((lambda (f x1) (f 1 x1)) + #t)

Rename bound variables: ((lambda (f x1) (f 1 x1)) + #t)

Assign type variables:

expression	variables
((lambda (f x1) (f 1 x1)) + #t)	T0
(lambda (f x) (f 1 x))	T1
(f 1 x)	T2
f	Tf
1	Tnum1
x	Tx
+	T+
#t	T#t

Construct type equations:

expression	equation
((lambda (f x1) (f 1 x1)) + #t)	T1 = [T+ * T#t -->T0]
(lambda (f x) (f 1 x))	T1 = [Tf * Tx --> T2]
(f 1 x)	Tf = [Tnum1 * Tx --> T2]
1	Tnum1 = Number
+	T+ = [Number * Number --> Number]
#t	T#t = Boolean

Solving equations:

	equation	substitution
1	T1 = [T+ * T#t -->T0]	
2	T1 = [Tf * Tx --> T2]	
3	Tf = [Tnum1 * Tx --> T2]	
4	Tnum1 = Number	
5	T+ = [Number * Number --> Number]	
6	T#t = Boolean	

(T1 = [T+ * T#t --> T0]) ○ Substitution = (T1 = [T+ * T#t --> T0])

	equation	substitution
2	T1 = [Tf * Tx --> T2]	{ T1 := [T+ * T#t --> T0] }
3	Tf = [Tnum1 * Tx --> T2]	
4	Tnum1 = Number	
5	T+ = [Number * Number --> Number]	
6	T#t = Boolean	

(T1 = [Tf * Tnum1 * Tx -->]) ○ Substitution = ([T+ * T#t --> T0] = [Tf * Tx --> T2])

	equation	substitution
3	Tf = [Tnum1 * Tx --> T2]	{ T1 := [T+ * T#t --> T0] }
4	Tnum1 = Number	
5	T+ = [Number * Number --> Number]	
6	T#t = Boolean	
7	Tf = T+	
8	Tx = T#t	
9	T2 = T0	

(Tf = [Tnum1 * Tx -->T2]) ○ Substitution = Substitution ○ (Tf = [Tnum1 * Tx ->T2])

	equation	substitution
4	Tnum1 = Number	{ T1 := [T+ * T#t --> T0], Tf := [Tnum1 * Tx --> T2] }
5	T+ = [Number * Number --> Number]	
6	T#t = Boolean	
7	Tf = T+	
8	Tx = T#t	
9	T2 = T0	

(Tnum1 = Number) ○ Substitution = Substitution ○ (Tnum1 = Number)

	equation	substitution
5	T+ = [Number * Number --> Number]	{ T1 := [T+ * T#t --> T0], Tf := [Number * Tx --> T2], Tnum1 = Number }
6	T#t = Boolean	
7	Tf = T+	
8	Tx = T#t	
9	T2 = T0	

(T+ = [Number * Number --> Number]) ○ Substitution = Substitution ○ (T+ = [Number * Number -> Number])

	equation	substitution
6	T#t = Boolean	{ T1 := [[Number * Number --> Number] * T#t --> T0], Tf := [Number * Tx --> T2], Tnum1 = Number , T+ = [Number * Number -> Number]}
7	Tf = T+	
8	Tx = T#t	
9	T2 = T0	

(T#t = Boolean) ○ Substitution = Substitution ○ (T#t = Boolean)

	equation	substitution
7	Tf = T+	{ T1 := [[Number * Number --> Number] * Boolean --> T0], Tf := [Number * Tx --> T2],T+ = [Number * Number -> Number] , T#t = Boolean }
8	Tx = T#t	
9	T2 = T0	

(Tf = T+) ○ Substitution = Substitution ○ (Tf = T+)

	equation	substitution
8	Tx = T#t	{ T1 := [[Number * Number --> Number] * Boolean --> T0], Tf := [Number * Tx --> T2],T+ = [Number * Number -> Number], T#t = Boolean }
9	T2 = T0	
10	Tx = Number	
11	T2 = Number	

(Tx = T#t) ○ Substitution = Substitution ○ (Tx = T#t)

	equation	substitution
9	T2 = T0	{ T1 := [[Number * Number --> Number] * Boolean --> T0], Tf := [Number * T#t --> T2],T+ = [Number * Number -> Number], T#t = Boolean, Tx =T#t }
10	Tx = Number	
11	T2 = Number	

(T2 = T0) ○ Substitution = Substitution ○ (T2 = T0)

	equation	substitution
10	Tx = Number	{ T1 := [[Number * Number --> Number] * Boolean --> T0], Tf := [Number * T#t --> T0],T+ = [Number * Number -> Number], T#t = Boolean, Tx = T#t}
11	T2 = Number	

(Tx = Number) ○ Substitution = Substitution ○ (Tx = Number)

now we got Tx= Boolean and Tx = Number so we can say that the expression is not well-typed