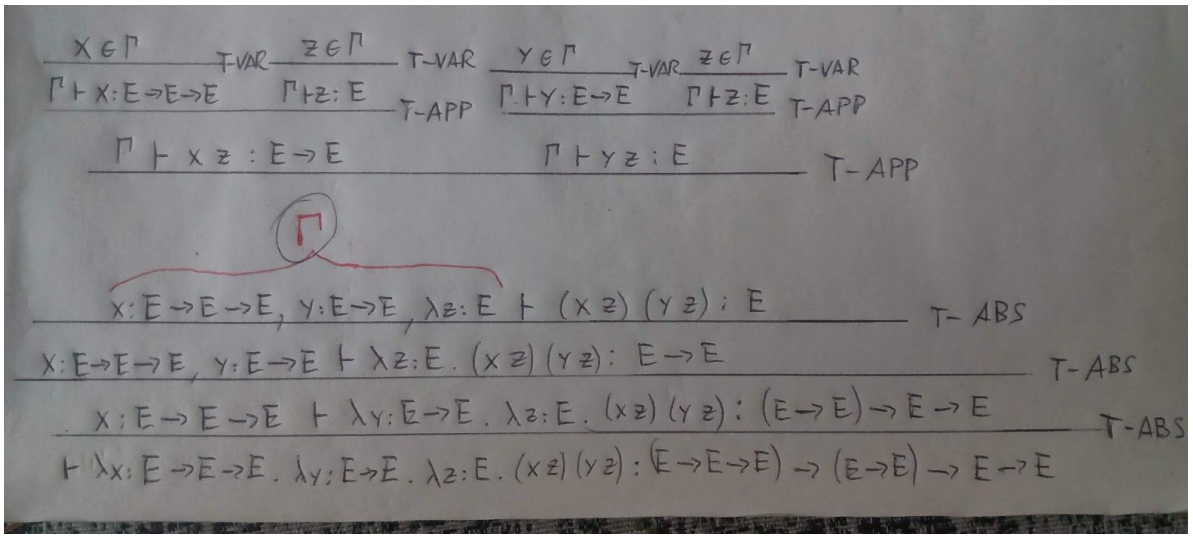
Trabajo práctico n° 3

Integrantes:

Ignacio Sebastián Moliné. Legajo M-6466/1

Sebastián Morales. Legajo M-6501/3

Ejercicio 1:



Ejercicio 2:
La función *infer* retorna un valor de tipo **Either String Type** debido a que la expresión que toma como argumento puede no tipar y, en consecuencia, devolver un mensaje de error en lugar de un tipo.

La función >>= toma un argumento de tipo **Either String Type** y una función de tipo **(Type -> Either String Type)**. Si el valor del primer argumento es un *Left error*, se comporta como la identidad y propaga un posible error en la inferencia de tipos. Si el valor del primer argumento es un *Right v*, devuelve el resultado de aplicar la función argumento a *v*.

Ejercicio 3:
Los archivos modificados para extender el intérprete fueron *code/src/Parse.y, code/src/PrettyPrinter.hs* y *code/src/SimplyTyped.hs*

Para legibilidad del informe, se decidió omitir la escritura de código en el mismo y estructurar los archivos de forma comentada, clara y concisa, separada por las distintas secciones según la estructura de los enunciados, de forma tal que se puedan consultar directamente para cada sección del informe. La sección correspondiente al ejercicio 3 es la *sección 6.*

Como excepción a esta metodología, mostraremos el procedimiento realizado en el archivo Parse.y (donde por una cuestión de estructura, no se pudo dejar una documentación delimitada en secciones). Cuando se pida volver a extender el intérprete con alguna funcionalidad nueva, léase este archivo en el mismo patrón para identificar los cambios realizados al parser, a la gramática y al lexer.

Parse.y: Agregamos los tokens TLet y TIn al listado

```
LET   { TLet }
IN    { TIn }
```

Los pusimos al mismo nivel de la abstracción en el listado de precedencias

```
%right VAR
%left '='
%right '->'
%right '\\' '.' LET IN
```

Agregamos una regla para Exp en la gramática

```
Exp   :: { LamTerm }
      : '\\' VAR ':' Type '.' Exp      { LAbs $2 $4 $6 }
      | LET VAR '=' Exp IN Exp         { LLet $2 $4 $6 }
```

Agregamos TLet y TIn al tipo de dato de los tokens

```
data Token = ...
     | TLet
     | TIn
```

