1 1C2024 Recu

1.1 Ejercicio 1

En este ejercicio vamos a modelar lógica proposicional en Haskell, de modo de poder construir fórmulas proposicionales y evaluarlas bajo distintas valuaciones.

```haskell
data Prop = Var String | No Prop | Y Prop Prop | O Prop Prop | Imp Prop Prop

type Valuacion = String -> Bool
```

Por ejemplo, la expresión:

```haskell
Y (Var "P") (No (Imp (Var "Q") (Var "R")))
```

representa la proposición lógica: $P \wedge \neg(Q \rightarrow R)$. Las valuaciones se representan como funciones que a cada variable proposicional le asignan un valor booleano. Por ejemplo:

```haskell
\x -> x == "P"
```

asigna el valor Verdadero a la variable `P`, y Falso a todas las demás.

1. Nos piden dar el tipo y definir **foldProp** y **recProp** que implementan los esquemas de recursión estructural y primitiva del tipo **Prop**.

```haskell
-- fold (recursi n estructural)
foldProp :: (String -> a) -> (a -> a) -> (a -> a -> a) -> (a -> a -> a) -> (a -> a -> a) -> Prop -> a
foldProp fVar fNo fY fO fImp prop = case prop of
        Var s -> fVar s
        No p -> fNo (rec p)
        Y p p' -> fY (rec p) (rec p')
        O p p'-> fO (rec p) (rec p')
        Imp p p'-> fImp (rec p) (rec p')
        where rec = foldProp fVar fNo fY fO fImp

-- la recursi n primitiva lo  nico  que agrega es la posibilidad de mantener referencia sin procesar de la estructura
recProp :: (String -> a) -> (Prop -> a -> a) -> (Prop -> Prop -> a -> a -> a) -> (Prop -> Prop -> a -> a -> a) -> (Prop -> Prop -> a -> a -> a) -> Prop -> a
recProp fVar fNo fY fO fImp prop = case prop of
        Var s -> fVar s
        No p -> fNo p (rec p)
        Y p p' -> fY p p' (rec p) (rec p')
        O p p'-> fO p p' (rec p) (rec p')
        Imp p p'-> fImp p p' (rec p) (rec p')
        where rec = recProp fVar fNo fY fO fImp
```

2. Para el punto b nos piden definir la función **variables** que básicamente devuelve, sin repetidos, las variables de cierta proposición

```haskell
variables :: Prop -> [String]
variables prop = eliminarRepetidos (foldProp (\s -> [s]) (\p -> p) (++) (++) (++) prop)
-- ejemplo :
ej1 = (O (Var "P") (No(Y (Var "Q") (Var "P"))))

eliminarRepetidos :: Eq a => [a] -> [a]
eliminarRepetidos [] = []
eliminarRepetidos (x:xs) = if (elem x xs) then eliminarRepetidos xs else x : eliminarRepetidos xs
```

3. La función **evaluar** indica el valor de verdad de la fórmula

```haskell
-- evaluar
evaluar :: Valuacion -> Prop -> Bool
evaluar val = foldProp val (not) (&&) (||) (\p q -> not p || q)
```

4. La función **estaEnFNN** nos indica si la fórmula está en su forma normal negada (sin implicaciones y cuando la negación sólo seaplica a variables) *(consultar)*

1.2 Ejercicio 2

Considerar las siguientes definiciones sobre árboles con información en las hojas

```haskell
data AIH a = Hoja a | Bin (AIH a) (AIH a)

    der :: AIH a -> AIH a
{D} der (Bin _ d) = d

    esHoja :: AIH a -> Bool
{E0}esHoja (Hoja _) = True
{E1}esHoja (Bin _ _) = False

    mismaEstructura :: AIH a -> (AIH a -> Bool)
{M0}mismaEstructura (Hoja _) = esHoja
{M1}mismaEstructura (Bin i d) = \t ->
      not (esHoja t) &&
      mismaEstructura i (izq t) &&
      mismaEstructura d (der t)

    izq :: AIH a -> AIH a
{I} izq (Bin i _) = i
```

