

ΔΙΑΙΠΕΙ ΚΑΙ ΒΑΣΙΛΕΥΕ

El título de esta hoja es una estrategia muy socorrida para el diseño de algoritmos, que consiste en construir la solución de un problema a partir de las de varios casos¹ más sencillos del mismo problema (típicamente, con datos de entrada de tamaño menor).

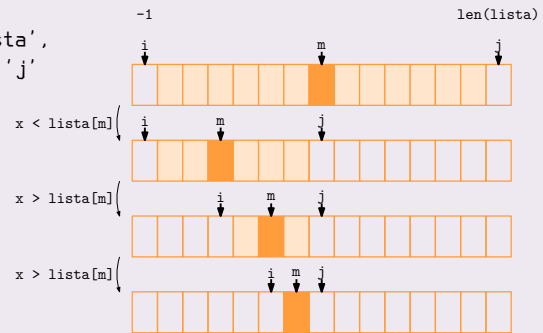
De esta manera, un algoritmo de este tipo consta de tres etapas:

- **Descomposición** del problema en subproblemas homogéneos,
- **resolución** (recursiva) de estos y
- **combinación** de sus soluciones para alcanzar la del problema original.

La búsqueda por bipartición (en un *array* **ordenado** de números) constituye un ejemplo archiconocido de esta estrategia. Se distingue entre los algoritmos de la categoría general que estamos tratando en que caso paso recursivo es una reducción a un solo subproblema ($a = 1$).

```
def busca(x, lista):  
    return busca_aux(x, lista, 0, len(lista))
```

```
def busca_aux(x, lista, i, j):  
    # Busca el elemento 'x' en la lista ordenada 'lista',  
    # a partir del índice 'i' (incluido) y antes del 'j'  
    # (excluido).  
  
    if i >= j:                # rango vacío  
        return None  
    else:  
        m = (i + j - 1) >> 1  
        if lista[m] == x:  
            return m  
        elif lista[m] < x:  
            return busca_aux(x, lista, m + 1, j)  
        else:  
            return busca_aux(x, lista, i, m)
```



- 1) Traduzca el código dado de python a Scheme, utilizando list, car y cdr y que el código sea también válido si se utiliza el tipo vector.
- 2) Calcule una cota superior para la cantidad de llamadas a *busca_aux* que realiza el algoritmo en función de la longitud de la lista de entrada. Acote la complejidad espacial y temporal de la búsqueda por bipartición.
- 3) Suponiendo que la extensión de la representación binaria de cada uno de los números de la lista está acotada por $\log_2 n$, deduce una cota superior asintótica para la complejidad computacional del algoritmo en términos de operaciones bit.
- 4) ¿Se puede aprovechar el teorema maestro para las dos preguntas anteriores?

¹En cantidad a , según la notación del *teorema maestro*.

Teorema maestro (CORMEN *et al.*, § 4.5)

$$T(n) = a \cdot T(\lfloor n/b \rfloor) + f(n) \quad \vee \quad T(n) = a \cdot T(\lceil n/b \rceil) + f(n)$$

$$\Downarrow$$

$$\left\{ \begin{array}{ll} \text{Si } f(n) \in O(n^d), \text{ con } d < \log_b a, & T(n) \in \Theta(n^{\log_b a}); \\ \text{si } f(n) \in \Theta(n^d), \text{ con } d = \log_b a, & T(n) \in \Theta(n^{\log_b a} \cdot \log n); \\ \text{si } f(n) \in \Omega(n^d), \text{ con } d > \log_b a, & T(n) \in \Theta(f(n)). \end{array} \right.$$

y $\exists c < 1, n_0 \in \mathbb{N}, \forall n \geq n_0, a \cdot f(\lceil n/b \rceil) \leq c \cdot f(n)$,

- 5) Escribe una versión iterativa del algoritmo de búsqueda por bipartición, dado arriba en forma recursiva.