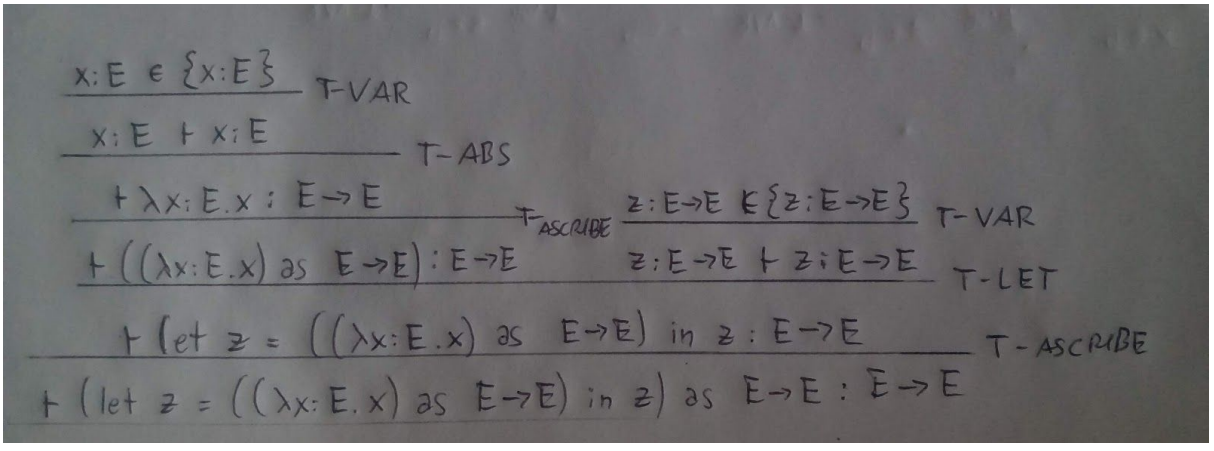
Agregamos los patrones de *let* e *in* para el lexer

```
lexer cont s = case s of
...
    ("let",rest)  -> cont TLet rest
    ("in",rest)   -> cont TIn rest
```

Ejercicio 4:

Véase la *sección 7* en los archivos PrettyPrinter.hs y SimplyTyped.hs, junto con las anotaciones referidas en el ejercicio 3 sobre Parse.y

Ejercicio 5:



Ejercicio 6:

Véase la *sección 8* en los archivos PrettyPrinter.hs y SimplyTyped.hs, junto con las anotaciones referidas en el ejercicio 3 sobre Parse.y

Ejercicio 7:

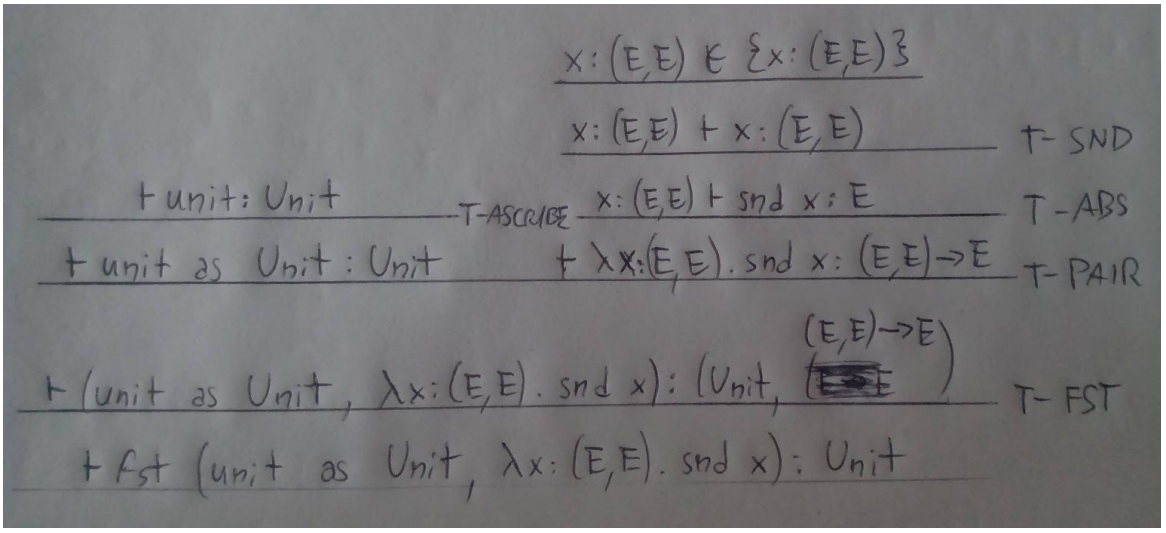
$$\frac{t_1 \to t_1'}{(t_1, t_2) \to (t_1', t_2)}\ \text{E}-\textsc{Pair1} \qquad \frac{t_2 \to t_2'}{(v, t_2) \to (v, t_2')}\ \text{E}-\textsc{Pair2}$$

$$\frac{t \to t'}{fst\ t \to fst\ t'}\ \text{E}-\textsc{Fst1} \qquad \frac{t \to t'}{snd\ t \to snd\ t'}\ \text{E}-\textsc{Snd1}$$

$$fst\ (v_1, v_2) \to v_1\ (\text{E}-\textsc{Fst2}) \qquad snd\ (v_1, v_2) \to v_2\ (\text{E}-\textsc{Snd2})$$

Ejercicio 8:
Véase la *sección 9* en los archivos PrettyPrinter.hs y SimplyTyped.hs, junto con las anotaciones referidas en el ejercicio 3 sobre Parse.y

Ejercicio 9:



Ejercicio 10:
Véase la *sección 10* en los archivos PrettyPrinter.hs y SimplyTyped.hs, junto con las anotaciones referidas en el ejercicio 3 sobre Parse.y

Ejercicio 11
Ejemplos/Ack.lam

```
def AckAux = \a:Nat -> Nat. \n:Nat. R (a (succ 0)) (\x:Nat.
\y:Nat. a x) n

def Ack = \m:Nat. R (\n:Nat. succ n) (\f:Nat->Nat. \y:Nat. AckAux
f) m
```