Introducción a la programación

Práctica 5: Recursión sobre listas

Ejercicio 2.1

Implementar `pertenece :: (Eq t) => t -> [t] -> Bool`
según la siguiente especificación:

`problema pertenece` (e: T, s: $seq\langle T \rangle$) : $\mathbb{B}$ {
  `requiere:` { $True$ }
  `asegura:` { $resultado = true \leftrightarrow e \in s$ }
}

Ejercicio 2.4

Ejercicio 2.4 Definir la siguiente función sobre listas:

```
hayRepetidos :: (Eq t) => [t] -> Bool
```

`problema` `hayRepetidos` (s: $seq\langle T \rangle$) : $\mathbb{B}$ {
  `requiere:` { $True$ }
  `asegura:` { $resultado = true \leftrightarrow$ existen dos posiciones distintas de s con igual valor }
}

Sugerencia: Utilizar la función `pertenece` del ej 2.1

Ejercicio 2.5

Implementar `quitar :: (Eq t) => t -> [t] -> [t]`, que dados un entero x y una lista xs, elimina la primera aparición de x en la lista xs (de haberla).

Ejercicio 3.3

Ejercicio 3.3 Definir las siguientes funciones sobre listas de enteros

`problema maximo` (s: $seq\langle\mathbb{Z}\rangle$) : $\mathbb{Z}$ $\{$
  `requiere:` $\{ |s| > 0 \}$
  `asegura:` $\{ resultado \in s \wedge$ todo elemento de s es menor o igual a $resultado\}$
$\}$

Ejercicio 3.9

Ejercicio 3.9 Definir las siguientes funciones sobre listas de enteros

`problema ordenar` (s: $seq\langle\mathbb{Z}\rangle$) : $seq\langle ent\rangle$ {
  `requiere:` { True }
  `asegura:` { $resultado$ contiene los elementos de s ordenados de forma creciente}
}

Sugerencia: Hay muchas formas distintas de ordenar secuencias. Una opción puede ser utilizar la función `maximo` y la función `quitar` que ya implementamos.