

# Familiarización con el Entorno Haskell

Curso 2003/04, Fecha:10/10/2003

**Enunciado 1 (Compilar)** *Cargar el siguiente programa en un entorno Haskell*

```
> module Main(main) where
> f (x,y) = if x == 'a' then True
>           else y
>
> g :: Int → Int
> g x = 2 * x + 1
>
> reaplica f x = f (f x)
>
> h :: (Float,Float) → Float
> h (x,y) = if x < 5.5 then x + 1
>           else y
>
> suma1 [] = []
> suma1 (x:r) = (x + 1): suma1 r
>
> main = do putStrLn "Introduce tu nombre"
>           cad ← getLine
>           putStrLn ("Nombre al revés:_" ++ reverse cad)
```

**Enunciado 2 (Tipo)** *Indicar cuál es el tipo de la función f y comprobar que coincide con el tipo inferido por el sistema*

**Enunciado 3 (reaplica)** *Evaluar el resultado de la expresión reaplica g 3*

**Enunciado 4 (evaluaError)** *Evaluar el resultado de la expresión h (6, 1/0)*

**Enunciado 5 (evaluaSinError)** *Evaluar el resultado de la expresión h (4, 1/0)*

**Enunciado 6 (suma2)** *Construir una función similar a la función suma1 pero que sume 2 a todos los elementos de una lista*

**Enunciado 7 (por2)** *Construir una función similar a la función suma1 pero que multiplique por 2 todos los elementos de una lista*

**Enunciado 8 (longs)** Construir una función que toma una lista de palabras y devuelva una lista de enteros correspondientes a la longitud de cada palabra. Puede utilizarse la función predefinida **length** que calcula la longitud de una lista. Tipo:

> *longs* :: [**String**] → [**Int**]

?- *longs* ["juan", "ana", "pedro"] [4, 3, 5]

**Enunciado 9 (aplica)** Construir una función que tome como argumento una función y una lista y devuelva la lista resultante de aplicar la función a cada elemento de la lista. Tipo:

> *aplica* :: (a → b) → [a] → [b]

?- *aplica* (\*5) [2, 3, 4] [10, 15, 20]

**Enunciado 10 (Reescribir)** Reescribir las funciones *suma1*, *suma2*, *por2* y *longs* utilizando la función *aplica*

**Enunciado 11 (Mayusculas(Opcional))** Modificar el programa para que imprima el nombre en mayúsculas. Puede utilizarse la función **toUpper** :: **Char** → **Char** que convierte un carácter en mayúsculas.