CS 381 – Type Activity

For this activity you will be modifying the ExprPair code in the typeAct.hs starter file by implemnting the following operations applied to expression pairs.

1) The “Plus” function. Add a Plus function that takes two expressions as arguments.

a) If both arguments are integers then the Plus operator returns an integer that is the sum of the two arguments.

For example: sem (Plus (N 5) (N 10)) ➔ I 15

b) If both arguments are pairs then the Plus operator returns a pair.

For example: sem (Plus (Pair (N 5) (N 10)) (Pair (N 3) (N 4))) ➔ P 8 14

c) Any other cases will result in an error discovered by the static type checker.

For example: sem (Plus (Pair (N 5) (N 10)) (N 4)) ➔ Error

For example: semTC (Plus (Pair (N 5) (N 10)) (N 4)) ➔ TCError

2) The “Equal” function. Add an Equal function that takes two expressions as arguments and returns a boolean value. *Note: You will need to add B Bool to the data Val.*

a) If both arguments are integers then the Equal operator returns the appropriate boolean value.

For example: sem (Equal (N 5) (N 10)) ➔ B False

a) If both arguments are pairs then the Equal operator returns the appropriate boolean value.

For example: sem (Equal (Pair (N 10) (N 3)) (Pair (N 10) (N 3)))➔ B True

c) Any other cases will result in an error discovered by the static type checker.

For example: sem (Equal (Pair (N 5) (N 10)) (N 4)) ➔ Error

For example: semTC (Equal (Pair (N 5) (N 10)) (N 4)) ➔ TCError

3) The “Div” function. Add a Div function that takes two expressions e1 and e2 as arguments.

a) If both arguments are integers then the Div operator returns the integer result of e1 \`div\` e2.

For example: sem (Div (N 10) (N 2)) ➔ I 5

b) If the divisor is evaluated to zero then a “run-time” error occurs while executing the sem function.

For example: sem (Div (N 10) (N 0)) ➔ Error

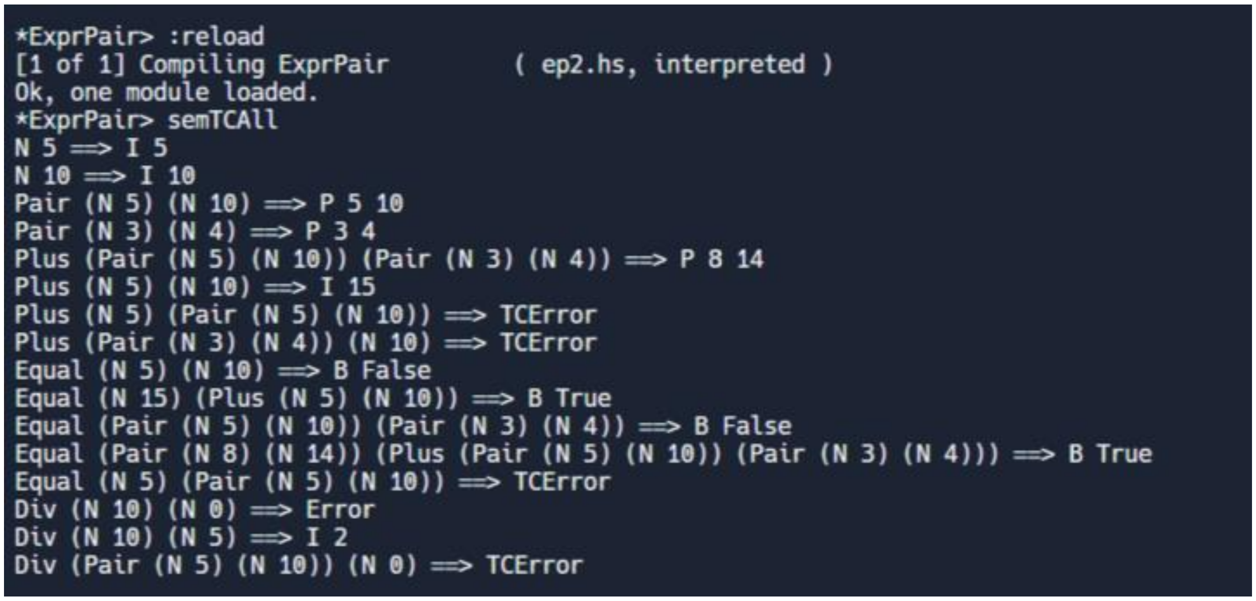
c) Any other case will result in an error discovered by the static type checker.

For example: semTC (Div (Pair (N 5) (N 10)) (N 4)) ➔ TCError

CS 381 – Type Activity

The following are the results with the test data:

```
a = N 5
b = N 10
zero = N 0
c = Pair a b
d = Pair (N 3) (N 4)
e = Plus c d
f = Plus a b
g = Plus a c
h = Plus d b
i = Equal a b
j = Equal (N 15) f
k = Equal c d
l = Equal (Pair (N 8) (N 14)) e
m = Equal a c
n = Div (N 10) zero
o = Div b a
p = Div c zero
ex = [a,b,c,d,e,f,g,h,i,j,k,l,m,n,o, p]
```



```
*ExprPair> :reload
[1 of 1] Compiling ExprPair         ( ep2.hs, interpreted )
Ok, one module loaded.
*ExprPair> semTCAll
N 5 ==> I 5
N 10 ==> I 10
Pair (N 5) (N 10) ==> P 5 10
Pair (N 3) (N 4) ==> P 3 4
Plus (Pair (N 5) (N 10)) (Pair (N 3) (N 4)) ==> P 8 14
Plus (N 5) (N 10) ==> I 15
Plus (N 5) (Pair (N 5) (N 10)) ==> TCError
Plus (Pair (N 3) (N 4)) (N 10) ==> TCError
Equal (N 5) (N 10) ==> B False
Equal (N 15) (Plus (N 5) (N 10)) ==> B True
Equal (Pair (N 5) (N 10)) (Pair (N 3) (N 4)) ==> B False
Equal (Pair (N 8) (N 14)) (Plus (Pair (N 5) (N 10)) (Pair (N 3) (N 4))) ==> B True
Equal (N 5) (Pair (N 5) (N 10)) ==> TCError
Div (N 10) (N 0) ==> Error
Div (N 10) (N 5) ==> I 2
Div (Pair (N 5) (N 10)) (N 0) ==> TCError
```