Universidad Nacional Autónoma de México

Facultad de Ciencias

Análisis de Algoritmos

Práctica 03
Ordenamientos

María de Luz Gasca Soto
Teresa Becerril Torres
Rodrigo Fernando Velázquez Cruz

Autor:

Hermes Alberto Delgado Díaz
319258613

15 de noviembre del 2024

Lenguaje de programación utilizado

Para esta práctica se utilizo Python en su versión **Python 3.12.6**
Además es necesario tener instalada la biblioteca `colorama`
En caso de no tenerla instalada, en terminal:

```
?- pip install colorama
```

Comandos

Para ejecutar el programa

```
?- Python ordenamiento.py
```

Experimentación

Para esta práctica se realizo un epxperimento donde se ejecuto los algoritmos `Merger Sort`, `Local Insertion Sort` y `Insertion Sort`, para distintos tamaños de una secuencia con dos valores de k distintos para cada ejemplar y contar el número de operaciones elementales que toma cada ejecución.
Los tamaños en este experimento son $n = 10000, 5000, 2500, 1000$ y k serán $k = 3, 6$

n	k	Número de Operaciones		
		Merge Sort	Local Insertion Sort	Insertion Sort
1000	3	9071	253495	249999
1000	6	8911	254239	249252
2500	3	25110	1571245	1562499
2500	6	24778	1573114	1560627
5000	3	54012	6267494	6250000
5000	6	53263	6271243	6246248
10000	3	115478	25034995	24999999
10000	6	114063	25042489	24992502

Con los datos anteriores, se puede notar que en secuencias con datos grandes `Merge Sort` es el más eficiente, también se puede ver que mientras k sea más grande, se reducirá el número de operaciones en el algoritmo.

Conclusión

Para datos de gran tamaño el algoritmo `Merge Sort` es el más eficiente, después `Local Insertion Sort`, y el peor es `Insertion Sort`. Mientras k sea más grande, el número de operaciones en el algoritmo reduce.
Para datos de menor tamaño, el algoritmo más eficiente es `Insertion Sort`.