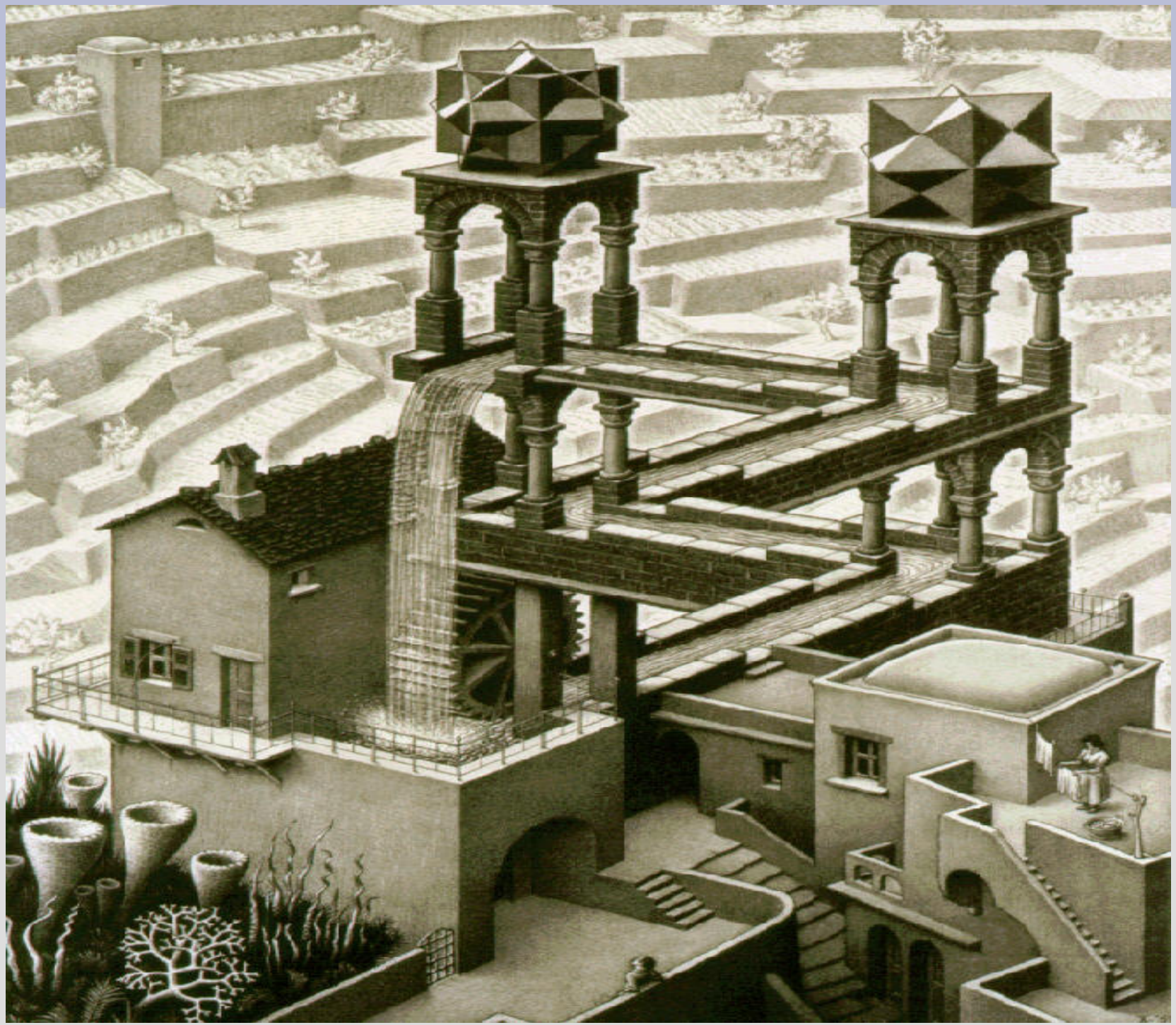
