xs)) (cons (head xs) (self f (tail xs)))
(self f (tail xs))
                )
             )
```

No files uploaded

Q4.10
2 Points

zipwith =

```
10. zipwith = fix ( λself:(ℤ→ℤ→ℤ)→[[ℤ]]→[[ℤ]]→[[ℤ]].
                (
                   λf:ℤ→ℤ→ℤ.
                      λxs:[[ℤ]].
                         λys:[[ℤ]].
                               if (isnil xs)
                                  (ys)
                                  (cons (f (head xs) (head ys)) (self f (tail xs)
(tail ys)))
                )
             )
```

No files uploaded

Q5 Extra Credit!
5 Points

maxlist =

No files uploaded

Homework 6 - STLC

● Graded

Student
Isaac Kim

Total Points
69.5 / 80 pts

Question 1		
Proof Trees		25 / 25 pts
1.1	(no title)	5 / 5 pts
1.2	(no title)	5 / 5 pts
1.3	(no title)	5 / 5 pts
1.4	(no title)	5 / 5 pts
1.5	(no title)	5 / 5 pts
Question 2		
Type Errors		8.5 / 10 pts
2.1	(no title)	5 / 5 pts
2.2	(no title)	3.5 / 5 pts
Question 3		
Language Extensions		16 / 20 pts
3.1	(no title)	6 / 8 pts
3.2	(no title)	10 / 12 pts
Question 4		
Encodings		20 / 20 pts
4.1	(no title)	2 / 2 pts
4.2	(no title)	2 / 2 pts
4.3	(no title)	2 / 2 pts
4.4	(no title)	2 / 2 pts
4.5	(no title)	2 / 2 pts

4.6	(no title)	2 / 2 pts
4.7	(no title)	2 / 2 pts
4.8	(no title)	2 / 2 pts
4.9	(no title)	2 / 2 pts
4.10	(no title)	2 / 2 pts

Question 5
Extra Credit! 0 / 5 pts