**ANÁLISIS DEL RETO**

Santiago Toro Márquez, 202013718, s.torom@uniandes.edu.co

Carlos David Rodriguez, 202310137, cd.rodriguezb1@uniandes.edu.co

Michelle Alejandra Casallas, 201813052, ma.casallas@uniandes.edu.co

# **Requerimiento 1**

## **Descripción**

Este requerimiento cumple con la necesidad de ingresar un país y un nivel de experiencia para mostrar las ofertas más recientes.

def sort\_country\_experience(catalog, pais, experiencia): # REQUERIMIENTO 1 --------------------------------------------------------------------------

    """

    Función que soluciona el requerimiento 1

    """

    # TODO: Realizar el requerimiento 1

    jobsf = lt.newList("ARRAY\_LIST")

    jobs\_p = me.getValue(mp.get(catalog['mapa\_paises'], pais))["jobs"]    # pais

    jobs\_e = me.getValue(mp.get(catalog['mapa\_experiencia'], experiencia))["jobs"]    # experiencia

    if (lt.size(jobs\_p) <= lt.size(jobs\_e)): # Revisar camino con menor cantidad de comparaciones

        for job in lt.iterator(jobs\_e):

            if (flt\_experiencia(job, experiencia) == True): # REALIZAR FUNCIONES DE FILTRADO Y VERIFICAR OUTPUT

                lt.addLast(jobsf,job)

    else:

        for job in lt.iterator(jobs\_e):

            if (flt\_pais(job, pais) == True):

                lt.addLast(jobsf, job)

|  |  |
| --- | --- |
| **Entrada** | La estructura de datos a analizar, el código del país y la experiencia deseada. |
| **Salidas** | Ofertas de trabajo filtradas por país y experiencia |
| **Implementado (Sí/No)** | Implementado – Est 1 |

## **Análisis de complejidad**

Análisis de complejidad de cada uno de los pasos del algoritmo

|  |  |
| --- | --- |
| **Pasos** | **Complejidad** |
| Paso 1 | O(...) |
| Paso 2 | O(...) |
| Paso …. | O(...) |
| ***TOTAL*** | ***O(...)*** |

## **Pruebas Realizadas**

Para las pruebas realizadas, se utilizan los csv de menor cantidad de datos, desde el 10 hasta el 100% de los datos cada 10%. Se registra el tiempo en [ms] y memoria utilizada en [kB] para obtener un O(n) para poder comparar el Big O teórico calculado contra el actual. Para los datos de entrada, se utilizaron las mismas entradas en la función, correspondientes a país = PL y experiencia = mid e imprimir las 3 ofertas más recientes.

|  |  |
| --- | --- |
| Procesadores | AMD Ryzen 7 4800HS with Radeon Graphics |
| Memoria RAM | 8 GB |
| Sistema Operativo | Windows 10 |

|  |  |  |  |
| --- | --- | --- | --- |
| **Entrada** | **Tiempo (ms)** | **Memoria (kB)** |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

