Algoritmos de búsqueda

ICM.1er. Cuatrimestre del 2018

Ejercicio 1

- Implementar en python las siguientes funciones de búsqueda, asumiendo que las listas que recibe como parámetro no contienen elementos repetidos.

a) Def busquedalineal(x,L): #devuelve -1 si x no está en L

b) Def busquedabinaria(x,L) #devuelve -1 si x no está en L y se
#asumen que L está ordenada.

c) Comparar la performance temporal entre ambos.

Algo más de Python

- import time # les permite importar operaciones para medir el tiempo.
- t0 = time.clock() # guarda en la variable t0 el estado del reloj interno.

- Import math # les permite incluir varias operaciones matemáticas.

- round() # redondea correctamente al número más próximo.

```
>>> round(4.35)
4
>>> round(4.35, 1)
4.3
>>> round(154,-1)
150
```

Algo más de Python (2)

- floor() y ceil() #son funciones de redondeo pero de un solo parámetro.

```
>>> floor(5 / 2)
2
>>> from math import ceil
>>> ceil(5 / 2)
3
```

Más información en http://www.mclibre.org/consultar/python/lecciones/

Ejercicio 2

- Implementar en Python una función en Python que dada una lista ordenada L y un elemento x devuelve la cantidad de apariciones de x en L en O(log(|L|).

```
def cantapariciones(x,L):          #pre: L está ordenada.
```

Ejercicio 3

- Implementar en Python una función que dadas dos listas **L1 y L2, ambas** sin elementos repetidos, devuelva la cantidad de elementos de **L1 que aparecen en L2:**

```
def enambaslistas(L1, L2):          #pre: L1 y L2 sin repetidos.
def enambaslistasord(L1, L2):       #pre: L1 y L2 ordenadas y sin
                                    #elementos repetidos.
                                    #con O(|L1| + |L2|)
```