Programación Funcional

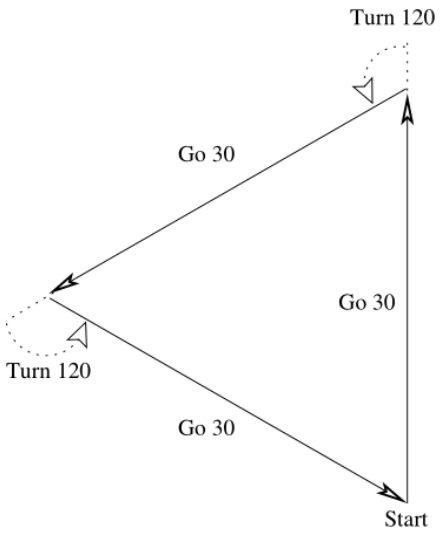
2do Parcial - La tortuga - Primer semestre 2021

La idea de gráficos de tortuga es originaria del lenguaje Logo. Sin embargo, era solamente uno de los componentes de tal lenguaje, aunque sí el más conocido y difundido. En esta evaluación utilizaremos una versión simplificada de un pequeño lenguaje para programar el dibujo de gráficos a través de la "tortuga", y realizaremos diferentes tratamientos a los programas escritos en este lenguaje. El tipo de datos que utilizaremos para expresar programas en este lenguaje será el siguiente:

```
data Point    = P Float Float
data Pen      = NoColour | Colour Float Float Float
type Angle    = Float
type Distance = Float
data Turtle   = T Pen Angle Point
data TCommand = Go Distance
              | Turn Angle
              | GrabPen Pen
              | TCommand :#: TCommand -- :#: es un constructor INFIJO
```

La tortuga consiste de un cabezal de escritura, que puede no dibujar o dibujar con cierto color (expresado en base a números entre 0 y 1, resultado de dividir cada elemento del código RGB de un color por 255), un ángulo en el que apunta y un punto del plano bidimensional donde se encuentra actualmente. Los comandos establecen, respectivamente, que la tortuga se puede mover cierta distancia (siempre positiva) en la dirección en la que apunta según su ángulo, girar cierto ángulo (negativo o positivo), cambiar de color el cabezal (o levantarlo, para poder mover sin dibujar), y secuenciar dos comandos realizando primero uno y luego el otro desde el estado donde terminó el primero. Así, por ejemplo, el siguiente programa especifica que se dibuje un triángulo de lado 20, con vértice en el punto actual, utilizando el cabezal actual:

```
triangle :: TCommand
triangle = Go 30 :#: Turn 120
       :#: Go 30 :#: Turn 120
       :#: Go 30 :#: Turn 120
```



El lenguaje de comandos de tortuga se puede compilar a uno más simple, al que llamaremos assembler de líneas, consistente simplemente en una secuencia de instrucciones de dibujado de líneas específicas. Para representar programas en este nuevo lenguaje utilizaremos los siguientes tipos y funciones:

```
data Line = Line Pen Point Point
 -- Dibuja una línea del 1er punto hasta el 2do, con el color dado
data LineAssembly = LA [Line]
endPoint :: Point -> Angle -> Distance -> Point
-- Calcula el punto final al arrancar el dibujo en el punto dado,
-- apuntando en el ángulo dado, y moverse la distancia dada
```

Para compilar, se pedirá definir una función

```
compileT :: TCommand -> Turtle -> (LineAssembly, Turtle)
```

El significado de la compilación de un programa es una función que dado el estado de la tortuga actual, devuelve el programa compilado y el nuevo estado de la tortuga (observar que pueden haber cambiado cualquiera de los 3 componentes). La compilación del programa triangle dado antes sería entonces:

```
compileT triangle (T pen a p0) =
          let p1 = endPoint p0 a 30
              p2 = endPoint p1 (a+120) 30
              p3 = endPoint p2 (a+240) 30
           in (LA [ Line pen p0 p1, Line pen p1 p2, Line pen p2 p3 ]
              ,T pen (a+360) p3)
```