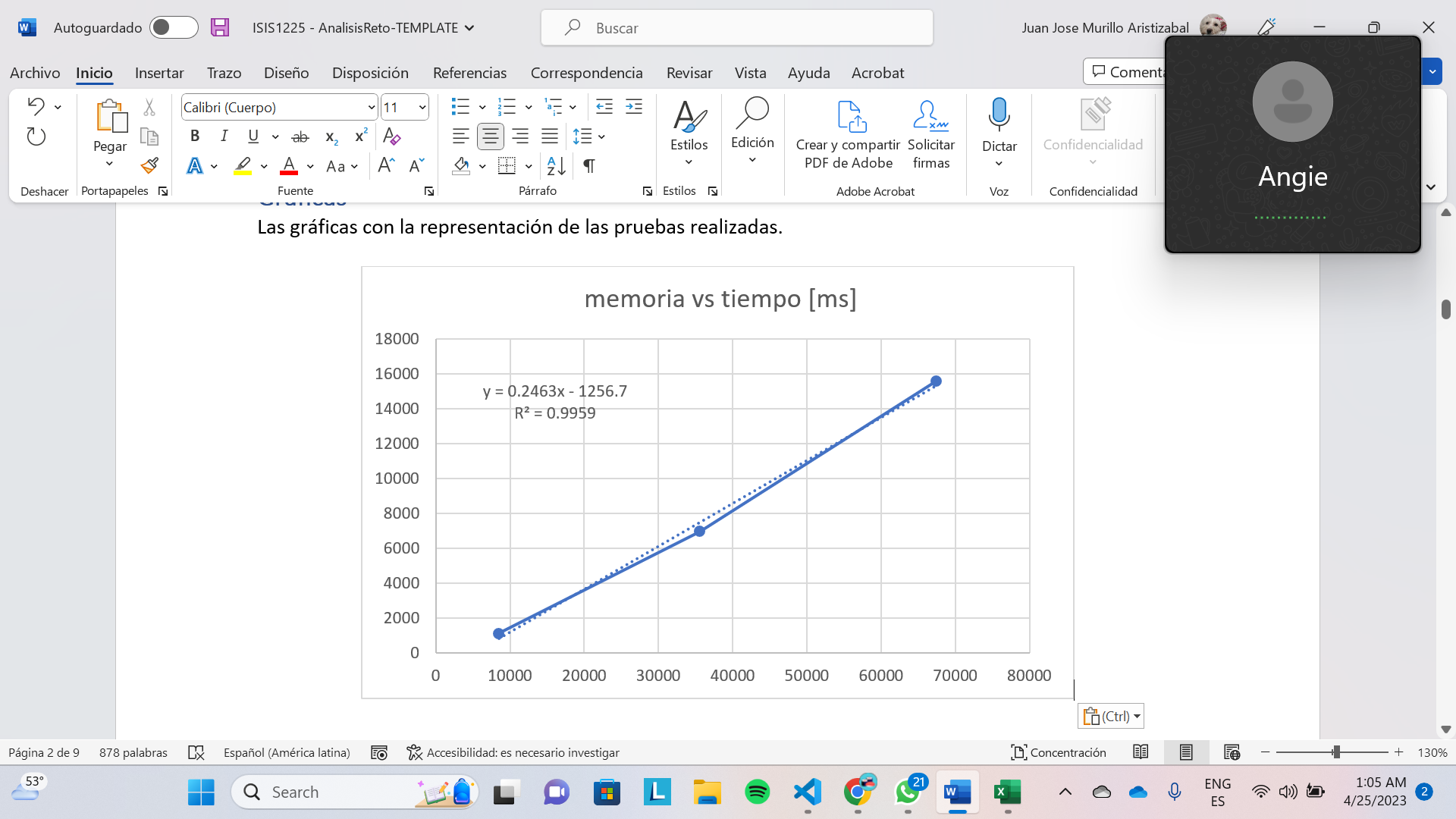
### **Tablas de datos**

Las tablas con la recopilación de datos de las pruebas.

|  |  |  |
| --- | --- | --- |
| Cantidad de Datos | kB | Tiempo [ms] |
| Small | 8512.214 | 1129.532 |
| 5% | 35478.420 | 6859.736 |
| 10% | 68352.329 | 15623.843 |

### **Graficas**

Las gráficas con la representación de las pruebas realizadas.



## **Análisis**

Análisis de resultados de la implementación, tener cuenta las pruebas realizadas y el analisis de complejidad.

# **Requerimiento Ejemplo**

## **Descripción**



Este requerimiento se encarga de retornar un dato de una lista dado su ID. Lo primero que hace es verificar si el elemento existe. Dado el caso que exista, retorna su posición, lo busca en la lista y lo retorna. De lo contrario, retorna None.

|  |  |
| --- | --- |
| **Entrada** | Estructuras de datos del modelo, ID. |
| **Salidas** | El elemento con el ID dado, si no existe se retorna None |
| **Implementado (Sí/No)** | Si. Implementado por Juan Andrés Ariza |

## **Análisis de complejidad**

Análisis de complejidad de cada uno de los pasos del algoritmo

|  |  |
| --- | --- |
| **Pasos** | **Complejidad** |
| Buscar si el elemento existe (isPresent) | O(n) |
| Obtener el elemento (getElement) | O(1) |
| ***TOTAL*** | ***O(n)*** |

