ANÁLISIS DEL RETO

Nicolás Santiago Sánchez Rodríguez, 202417072, ns.sanchezr1@uniandes.edu.co

Gabriela Martínez Hurtado, 202417100, g.martinezh@uniandes.edu.co

Samantha Puentes Pinzón, 202410295, s.puentesp2@uniandes.edu.co

Requerimiento 1

Descripción

Este requerimiento retorna el ultimo registro recopilado según un año de interés. Primero, recorre la lista de registros y verifica si el año de recopilación coincide con el año de interés. Si encuentra que el año del registro es igual al ingresado, actualiza el último registro con el más reciente encontrado y cuenta cuantos registros cumplen con la condición.

Entrada	Catalogo con los registros y el año de interés.
Salidas	Tupla con el tiempo de ejecución, número total de registros y diccionario del último registro
Implementado (Sí/No)	Sí

Análisis de complejidad

Análisis de complejidad de cada uno de los pasos del algoritmo

Pasos	**Complejidad**
Asignación de variables	O(1)
Recorres registros	O(n)
Comparación de años	O(1)
Actualización de valores	O(1)
Cálculo de tiempo	O(1)
TOTAL	***O(n)***

Pruebas Realizadas

Las pruebas fueron realizadas en una maquina con las siguientes especificaciones. El dato de entrada para cada prueba fue el año ‘2022’.

Procesadores	**Intel(R) Core(TM) i5-1235U**
Memoria RAM	16 GB
Sistema Operativo	Windows 11

Entrada	Tiempo (ms)
20%	27.806
40%	38.241
60%	58.753
80%	57.806
100%	78.019

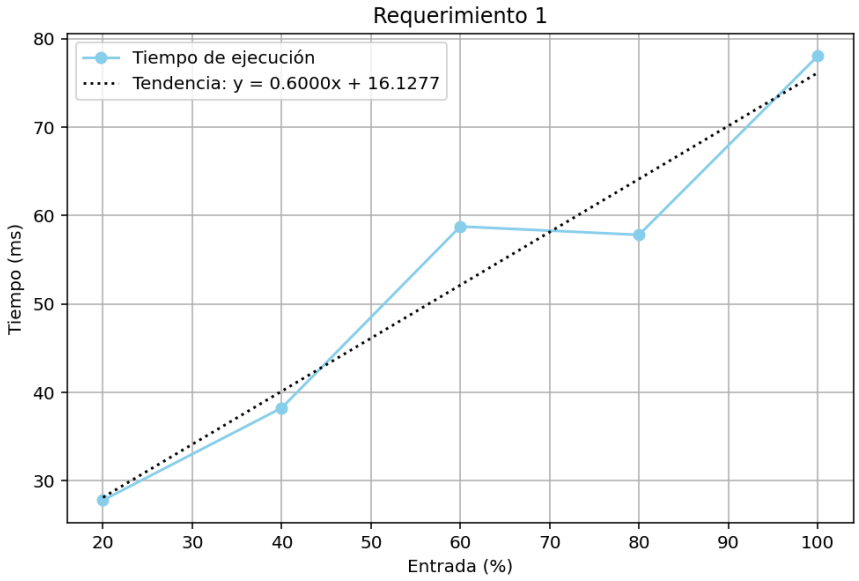
Tablas de datos

Las tablas con la recopilación de datos de las pruebas.

Muestra	Salida	Tiempo (ms)
20%	Dato1	27.806
40%	Dato2	38.241
60%	Dato3	58.753
80%	Dato4	57.806
100%	Dato5	78.019

Graficas

Las gráficas con la representación de las pruebas realizadas.



Análisis

A pesar de que acceder a un elemento en un ArrayList, dada su posición, tiene complejidad constante, la implementación de este requerimiento tiene un orden O(n). Esto se debe a que la función recorre todos

los elementos de la lista para encontrar los registros que coincidan con el año dado. En el peor de los casos, es necesario iterar sobre toda la lista, lo que implica una complejidad lineal.

Este comportamiento se puede evidenciar experimentalmente en la gráfica. Como el tiempo de ejecución depende del número de registros analizados, la curva obtenida sigue un crecimiento proporcional al tamaño del catálogo, lo que coincide con el comportamiento lineal esperado.

