**TAREA 45: DICCIONARIO**

* **BUSQUEDA SECUENCIAL:** Consiste en ir comparando el elemento a buscar con cada elemento del vector hasta encontrarlo hasta que se llegue al final, esto hace que la búsqueda sea secuencialmente. La existencia se puede asegurar cuando el elemento es localizado, pero no podemos asegurar la no existencia hasta no haber analizado todos los elementos del vector. Se utiliza sin importar si el vector está ordenado o no.
* **BUSQUEDA BINARIA:** Se utiliza cunado el vector en el que queremos determinar la existencia de un elemento está previamente ordenado. Este algoritmo reduce el tiempo de búsqueda considerablemente, ya que disminuye exponencialmente el número de iteraciones necesarias. En el peor de los casos el número máximo de comparaciones es [log2n+1], donde n es el número de elementos del vector.

Para implementar el algoritmo se compara el elemento a buscar con el elemento central del vector. Si el valor es mayor que el elemento buscado, se repite el procedimiento, pero siendo ahora el valor final el valor central – 1. De esta manera obtenemos cada vez intervalos mas pequeños, hasta tener un intervalo indivisible, que será el valor buscado o devolverá que no existe el valor buscado en el vector.

**Implementación recursiva:**

**def** bin(a,x,low,hi):

ans = -1

**if** low==hi: ans = -1

**else**:

mid = (low+((hi-low)//2))

**if** x < a[mid]: ans = bin(a,x,low,mid)

**elif** x > a[mid]: ans = bin(a,x,mid+1,hi)

**else**: ans = mid

**return** ans

*# Así se hace el llamado: print(bin(Lista, numero\_a\_buscar, 0, len(Lista)))*

*# Retorna el índice que coincide con 'numero\_a\_buscar', si no está retorna -1*

*# Tiempo: (log n)*

**Implementación iterativa:**

**def** bin(a, c):

ans = -1

**if** a[0] >= c: ans = -1

**else**:

low, hi = 0, len(a)

**while** low+1 != hi:

mid = low + ((hi-low)//2)

**if** a[mid] < c: low = mid

**else**: hi = mid

ans = low

**return** ans

*# Así se hace el llamado: print(bin(lista(), numero\_a\_buscar))*

*# Retorna el índice que coincide con 'numero\_a\_buscar', si no está retorna -1*

*# Tiempo: (log n)*