## **Pruebas Realizadas**

Las pruebas realizadas fueron realizadas en una maquina con las siguientes especificaciones. Los datos de entrada fueron el ID 1.

|  |  |
| --- | --- |
| Procesadores | AMD Ryzen 7 4800HS with Radeon Graphics |
| Memoria RAM | 8 GB |
| Sistema Operativo | Windows 10 |

|  |  |
| --- | --- |
| **Entrada** | **Tiempo (ms)** |
| small | 0.05 |
| 5 pct | 0.33 |
| 10 pct | 1.28 |
| 20 pct | 2.54 |
| 30 pct | 4.98 |
| 50 pct | 7.51 |
| 80 pct | 13.81 |
| large | 25.97 |

### **Tablas de datos**

Las tablas con la recopilación de datos de las pruebas.

|  |  |  |
| --- | --- | --- |
| **Muestra** | **Salida** | **Tiempo (ms)** |
| small | Dato1 | 0.05 |
| 5 pct | Dato2 | 0.33 |
| 10 pct | Dato3 | 1.28 |
| 20 pct | Dato4 | 2.54 |
| 30 pct | Dato5 | 4.98 |
| 50 pct | Dato6 | 7.51 |
| 80 pct | Dato7 | 13.81 |
| large | Dato8 | 25.97 |

### **Graficas**

Las gráficas con la representación de las pruebas realizadas.

## **Análisis**

A pesar de que obtener un elemento en un *ArrayList,* dada su posición, tiene complejidad constante, la implementación de este requerimiento tiene un orden lineal O(n). Esto debido a que, lo primero que se hace es verificar si el elemento hace parte de la lista. Específicamente, a la hora de buscar un elemento en una lista, en el peor de los casos es necesario recorrer toda la lista, es decir, complejidad lineal.

Este comportamiento se puede evidenciar experimentalmente en la gráfica. Ya que, gracias a que los datos no se encuentran tan dispersos con respecto a la línea de tendencia, la curva coincide con el comportamiento lineal esperado.

# **Requerimiento <<2>>**

## Listar las últimas ofertas de trabajo por empresa y ciudad



Breve descripción de como abordaron la implementación del requerimiento

|  |  |
| --- | --- |
| **Entrada** | El número (N) de ofertas a listar, nombre completo de la empresa a consultar, ciudad de la oferta (nombre de la ciudad) |
| **Salidas** | El total de ofertas ofrecida por la empresa y ciudad. |
| **Implementado (Sí/No)** | Si se implementó |

## **Análisis de complejidad**

Análisis de complejidad de cada uno de los pasos del algoritmo

|  |  |
| --- | --- |
| **Pasos** | **Complejidad** |
| Se accede a los trabajos asociados a una ciudad y una empresa, lo cual implica acceder a las respectivas estructuras de datos (mapa\_ciudades y mapa\_empresas) para buscar las ciudades y empresas específicas. | O(1) |
| Se itera sobre la lista de trabajos de la ciudad o de la empresa, dependiendo de cuál sea más pequeña. | O(n) |
| Se filtran los trabajos según la empresa o la ciudad, lo que implica verificar una condición en cada trabajo. | O(n) |
| ***TOTAL*** | ***O(n+m log m)*** |

## **Pruebas Realizadas**

|  |  |
| --- | --- |
| Procesadores | AMD Ryzen 7 4800HS with Radeon Graphics |
| Memoria RAM | 8 GB |
| Sistema Operativo | Windows 10 |

|  |  |
| --- | --- |
| **Entrada** | **Tiempo (ms)** |
| small | 0.71 |
| 5 pct | 0.72 |
| 10 pct | 2.15 |
| 20 pct | 4.10 |
| 30 pct | 4.39 |
| 50 pct | 4.69 |
| 80 pct | 4.86 |
| large | 5.24 |

