Sesión 1.0

Algoritmia-Prácticas de Laboratorio

Contenido

[Tiempos del algoritmo A1 en Python 2](#_Toc159144010)

[Comparación del Algoritmo A1 en dos ordenadores diferentes 2](#_Toc159144011)

[Implementación del Algoritmo 3](#_Toc159144012)

[Optimización del Algoritmo 4](#_Toc159144013)

# Tiempos del algoritmo A1 en Python

En esta primera parte, se analizara los tiempos de ejecución del algoritmo A1 para diferentes valores de n

|  |  |
| --- | --- |
| Elementos | Tiempo(ms) |
| 10000 | 2315 |
| 20000 | 9316 |
| 40000 | 37937 |
| 80000 | FdT |
| 160000 | FdT |
| 320000 | FdT |
| 640000 | FdT |
| 1280000 | FdT |

Con los tiempos obtenidos podemos concluir que al duplicar la cantidad de elemento es tiempo se cuatriplica, estos tiempos están acorde con la complejidad del algoritmo la cual es O(n^2).

# Comparación del Algoritmo A1 en dos ordenadores diferentes

Este ejercicio se analizará como la potencia del ordenador influye en los resultados.

Ordenador 1: Ordenador 2:

CPU 🡪 Intel(R) Core(TM) i7-10510U CPU 🡪Intel® Core™ i7-1355U

Memoria 🡪 16GB Memoria 🡪 16GB

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Elementos | Tiempo(ms) |  |  | Elementos | Tiempo(ms) |
| 10000 | 2315 |  |  | 10000 | 1409 |
| 20000 | 9316 |  |  | 20000 | 6012 |
| 40000 | 37937 |  |  | 40000 | 29632 |
| 80000 | FdT |  |  | 80000 | FdT |
| 160000 | FdT |  |  | 160000 | FdT |
| 320000 | FdT |  |  | 320000 | FdT |
| 640000 | FdT |  |  | 640000 | FdT |
| 1280000 | FdT |  |  | 1280000 | FdT |

Como podemos ver la potencia del ordenador afecta al tiempo de procesamiento de los datos

# Implementación del Algoritmo

En este apartado vamos a analizar como el lenguaje de programación y sus optimización afectan a los resultados, para ello vamos a tomar los tiempos del algoritmo A1 en Python y compararlos con los tiempos obtenidos en Java

|  |  |  |
| --- | --- | --- |
|  | Python | Java(Sin optimizar) |
| Elementos | Tiempo(ms) | Tiempo(ms) |
| 10000 | 2315 | 738 |
| 20000 | 9316 | 3385 |
| 40000 | 37937 | 40043 |
| 80000 | FdT | FdT |
| 160000 | FdT | FdT |
| 320000 | FdT | FdT |
| 640000 | FdT | FdT |
| 1280000 | FdT | FdT |

Como se puede apreciar el lenguaje afecta a los resultados obtenidos

# Optimización del Algoritmo

En este apartado analizaremos como diferentes implementaciones del algoritmo influyen en los tiempos resultantes.

Para ello analizaremos el algoritmo A1, en diferentes versiones y diferentes lenguajes

Python:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | A1 | A2 | A3 | A4 |
| Elementos | Tiempo(ms) | Tiempo(ms) | Tiempo(ms) | Tiempo(ms) |
| 10000 | 2315 | 274 | 134 | 41 |
| 20000 | 9316 | 1008 | 498 | 168 |
| 40000 | 37937 | 3675 | 1871 | 571 |
| 80000 | FdT | 13864 | 7157 | 2060 |
| 160000 | FdT | 53153 | 31538 | 6394 |
| 320000 | FdT | FdT | FdT | 18150 |
| 640000 | FdT | FdT | FdT | 65286 |
| 1280000 | FdT | FdT | FdT | FdT |

Java (Sin optimización):

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | A1 | A2 | A3 | A4 |
| Elementos | Tiempo(ms) | Tiempo(ms) | Tiempo(ms) | Tiempo(ms) |
| 10000 | 738 | 151 | 78 | 71 |
| 20000 | 3385 | 544 | 318 | 258 |
| 40000 | 40043 | 2083 | 1261 | 919 |
| 80000 | FdT | 7001 | 4994 | 3248 |
| 160000 | FdT | 25332 | 14925 | 11015 |
| 320000 | FdT | FdT | 51535 | 31509 |
| 640000 | FdT | FdT | FdT | FdT |
| 1280000 | FdT | FdT | FdT | FdT |

Java(con optimización):

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | A1 | A2 | A3 | A4 |
| Elementos | Tiempo(ms) | Tiempo(ms) | Tiempo(ms) | Tiempo(ms) |
| 10000 | 113 | 84 | 49 | 23 |
| 20000 | 463 | 293 | 174 | 47 |
| 40000 | 1735 | 1066 | 643 | 132 |
| 80000 | 7253 | 3984 | 2441 | 378 |
| 160000 | 23828 | 13646 | 9635 | 1217 |
| 320000 | FdT | 50911 | 30270 | 3568 |
| 640000 | FdT | FdT | FdT | 12378 |
| 1280000 | FdT | FdT | FdT | 37792 |

Como podemos apreciar en la imagen, la forma de implementar el algoritmo afecta al tiempo de ejecución, también podemos concluir que el lenguaje afecta a los tiempo y como se puede observar en la tablas obtenidas anteriormente, el lenguaje Java, y con las optimizaciones de la máquina virtual, hace que los tiempos sea menores que en Python