Requerimiento 2

Descripción

Este requerimiento filtra registros de una lista según el departamento.

Primero, se inicializan el tiempo de ejecución, una asignación para los datos filtrados y un contador para total de registros. Luego, el programa recorre los registros, verificando si cumplen con los criterios. Si es así, se va cambiando el valor del último departamento y se actualizan los contadores según su origen.

Finalmente, se retorna el tiempo de ejecución, el ultimo departamento.

Entrada	Catalog que contiene los registros y el departamento de interés.
Salidas	Tupla con el tiempo de ejecución, total de registros que cumplieron el filtro por el departamento de interés.
Implementado (Sí/No)	Si.

Análisis de complejidad

Análisis de complejidad de cada uno de los pasos del algoritmo

Pasos	**Complejidad**
Inicializar el tiempo	O (1)
Inicializar ultimo registro depo	O(1)
Inicializar fecha max	O(1)
Inicializar total registros	O(1)
Bucle para acceder a cada registro	O(n)
Condicional del departamento	O(1)
Fecha actual	O(1)
Condicional de la fecha max o fecha actual	O(1)
Fecha max	O(1)
Ultimo registro con el valor	O(1)
Total de registros mas 1	O(1)
Finalizar el tiempo	O(1)
Obtener el tiempo final y calcular la diferencia	O(1)

Retornar los valores	O(1)
TOTAL	***O(n)***

Pruebas Realizadas

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

Procesadores	AMD Ryzen 7 4800HS with Radeon Graphics
Memoria RAM	16 GB
Sistema Operativo	Windows 11

Entrada	**Tiempo (ms)**
20%	68.447
40%	113.790
60%	149.246
80%	189.557
100%	239.439