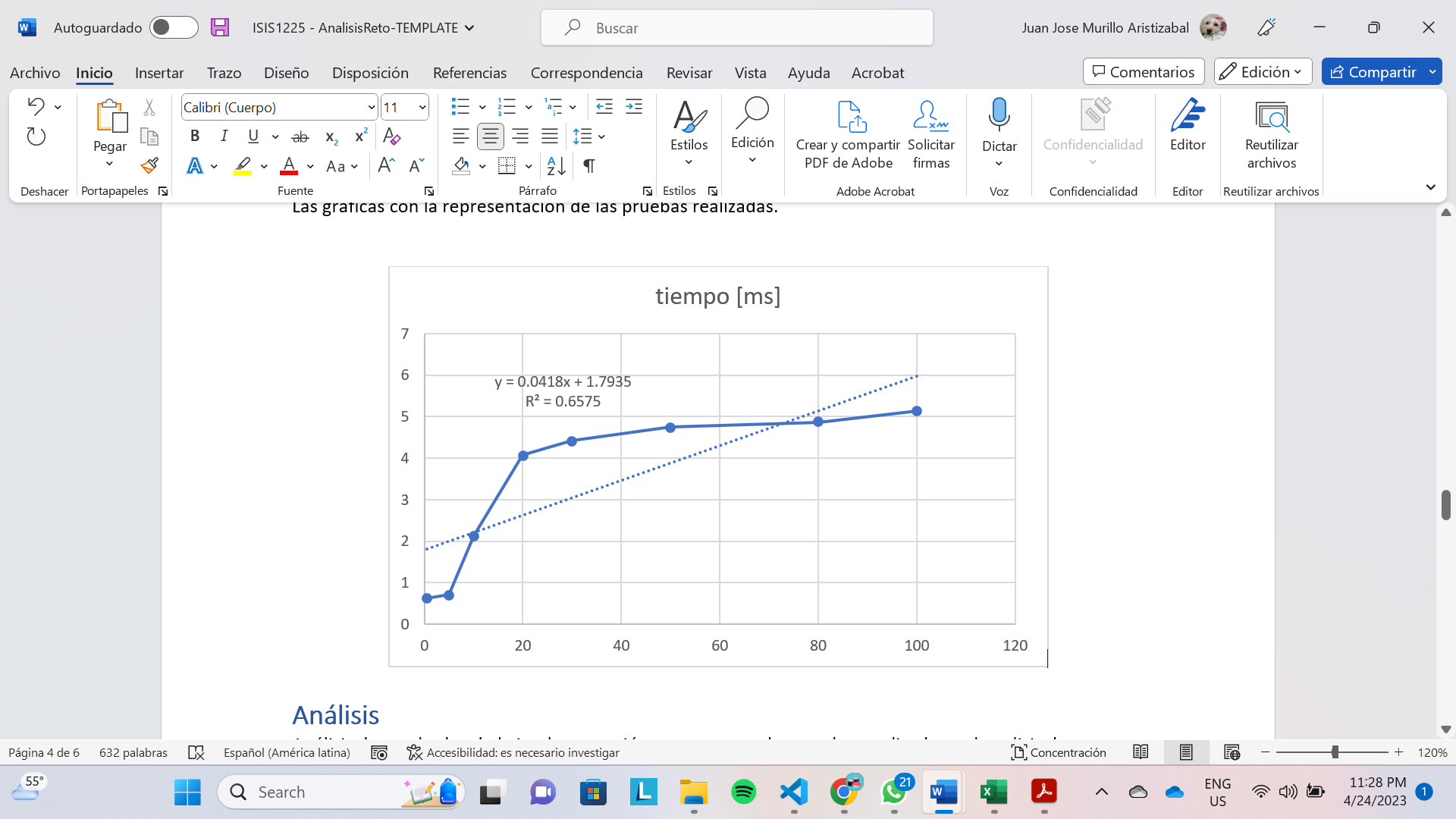
### **Tablas de datos**

Las tablas con la recopilación de datos de las pruebas.

|  |  |  |
| --- | --- | --- |
| **Muestra** | **Salida** | **Tiempo (ms)** |
| small | Dato1 | 0.71 |
| 5 pct | Dato2 | 0.72 |
| 10 pct | Dato3 | 2.15 |
| 20 pct | Dato4 | 4.10 |
| 30 pct | Dato5 | 4.39 |
| 50 pct | Dato6 | 4.69 |
| 80 pct | Dato7 | 4.86 |
| large | Dato8 | 5.24 |

### **Graficas**

Las gráficas con la representación de las pruebas realizadas.



