

# Erlang

## Ejercicio 2: Recursión y Listas

### 1 Iteración

Implemente las siguientes definiciones utilizando recursión:

- `iter:ford(N)` que visualice en pantalla todos los enteros desde  $N$  hasta 1.
- `iter:fora(N)` que visualice en pantalla todos los enteros desde 1 hasta  $N$ .
- `iter:suma(N)` que calcule la suma de todos los enteros desde 1 hasta  $N$ .

La función `io:format` puede utilizarse para visualizar valores como efecto colateral de su evaluación. Por ejemplo, `io:format("~w~n", [Numero])` visualizaría el valor asociado al identificador `Numero`.

### 2 Recursión sobre listas

Defina las siguientes funciones sobre listas (todas definidas en el módulo `listas`):

- `listas:nth(N, L)` devuelve el  $N$ -ésimo elemento de la lista  $L$
- `listas:sublist(L, N)` devuelve una lista con los  $N$  primeros elementos de la lista  $L$
- `listas:seq(Inferior, Superior)` devuelve una lista con todos los números entre *Inferior* y *Superior*.

Por ejemplo,

```
1> listas:nth(3, [2, 3, 4, 5]).
4
2> listas:sublist([2,3,4,5], 2).
[2, 3]
3> listas:seq(2,5).
[2,3,4,5]
```

### 3 Ordenación

- Defina una función `ordena:inserta(X, ListaOrdenada)`, que dado un `number(X)` y una lista con números en orden creciente (*ListaOrdenada*) devuelve una nueva lista ordenada con  $X$  y todos los valores de *ListaOrdenada*. Por ejemplo:

```
1> ordena:inserta(4, [1,3,5,7]).
[1,3,4,5,7]
2> ordena:inserta(4, []).
[4]
```

- Utilizando la función anterior, defina `ordena:ord_insercion(L)`, que dada una lista  $L$  con números devuelve una lista con los mismos números ordenada de forma ascendente.

```
3> ordena:ord_insercion([2,5,1,4,3,6]).
[1,2,3,4,5,6]
```

- Implemente la siguiente definición recursiva en el módulo `ordena`:

```
qs([]) ->
  [];
qs([X|Xs]) ->
  qs( [Y || Y <- Xs, Y < X] ) ++ [X] ++ qs( [Y || Y <- Xs, Y >= X] ).
```

## 4 Más definiciones sobre listas

Defina las siguientes funciones clásicas sobre listas:

- `listas:append(L1, L2)`, concatenación de las listas  $L1$  y  $L2$ .
- `listas:reverse(L)`, lista  $L$  en orden inverso.
- `listas:flatten(L)`, aplanar  $L$

Por ejemplo,

```
1> listas:append([1,2,3], [4,5,6]).
[1,2,3,4,5,6]
2> listas:reverse([1,2,3]).
[3,2,1]
3> listas:flatten([1, [2,3,4], [], 5, [6, [7, 8], 9], 10]).
[1,2,3,4,5,6,7,8,9,10]
```