Nos piden demostrar:

$$\forall t, u : AIH\ a. \quad mismaEstructura\ t\ u = mismaEstructura\ u\ t$$

1. Primer paso de la demostración es decidir qué vamos a hacer. En este caso, vamos a hacer inducción sobre **AIH a**. En particular, sobre el primero. Entonces, decimos

$$\forall x, u : AIH\ a. \quad mismaEstructura\ x\ u = mismaEstructura\ u\ x$$

- **QVQ** caso base :

$$\begin{aligned} & mismaEstructura\ (Hoja\ a)\ u = mismaEstructura\ u\ (Hoja\ a) \\ & \overset{\{M0\}}{\equiv} esHoja\ u = mismaEstructura\ u\ (Hoja\ a) \end{aligned}$$

  Esto nos lleva a destrabar a **u** por extensionalidad y separar en 2 casos :

  - Caso **u es Hoja**

$$\begin{aligned} & esHoja\ (Hoja\ a) = mismaEstructura\ (Hoja\ a)\ (Hoja\ a) \\ & \overset{\{E0\}}{\equiv} True = mismaEstructura\ (Hoja\ a)\ (Hoja\ a) \\ & \overset{\{M0\}}{\equiv} True = esHoja\ (Hoja\ a) \\ & \overset{\{E0\}}{\equiv} True = True \\ & \equiv True \end{aligned}$$

  - Caso **u no es Hoja**

$$\begin{aligned} & esHoja\ (Bin\ (AIH\ a)\ (AIH\ a)) = mismaEstructura\ (Bin\ (AIH\ a)\ (AIH\ a))\ (Hoja\ a) \\ & \overset{\{E1\}}{\equiv} False = mismaEstructura\ (Bin\ (AIH\ a)\ (AIH\ a))\ (Hoja\ a) \\ & \overset{\{M1\}}{\equiv} False = (\backslash t -> not\ (esHoja\ t)\ \&\&\ mismaEstructura\ i\ (izq\ t) \&\&\ mismaEstructura\ d\ (der\ t))\ (Hoja\ a) \\ & \overset{\beta}{\equiv} False = (not\ (esHoja\ (Hoja\ a))\ \&\&\ mismaEstructura\ i\ (izq\ (Hoja\ a)) \&\&\ mismaEstructura\ d\ (der\ (Hoja\ a) \\ & \overset{\{E0\}}{\equiv} False = (not\ (True)\ \&\&\ mismaEstructura\ i\ (izq\ (Hoja\ a)) \&\&\ mismaEstructura\ d\ (der\ (Hoja\ a))) \\ & \overset{not}{\equiv} False = False\ \&\&\ mismaEstructura\ i\ (izq\ (Hoja\ a)) \&\&\ mismaEstructura\ d\ (der\ (Hoja\ a))) \\ & \equiv False = False \\ & \equiv True \end{aligned}$$

- Ahora que probamos para el caso base, **QVQ**

$$\forall i, d : AIH\ a, r : a. \quad P(i) \wedge P(d) \implies P(Bin\ i\ r\ d)$$

  Con esto, queremos demostrar que :

$$P(Bin\ i\ r\ d) = mismaEstructura\ (Bin\ i\ r\ d)\ u = mismaEstructura\ u\ (Bin\ i\ r\ d)$$

Asumimos que valen $P(i) \wedge P(d)$ y serán nuestras HI :

$$P(i) = mismaEstructura\ i\ u = mismaEstructura\ u\ i$$
$$P(d) = mismaEstructura\ d\ u = mismaEstructura\ u\ d$$

Con esto en mente, procedemos a demostrarlo:

$$mismaEstructura\ (Bin\ i\ r\ d)\ u = mismaEstructura\ u\ (Bin\ i\ r\ d)$$
$$\overset{\{M1\}}{\equiv} (\backslash t-> not\ (esHoja\ t)\ \&\&\ mismaEstructura\ i\ (izq\ t)\&\&\ mismaEstructura\ d\ (der\ t))\ u = \ldots$$
$$\overset{\beta}{\equiv} not\ (esHoja\ u)\ \&\&\ mismaEstructura\ i\ (izq\ u)\&\&\ mismaEstructura\ d\ (der\ u) = \ldots$$