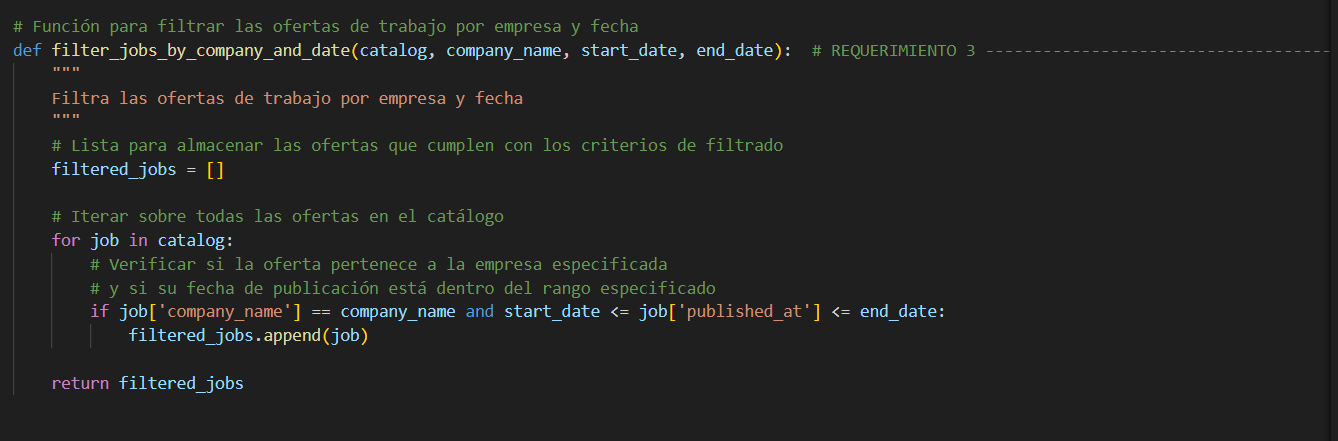
## **Análisis**

Los resultados de las pruebas muestran que el algoritmo es eficiente para conjuntos de datos pequeños y medianos, pero puede enfrentar desafíos de rendimiento con conjuntos de datos más grandes. Aunque los tiempos de ejecución son estables a medida que aumenta el tamaño de los datos, podría ser beneficioso investigar formas de mejorar el rendimiento del algoritmo, especialmente en lo que respecta al ordenamiento de los datos seleccionados.

# **Requerimiento <<3>>**

## **Descripción**

consultar las ofertas de trabajo publicadas por una empresa en un rango de fechas dado.



|  |  |
| --- | --- |
| **Entrada** | Nombre de la empresa, la fecha inicial del periodo a consultar y la fecha final del periodo a consultar |
| **Salidas** | Buscamos mostrar un resumen de ofertas de trabajo, incluyendo cuántas hay para cada nivel de experiencia (junior, mid y senior), y una lista detallada de todas las ofertas, ordenadas por fecha y país. Para cada oferta, se mostrará la fecha, título, nivel de experiencia requerido, ciudad y país de la empresa, tamaño de la empresa, tipo de lugar de trabajo y si se contratan ucranianos. |
| **Implementado (Sí/No)** | Si (Carlos Rodriguez) |

## **Análisis de complejidad**

Análisis de complejidad de cada uno de los pasos del algoritmo

|  |  |
| --- | --- |
| **Pasos** | **Complejidad** |
| Si el catálogo contiene n ofertas, la complejidad de esta iteración es O(n), donde n es el número total de ofertas en el catálogo. | O(n) |
| Dentro del bucle, se verifica si cada oferta pertenece a la empresa especificada y si su fecha de publicación está dentro del rango especificado. | O(1) |
| ***TOTAL*** | ***O(n)*** |