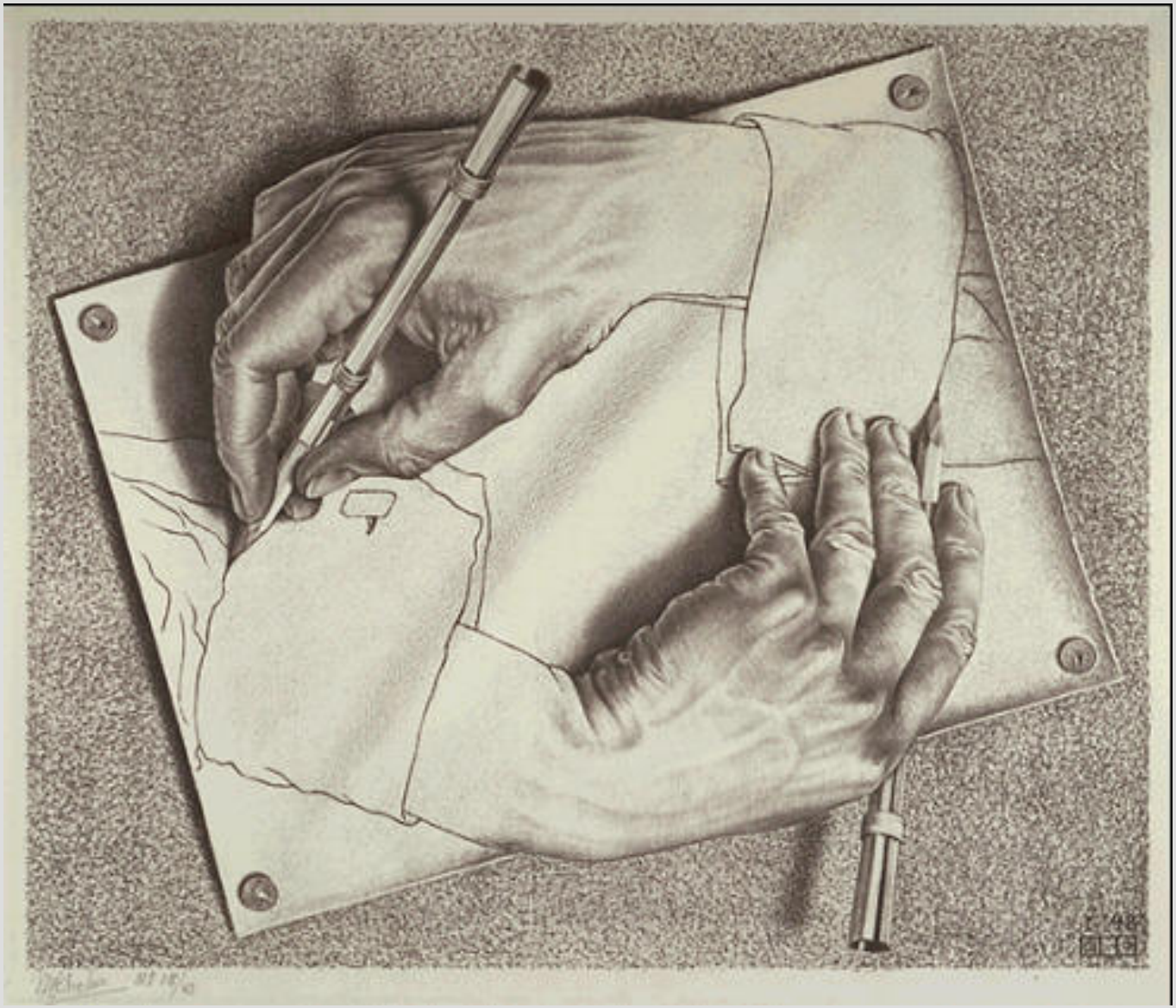
Fundamentos de programación

