Clase 02
Esquemas de recursión
Tipos de datos inductivos

Esquemas de recursión sobre listas

Recursión estructural

Sea `g :: [a] -> b` definida por dos ecuaciones:

```
g [] = <caso base>
g (x : xs) = <caso recursivo>
```

`g` está dada por recursión estructural si:

1. El caso base devuelve un valor z "fijo" (no depende de `g`).
2. El caso recursivo no puede usar las variables `g` ni `xs`, salvo en la expresión (`g xs`).

```
g []       = z
g (x : xs) = ...x...(g xs)...
```

Toda recursión estructural es una instancia de foldr.

Recursión primitiva

Sea `g :: [a] -> b` definida por dos ecuaciones:

```
g []       = <caso base>
g (x : xs) = <caso recursivo>
```

`g` está dada por recursión primitiva si:

1. El caso base devuelve un valor z "fijo" (no depende de `g`).
2. El caso recursivo no puede usar las variables g, salvo en la expresión (`g xs`). (sí la variable `xs`)

```
g []       = z
g (x : xs) = ...x...xs...(g xs)...
```

Observación:

- Todas las definiciones dadas por recursión estructural también están dadas por recursión primitiva.
- Hay definiciones dadas por recursión primitiva que no están dadas por recursión estructural.

Toda recursión primitiva es una instancia de recr

Recursión iterativa

Sea `g :: b -> [a] -> b` definida por dos ecuaciones:

```
g [] = <caso base>
g (x : xs) = <caso recursivo>
```

`g` está dada por recursión iterativa si:

1. El caso base devuelve el acumulador `ac`.
2. El caso recursivo invoca inmediatamente a (`g ac' xs`), donde `ac'` es el acumulador actualizado en función de su valor anterior y el valor de `x`.

Toda recursión iterativa es una instancia de foldl.

Tipos de datos algebraicos

En general un tipo de dato algebraico tiene la siguiente forma:

```
data T = CBase₁       <parámetros>
       ...
       | CBaseₙ       <parámetros>
       | CRecursivo₁  <parámetros>
       ...
       | CRecursivoₙ  <parámetros>
```

- Los constructores base no reciben parámetros de tipo `T`.
- Los constructores recursivos reciben al menos un parámetro de tipo `T`.
- Los valores de tipo `T` son los que se pueden construir aplicando constructores base y recursivos un número finito de veces y sólo esos.

Esquemas de recursión sobre otras estructuras

La recursión estructural se generaliza a tipos algebraicos en general. Supongamos que `T` es un tipo algebraico. Dada una función `g :: T -> Y` definida por ecuaciones:

```
g(CBase_1 <parámetros>) = <caso base_1>
...
g(CBase_n <parámetros>) = <caso base_1>
g(CRecursivo_1 <parámetros>) = <caso recursivo_1> \
...
g(CRecursivo_n \<parámetros\>) = \<caso recursivo_n\> \
```

g está dada por recursión estructural si:

1. Cada caso base se escribe combinando los parámetros.
2. Cada caso recursivo se escribe combinando:
   - Los parámetros del constructor que no son de tipo T.
   - El llamado recursivo sobre cada parámetro de tipo T.

   Pero:
   - Sin usar los parámetros del constructor que son de tipo T.
   - Sin hacer a otros llamados recursivos.