## **Pruebas Realizadas**

Las pruebas realizadas fueron realizadas en una maquina con las siguientes especificaciones. Los datos de entrada fueron

|  |  |
| --- | --- |
| Procesadores | AMD Ryzen 5 11400HS with Radeon Graphics |
| Memoria RAM | 8 GB |
| Sistema Operativo | Windows 11 |

|  |  |
| --- | --- |
| **Entrada** | **Tiempo (ms)** |
| small | 5.98 |
| 5 pct | 24.73 |
| 10 pct | 34.74 |
| 20 pct | 67.02 |
| 30 pct | 97.92 |
| 50 pct | 164.12 |
| 80 pct | 1372.3 |
| large | 1853.83 |

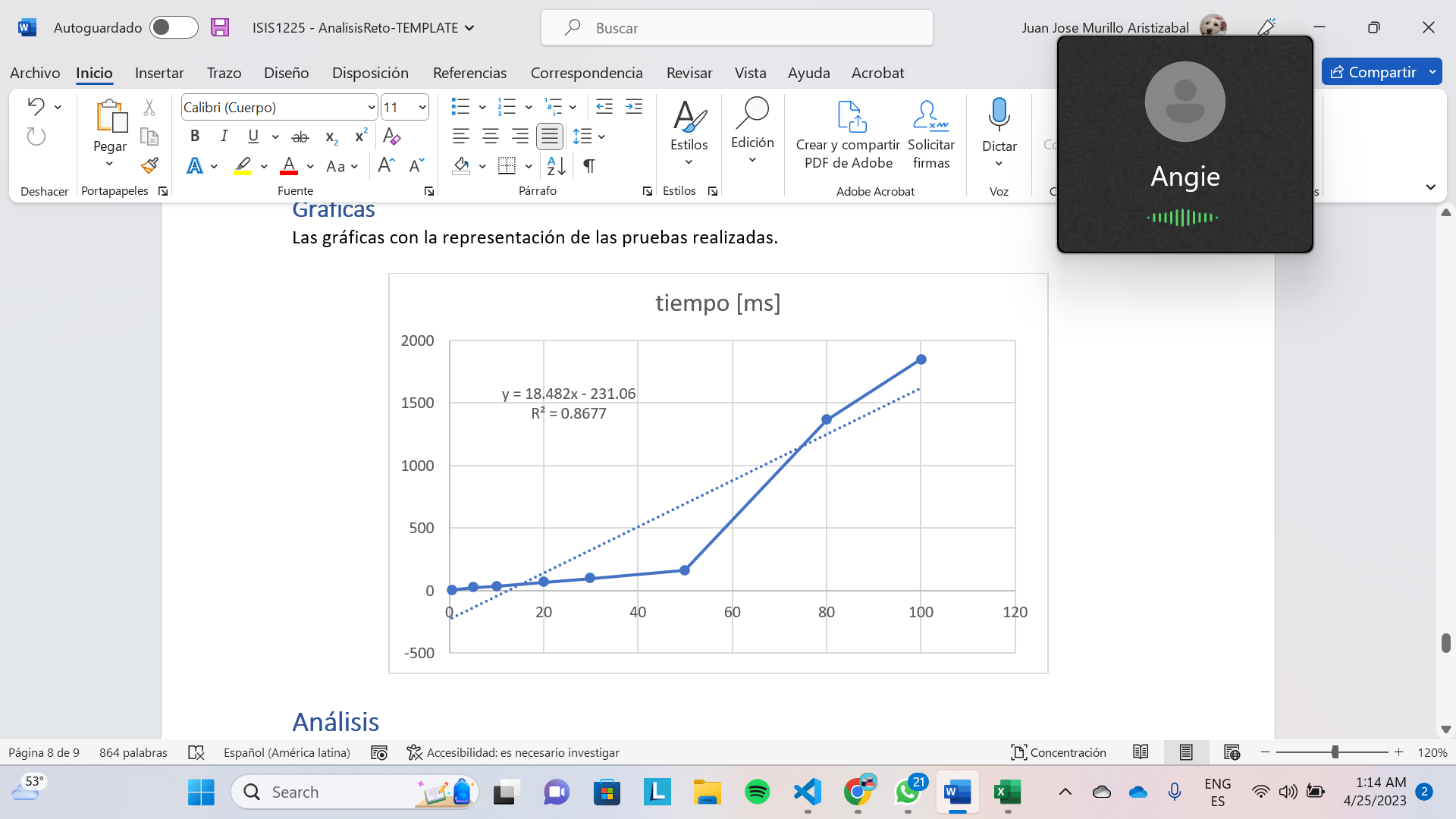
### **Tablas de datos**

Las tablas con la recopilación de datos de las pruebas.

