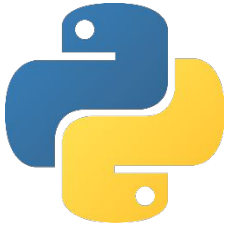
Universidad Nacional de Jujuy
Facultad de ingeniería
Metodología de la Programación

Clase 03 - Ordenación



Samuel Franco – José Zapana

Contenido

- Clasificación de los métodos de ordenación
- Método burbuja
- Método por inserción
- Método por selección
- Método interno de python sorted()

Introducción

- Ordenación o clasificación es el proceso de reordenar un conjunto de objetos en un orden específico. El propósito de la ordenación es facilitar la búsqueda de elementos en el conjunto ordenado.
- Existen muchos algoritmos de ordenación, siendo la diferencia entre ellos las ventajas de unos sobre otros en la eficiencia en tiempo de ejecución.

Clasificación de los métodos

- **Interno**: en la memoria del ordenador.
- **Externo**: en un lugar externo como un disco duro.
- **Natural**: tiempo mínimo si la entrada está casi-ordenada.
- **Estable:** mantiene el orden relativo que tenían originalmente los elementos con claves iguales:
  (4, 1) (3, 7) (3, 1) (5, 6) (original)
  (3, 7) (3, 1) (4, 1) (5, 6) (orden mantenido)
  (3, 1) (3, 7) (4, 1) (5, 6) (orden cambiado)
- **Complejidad:** (peor caso, caso promedio y mejor caso), se usa la notación O(n). Se puede aprovechar la estructura de las claves o no. O(n log n), $O(n^2)$, O(kn) donde k es el tamaño del espacio de claves.
- Otros: memoria (in situ), recursividad, paralelo, adaptativo

Ordenación burbuja

```
def bubbleSort(lst):
    n = len(lst)
    for i in range(n):
        for j in range(0, n-1):
            if lst[j] > lst[j+1]:
                lst[j], lst[j+1] = lst[j+1], lst[j]
    return lst
```

6 5 3 1 8 7 2 4

Ordenación por inserción

```
def insertionSort(lst):
    for step in range(1, len(lst)):
        key = lst[step]
        j = step - 1
        while j >= 0 and key < lst[j]:
            lst[j + 1] = lst[j]
            j = j - 1
        lst[j + 1] = key
    return lst
```

6 5 3 1 8 7 2 4

Ordenación por selección

```
def selectionSort(lst):
    n = len(lst)
    for step in range(n):
        j = step
        for i in range(step + 1, n):
            if lst[i] < lst[j]:
                j = i
        lst[step], lst[j] = lst[j], lst[step]
    return lst
```

Recursos

https://docs.python.org/es/3/howto/sorting.html
https://en.wikipedia.org/wiki/Sorting_algorithm
https://en.wikipedia.org/wiki/Timsort#Minimum_size_.28minrun.29