Funciones recursivas y listas

Fundamentos de programación
E.I.S.C.
Ángela Villota Gómez



Recursión



Introducción

- *¿qué es **Recursión**?* cierto diccionario mal intencionado dice lo siguiente:

Recursión (sustantivo): ver recursión

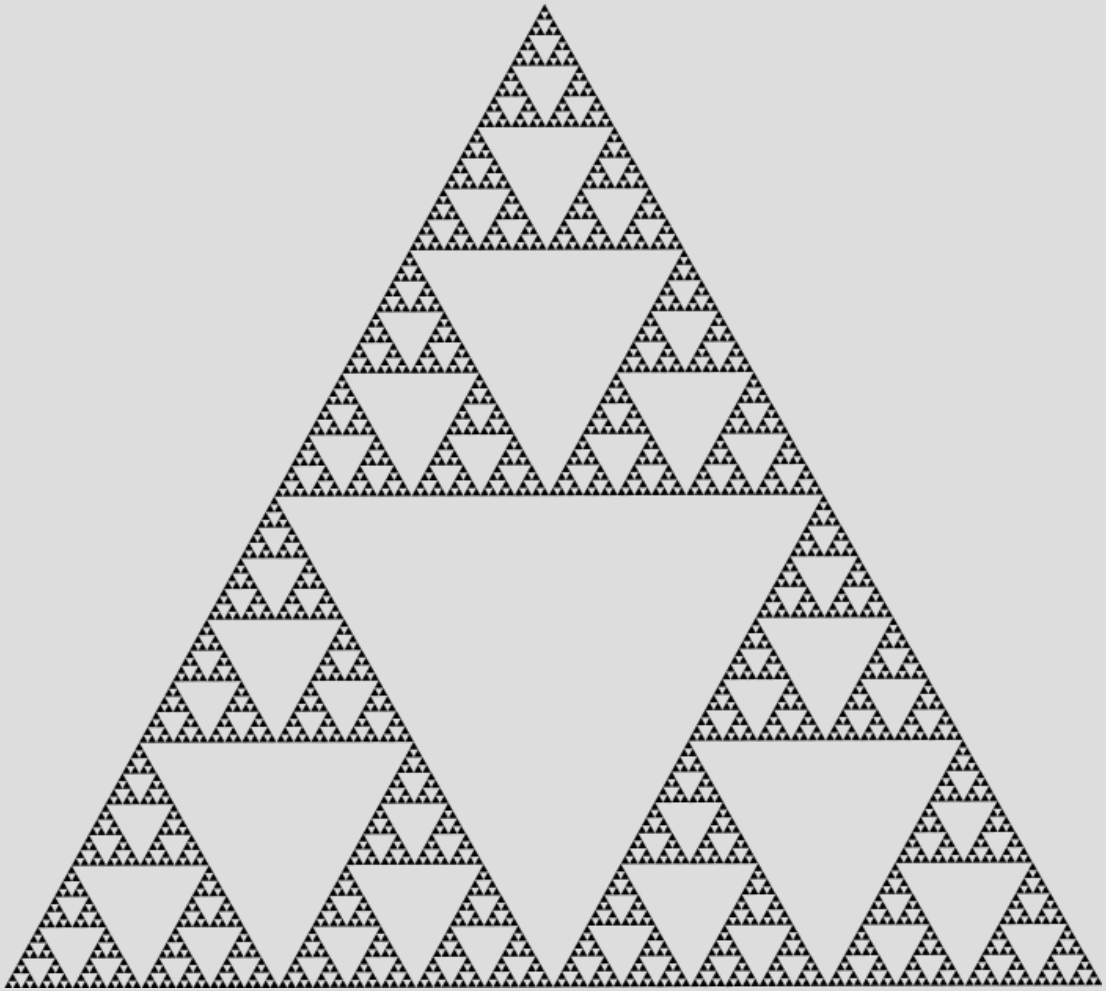
- ¿se imagina entonces qué es la ***recursión***?

  Es definir algo en términos de si mismo

Veamos un ejemplo: El triángulo de Sierpinski

Introducción

- El triangulo de Sierpinski es un fractal que puede construirse a partir de cualquier triangulo



Recursividad

- En computación, la **recursión** se presenta en funciones y en tipos de datos
- Una **función es recursiva** si en su definición tiene al menos un llamado a sí misma
- Un **tipo de datos es recursivo** si está definido en términos de sí mismo.