|  |  |  |
| --- | --- | --- |
| **Muestra** | **Salida** | **Tiempo (ms)** |
| small | Dato1 | 5.98 |
| 5 pct | Dato2 | 24.73 |
| 10 pct | Dato3 | 34.74 |
| 20 pct | Dato4 | 67.02 |
| 30 pct | Dato5 | 97.92 |
| 50 pct | Dato6 | 164.12 |
| 80 pct | Dato7 | 1372.3 |
| large | Dato8 | 1853.83 |

### **Graficas**

Las gráficas con la representación de las pruebas realizadas.



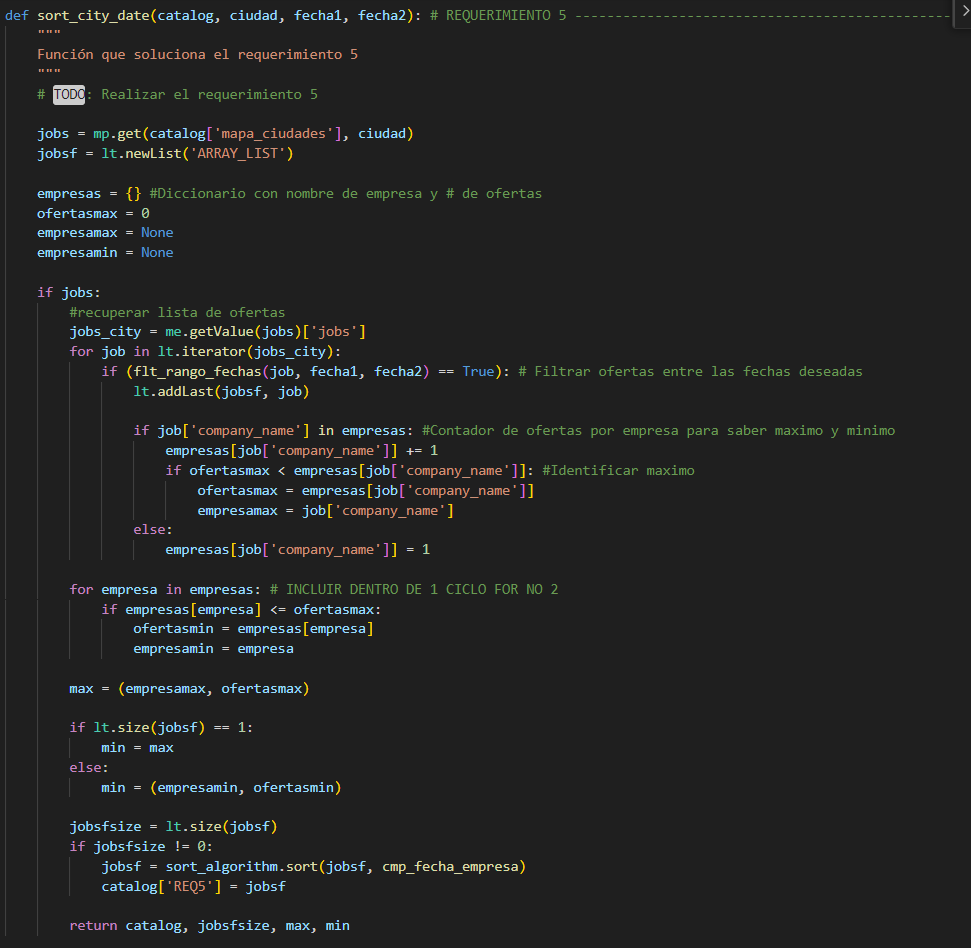
## **Análisis**

Los resultados de las pruebas sugieren que el algoritmo tiene un comportamiento lineal, lo que significa que su tiempo de ejecución aumenta linealmente con el tamaño del conjunto de datos. A medida que el tamaño del conjunto de datos aumenta, el tiempo de ejecución también aumenta significativamente, lo que indica que el algoritmo puede enfrentar problemas de rendimiento con conjuntos de datos más grandes. Este comportamiento es consistente con la complejidad lineal esperada del algoritmo.

