

# Paradigmas de Programación

## Práctica 11

En estos ejercicios vamos a intentar practicar un estilo de programación más imperativo. Para ello, evitaremos las definiciones recursivas y el uso del pattern-matching, pero, naturalmente, podrán usarse variables (valores de tipo  $\alpha$  ref) y bucles.

1. (opcional) Redefina en un archivo ***list\_loop.ml*** las funciones definidas a continuación. (Está prohibido el uso de la palabra reservada ***rec*** y cualquier valor definido fuera del módulo ***Stdlib***, salvo ***List.hd*** y ***List.tl***. Tampoco puede usarse la función (@))

```
let length = List.length

let rec last = function
  [] -> failwith "last"
| h :: [] -> h
| _ :: t -> last t

let nth = List.nth

let rev = List.rev

let append = List.append

let concat = List.concat

let for_all = List.for_all

let exists = List.exists

let find_opt = List.find_opt

let iter = List.iter

let fold_left = List.fold_left
```

El archivo ***list\_loop.ml*** debe compilar sin errores con la orden  
**`ocamlc -c list_loop.mli list_loop.ml`**

2. (opcional) Redefina en un archivo ***array\_loop.ml*** las funciones definidas a continuación. (Está prohibido el uso de la palabra reservada ***rec*** y cualquier valor definido fuera del módulo ***Stdlib***, salvo ***Array.length*** y ***Array.init***)

```
let append = Array.append
```

```
let sub = Array.sub
```

```
let copy = Array.copy
```

```
let fill = Array.fill
```

```
let blit = Array.blit
```

```
let to_list = Array.to_list
```

```
let iter = Array.iter
```

```
let fold_left = Array.fold_left
```

```
let for_all = Array.for_all
```

```
let exists = Array.exists
```

```
let find_opt = Array.find_opt
```

El archivo ***array\_loop.ml*** debe compilar sin errores con la orden  
***ocamlc -c array\_loop.mli array\_loop.ml***