Ejemplo:

el conjunto de potencias de 2: 1 2 4 8 16 32 64 ...

podemos definirlas como p(0) = 1 y p(n+1) = p(n)*2

Ejemplos de funciones recursivas

- Potencias de dos
- Factorial
- Serie de fibonacci
- Triangulo de Sierpinski

Ejemplos de funciones recursivas: potencias de dos

- Definición:

$$F(n) = \begin{cases} 1 & \text{si n=0} \\ 2 \cdot F(n-1) & \text{para n>0 en los naturales} \end{cases}$$

- En scheme:

```
(define (potencias-dos n)
  (cond
    [(= n 0) 1]
    [else
     (* 2 (potencias-dos (- n 1) ) )]))
```

Ejemplos de funciones recursivas: factorial

- Definición:

$$n! = \begin{cases} 1 & \text{si n=0} \\ n \cdot (n-1)! & \text{para n>0 en los naturales} \end{cases}$$

- en scheme:

```
(define (factorial n)
  (cond
    [(= n 0) 1]
    [else
     (* n (factorial (- n 1)))]))
```

Ejemplos de funciones recursivas: fibonacci

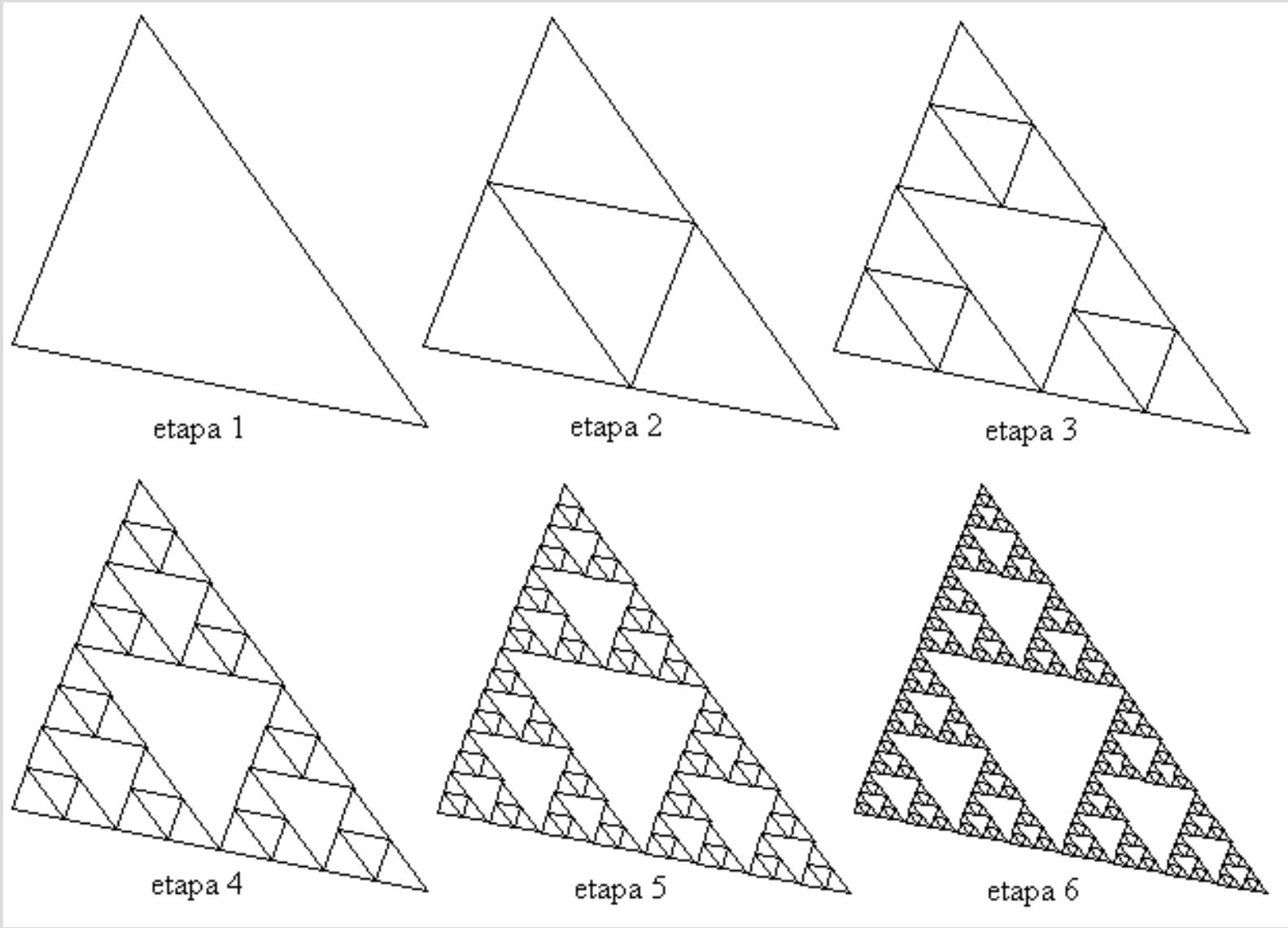
- Definición

$$F(n) = \begin{cases} 0 & \text{si } n = 0; \\ 1 & \text{si } n = 1; \\ F(n-1) + F(n-2) & \text{si } n > 1. \end{cases}$$

- En scheme:

```
(define (fibo n)
  (cond
    [(= n 0) 0]
    [(= n 1) 1]
    [else
     (+ (fibo (- n 1))
        (fibo (- n 2)))]))
```

Ejemplos de funciones recursivas: Triangulo



Funciones Recursivas

- Todas las funciones recursivas tienen las siguientes características:
  - tienen uno o más casos base
  - tienen un caso recursivo: se llama a la función

  Los ejemplos anteriores tienen caso base y caso recursivo?
  - Potencias de 2
  - Factorial
  - Fibonacci
  - Triangulo

Funciones Recursivas

- Cómo todas las funciones recursivas tienen la misma estructura, el cuerpo de la función (en scheme) será un condicional.
- ¿Cuantas condiciones debo poner?
  - Una por cada caso base
  - Una por el caso recursivo