# **Requerimiento <<5>>**

## **Descripción**

consultar las ofertas de trabajo publicadas en una ciudad dado un rango de fechas



|  |  |
| --- | --- |
| **Entrada** | Nombre de la empresa, la fecha inicial del periodo a consultar y la fecha final del periodo a consultar |
| **Salidas** | Se busca entregar un resumen del mercado laboral en una ciudad específica, incluyendo el total de ofertas, empresas participantes, la más activa y la menos activa, y un listado detallado de todas las ofertas con información relevante. |
| **Implementado (Sí/No)** | Si (Santiago Toro) |

## **Análisis de complejidad**

Análisis de complejidad de cada uno de los pasos del algoritmo

|  |  |
| --- | --- |
| **Pasos** | **Complejidad (Peor caso)** |
| Se accede al mapa de ciudades en el catálogo para obtener la lista de trabajos asociada a la ciudad. | La complejidad de esta operación es O(1) en promedio, ya que implica acceder a un elemento en un mapa. |
| Para cada trabajo en la lista de la ciudad, se verifica si está dentro del rango de fechas especificado. | O(m) |
| Se realiza un conteo de las ofertas de trabajo por empresa en la lista filtrada. | O(m) ya que se realiza dentro del ciclo que itera sobre los trabajos de la ciudad. |
| Si hay trabajos en la lista filtrada, se procede a ordenarla. | O(n log n) |
| ***TOTAL*** | ***O(m + n log n)*** |

## **Pruebas Realizadas**

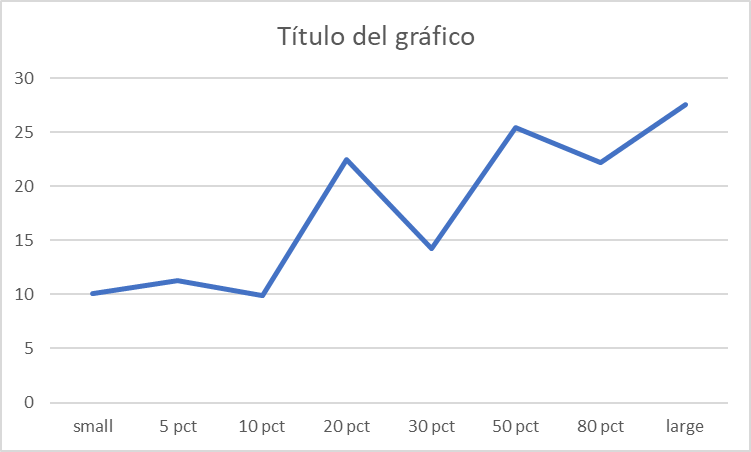
Las pruebas realizadas fueron realizadas en una maquina con las siguientes especificaciones. Los datos de entrada fueron

|  |  |
| --- | --- |
| Procesadores | AMD Ryzen 5 11400HS with Radeon Graphics |
| Memoria RAM | 8 GB |
| Sistema Operativo | Windows 11 |

|  |  |
| --- | --- |
| **Entrada** | **Tiempo (ms)** |
| small | 9.987 |
| 5 pct | 10.982 |
| 10 pct | 9.871 |
| 20 pct | 22.39 |
| 30 pct | 13.982 |
| 50 pct | 25.678 |
| 80 pct | 22.371 |
| large | 26.998 |

### **Graficas**

Las gráficas con la representación de las pruebas realizadas.



## **Análisis**

los resultados sugieren que el algoritmo puede ser eficiente para conjuntos de datos de tamaño moderado, pero puede enfrentar problemas de rendimiento con conjuntos de datos más grandes. Sería útil investigar formas de mejorar el rendimiento del algoritmo, especialmente para conjuntos de datos grandes, y comparar los resultados experimentales con las expectativas teóricas para comprender mejor su comportamiento.