Esto nos traba la situación y por extensionalidad separamos en dos casos :

- Caso **u es Hoja**

$$not\ (esHoja\ (Hoja\ a))\ \&\&\ mismaEstructura\ i\ (izq\ (Hoja\ a))\&\&\ mismaEstructura\ d\ (der\ (Hoja\ a)) = \ldots$$
$$\overset{\{E0\}}{\equiv} not\ (True)\ \&\&\ mismaEstructura\ i\ (izq\ (Hoja\ a))\&\&\ mismaEstructura\ d\ (der\ (Hoja\ a)) = \ldots$$
$$\overset{not}{\equiv} False\ \&\&\ mismaEstructura\ i\ (izq\ (Hoja\ a))\&\&\ mismaEstructura\ d\ (der\ (Hoja\ a)) = \ldots$$
$$\equiv False = mismaEstructura\ (Hoja\ a)\ (Bin\ i\ r\ d)$$
$$\overset{\{M0\}}{\equiv} False = esHoja\ (Bin\ i\ r\ d)$$
$$\overset{\{E1\}}{\equiv} False = False$$
$$\equiv True$$

- Caso **u no es Hoja**

$$not\ (esHoja\ (Bin\ (AIH\ a)\ (AIH\ a)))\ \&\&\ mismaEstructura\ i\ (izq\ (Bin\ (AIH\ a)\ (AIH\ a)))$$
$$\&\&mismaEstructura\ d\ (der\ (Bin\ (AIH\ a)\ (AIH\ a))) = \ldots$$
$$\overset{\{E1\},not}{\equiv} True\&\&\ mismaEstructura\ i\ (izq\ (Bin\ (AIH\ a)\ (AIH\ a)))\&\&$$
$$mismaEstructura\ d\ (der\ (Bin\ (AIH\ a)\ (AIH\ a))) = \ldots$$
$$\equiv mismaEstructura\ i\ (izq\ (Bin\ (AIH\ a)\ (AIH\ a)))\&\&mismaEstructura\ d\ (der\ (Bin\ (AIH\ a)\ (AIH\ a))) = \ldots$$
$$\overset{\{I\},\{D\}}{\equiv} mismaEstructura\ i\ (iu)\&\&mismaEstructura\ d\ (du) = \ldots \quad \texttt{\{donde iu, du : AIH a\}}$$
$$\overset{HI}{\equiv} True\&\&True = \ldots$$
$$\equiv True = mismaEstructura\ (Bin\ (AIH\ a)\ (AIH\ a))\ (Bin\ i\ r\ d)$$
$$\overset{\{M1\}}{\equiv} True = (\backslash t-> not\ (esHoja\ t)\ \&\&\ mismaEstructura\ iu\ (izq\ t)\&\&\ mismaEstructura\ du\ (der\ t))\ (Bin\ i\ r\ d)$$
$$\overset{\beta}{\equiv} True = not(esHoja\ (Bin\ i\ r\ d))\&\&\ mismaEstructura\ iu(izq(Bin\ i\ r\ d))\&\&mismaEstructura\ du(der(Bin\ i\ r\ d$$
$$\overset{\{E1\},not}{\equiv} True = True\&\&\ mismaEstructura\ iu\ (izq\ (Bin\ i\ r\ d))\&\&\ mismaEstructura\ du\ (der\ (Bin\ i\ r\ d))$$
$$\overset{\{I\},\{D\}}{\equiv} True = mismaEstructura\ iu\ i\ \&\&\ mismaEstructura\ du\ d$$
$$\overset{HI}{\equiv} True = mismaEstructura\ True\ \&\&\ True$$
$$\equiv True = True$$
$$\equiv True$$