  Ver ejemplos

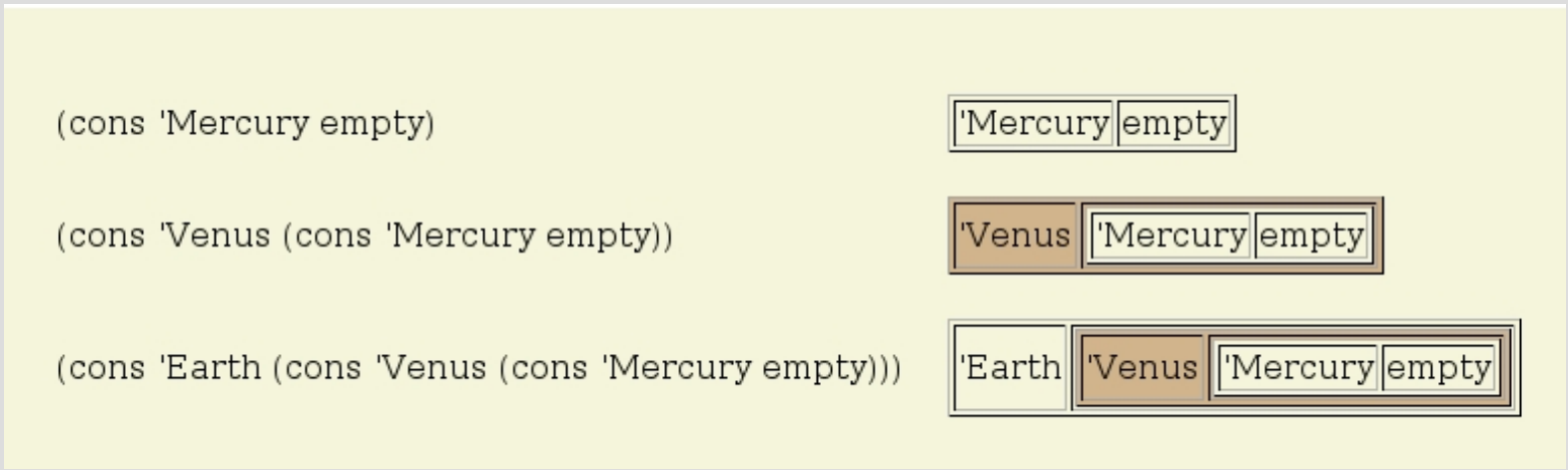
Ejercicio

- Desarrolle una función recursiva llamada *sumatoria* usando la siguiente definición:

$$\sum(n) = \begin{cases} n & \text{si n=0} \\ n + \sum(n-1) & \text{para n>0 en los naturales} \end{cases}$$

Datos Recursivos

- Listas: una lista es un tipo de datos recursivo porque se define en términos de sí misma:
  - Una lista es:
    - empty (lista vacía)
    - (cons elemento lista) en donde elemento puede ser de cualquier tipo y lista es una lista.
  - ejemplo:

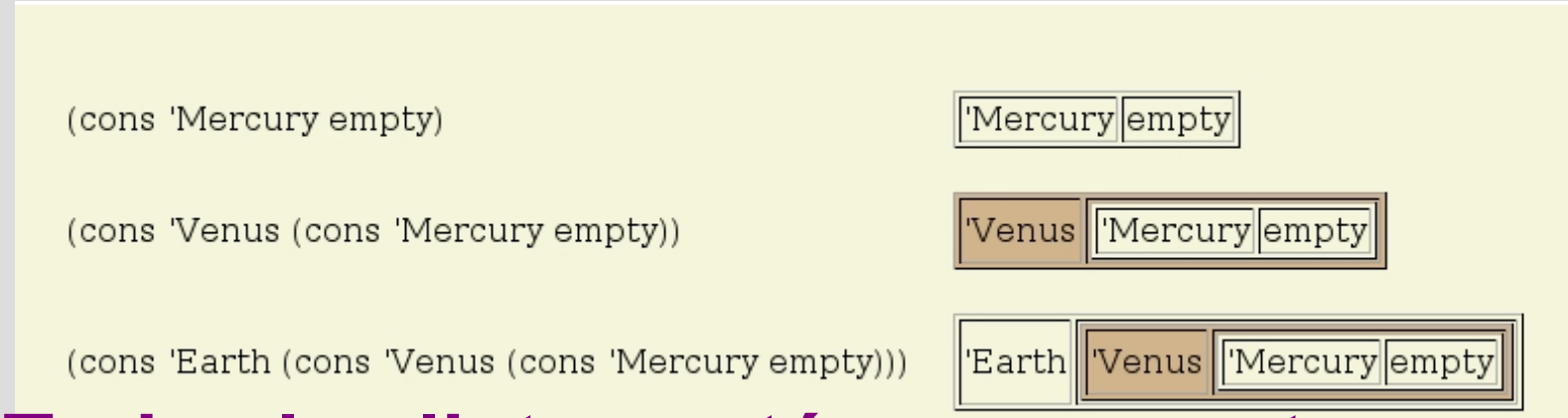


El operador CONS

- El operador cons nos permite construir listas. el operador cons se utiliza de la siguiente forma:

(cons cabeza cola)

En donde cabeza es un elemento y cola es una lista



Todas las listas están compuestas por cabeza y cola, excepto la lista vacía

Operadores first y rest

- Similar a los selectores de las estructuras, las listas tienen dos operaciones que permiten consultar el contenido de la cabeza y de la cola
- first: retorna el primer elemento de la lista
- rest: retorna la cola de una lista (que es una lista)
- Ejemplo
  - (first (cons 'venus (cons 'mercury empty))) retorna 'venus
  - (rest (cons 'venus (cons 'mercury empty))) retorna (cons 'mercury empty)