1.3 Ejercicio 3

Se desea extender el cálculo lambda simplemente tipado para modelar **Árboles con info en las hojas**. Para eso se extienden tipos y expresiones como sigue :

$$\tau ::= \ldots \mid AIH(\tau)$$
$$M ::= \ldots \mid Hoja(M) \mid Bin(M, M) \mid case\ M\ of\ Hoja\ x \rightsquigarrow M; Bin(i, d) \rightsquigarrow M$$

a) Nos piden introducir las reglas de tipado : tenemos que mirar los nuevos términos

$$\frac{\Gamma \vdash M : \tau}{\Gamma \vdash Hoja(M) : AIH(\tau)} \text{ T-HOJA}$$

$$\frac{\Gamma \vdash M_1 : AIH(\tau) \qquad \Gamma \vdash M_2 : AIH(\tau)}{\Gamma \vdash Bin(M_1, M_2) : AIH(\tau)} \text{ T-BIN}$$

$$\frac{\Gamma \vdash \ M_1 : AIH(\tau) \qquad \Gamma, x : \tau \vdash \ M_2 : \sigma \qquad \Gamma, i : AIH(\tau), d : AIH(\tau) \vdash \ M_3 : \sigma}{\Gamma \vdash case\ M_1\ of\ Hoja\ x \rightsquigarrow M_2; Bin(i, d) \rightsquigarrow M_3 : AIH(\tau)} \text{ T-CASE}$$

b) El nuevo conjunto de valores, las reglas de cómputo y congruencia

- Conjunto de valores

$$V ::= \ldots \mid Hoja(V) \mid Bin(V, V)$$

- Reglas de cómputo Para **case .. of ...** tenemos el caso en que M1 es Hoja (y el subtérmino es valor (estamos en reglas de cómputo))

$$\frac{}{case\ (Hoja(V))\ of\ Hoja\ x \rightsquigarrow M_2; Bin(i, d) \rightsquigarrow M_3 \rightarrow M2\{x := V\}} \text{ E-CASEHOJA1}$$

  Y el caso en que M1 es un Bin(V,V)

$$\frac{}{case\ (Bin(V_1, V_2))\ of\ Hoja\ x \rightsquigarrow M_2; Bin(i, d) \rightsquigarrow M_3 \rightarrow M3\{i := V_1\}\{d := V_2\}} \text{ E-CASEBIN1}$$

  Ahora bien, hay escenarios en los que los subtérminos de los extendidos a nuestra gramática no están en su forma normal y tendremos que evaluarlos, para ello se definen ...

- Reglas de congruencia

$$\frac{M \rightarrow M'}{case\ M\ of\ Hoja\ x \rightsquigarrow M_2; Bin(i, d) \rightsquigarrow M_3 \rightarrow case\ M'\ of\ Hoja\ x \rightsquigarrow M_2; Bin(i, d) \rightsquigarrow M_3} \text{ E-CASE}$$

$$\frac{M \rightarrow M'}{case\ (Hoja(M))\ of\ Hoja\ x \rightsquigarrow M_2; Bin(i, d) \rightsquigarrow M_3 \rightarrow M2\{x := M'\}} \text{ E-CASEHOJA2}$$

$$\frac{N \rightarrow N'}{case\ (Bin(N, V_2))\ of\ Hoja\ x \rightsquigarrow M_2; Bin(i, d) \rightsquigarrow M_3 \rightarrow M3\{i := N'\}\{d := V_2\}} \text{ E-CASEBIN2}$$

$$\frac{N \rightarrow N'}{case\ (Bin(V_1, N))\ of\ Hoja\ x \rightsquigarrow M_2; Bin(i, d) \rightsquigarrow M_3 \rightarrow M3\{i := V_1\}\{d := N'\}} \text{ E-CASEBIN3}$$

c) Cómo se reduce :

$$\begin{aligned}
&case\ (\lambda n : Nat.Hoja(n))\ Succ(zero)\ of\ Hoja\ x \rightsquigarrow Succ(Pred(x)); Bin(i, d) \rightsquigarrow zero \\
&\overset{E-CASE}{\rightarrow} case\ Hoja(Succ(zero))\ of\ Hoja\ x \rightsquigarrow Succ(Pred(x)); Bin(i, d) \rightsquigarrow zero \\
&\overset{E-CASEHOJA1}{\rightarrow} Succ(Pred(x))\{x := Succ(zero)\} \\
&\rightarrow Succ(Pred(Succ(zero))) \\
&\overset{E-PREDSUCC}{\rightarrow} Succ(zero))
\end{aligned}$$