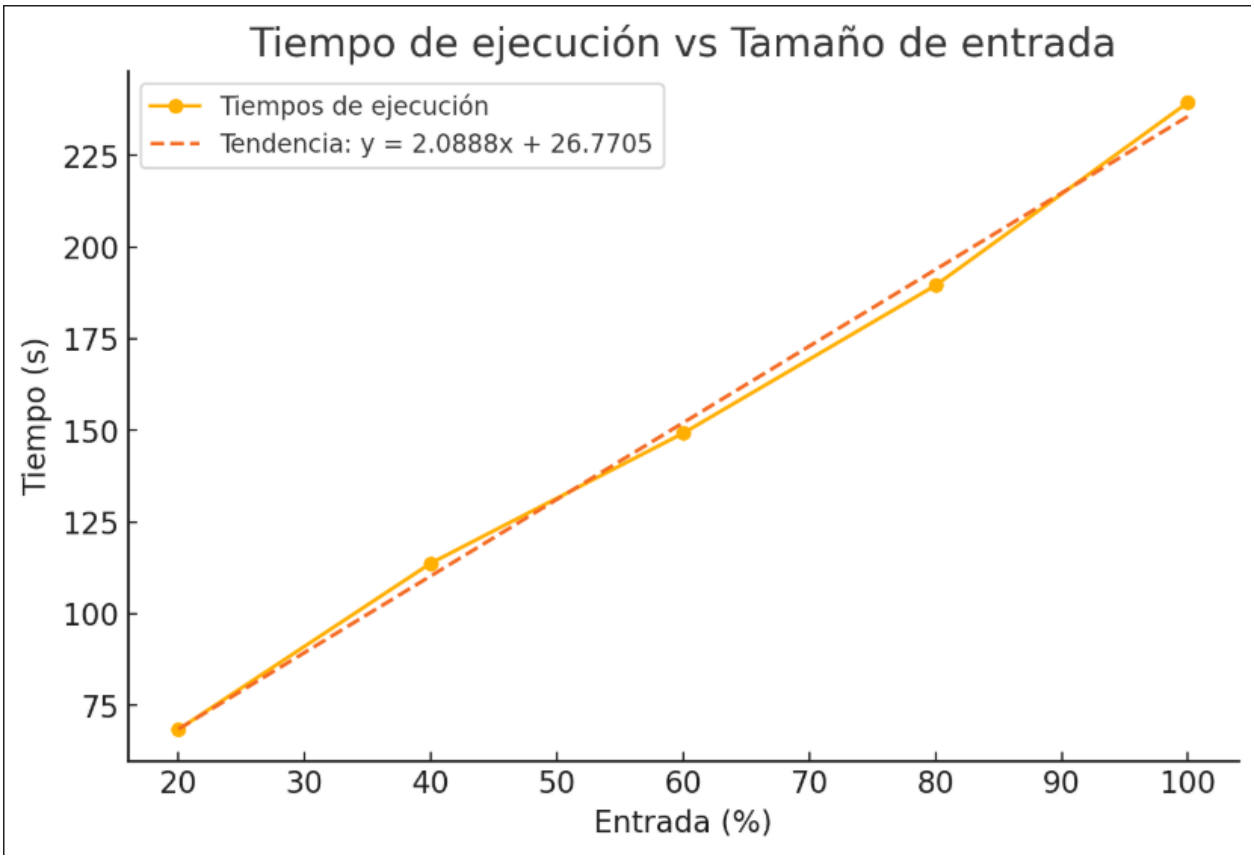
Tablas de datos

Las tablas con la recopilación de datos de las pruebas.

Muestra	**Salida**	**Tiempo (ms)**
20%	Dato1	68.447
40%	Dato2	113.790
60%	Dato3	149.246
80%	Dato4	189.557
100%	Dato5	239.439

Graficas

Las gráficas con la representación de las pruebas realizadas.



Análisis

La implementación tiene una complejidad lineal O(n), ya que recorre toda la lista para filtrar los registros del departamento buscado. En el peor caso, revisa todos los elementos, lo que hace que el tiempo de ejecución crezca proporcionalmente con el tamaño de los datos.

La gráfica confirma este comportamiento, mostrando una tendencia lineal en los tiempos de ejecución. Para optimizar, se podría usar un **diccionario indexado por departamento**, reduciendo la búsqueda a **O(1)**.

Requerimiento 3

Descripción

Este requerimiento filtra registros de una lista según el departamento y un rango de años.

Primero, se inicializan el tiempo de ejecución, una lista para los datos filtrados y contadores para survey y *census*. Luego, el programa recorre los registros, verificando si cumplen con los criterios. Si es así, se agregan a la lista y se actualizan los contadores según su origen.

Finalmente, se retorna el tiempo de ejecución, el total de registros filtrados y las cantidades de survey y census.

Entrada	Catalog que contiene los registros, departamento de interés, año inicial de interés y año final de interés
Salidas	Tupla con el tiempo de ejecución, número total de registros que cumplieron con el filtro, total de registros con 'source' SURVEY, total de registros con 'source' CENSUS y listado de registros filtrados resultantes.
Implementado (Sí/No)	Si, implementado por Samantha Puentes

Análisis de complejidad

Análisis de complejidad de cada uno de los pasos del algoritmo

Pasos	**Complejidad**
Inicializar el tiempo	O(1)
Asignar el valor survey	O(1)
Asignar el valor census	O(1)
Bucle para acceder a cada registro	O(n)
Condicional del departamento y año	O(1)
Agregar el valor a la lista	O(1)
Condicional de source en survey	O(1)
Contador survey	O(1)
Condicional de source en census	O(1)
Contador census	O(1)
Obtener el tamaño de la lista	O(1)
Finalizar el tiempo	O(1)
Obtener el tiempo final y calcular la diferencia	O(1)
Retornar los valores	O(1)
TOTAL	***O(n)***

Pruebas Realizadas

Descripción de las pruebas de tiempos de ejecución y memoria utilizada. Incluir descripción del procedimiento, las condiciones, las herramientas y recursos utilizados (librerías, computadores donde se ejecutan las pruebas, entre otros).

Procesadores	**AMD Ryzen 7 4800HS with Radeon Graphics**
Memoria RAM	16 GB
Sistema Operativo	Windows 11

Entrada	Tiempo (ms)
Agricultural-20	24.957
Agricultural-40	52.378
Agricultural-60	70.718
Agricultural-80	105.664
Agricultural-100	119.317

Tablas de datos

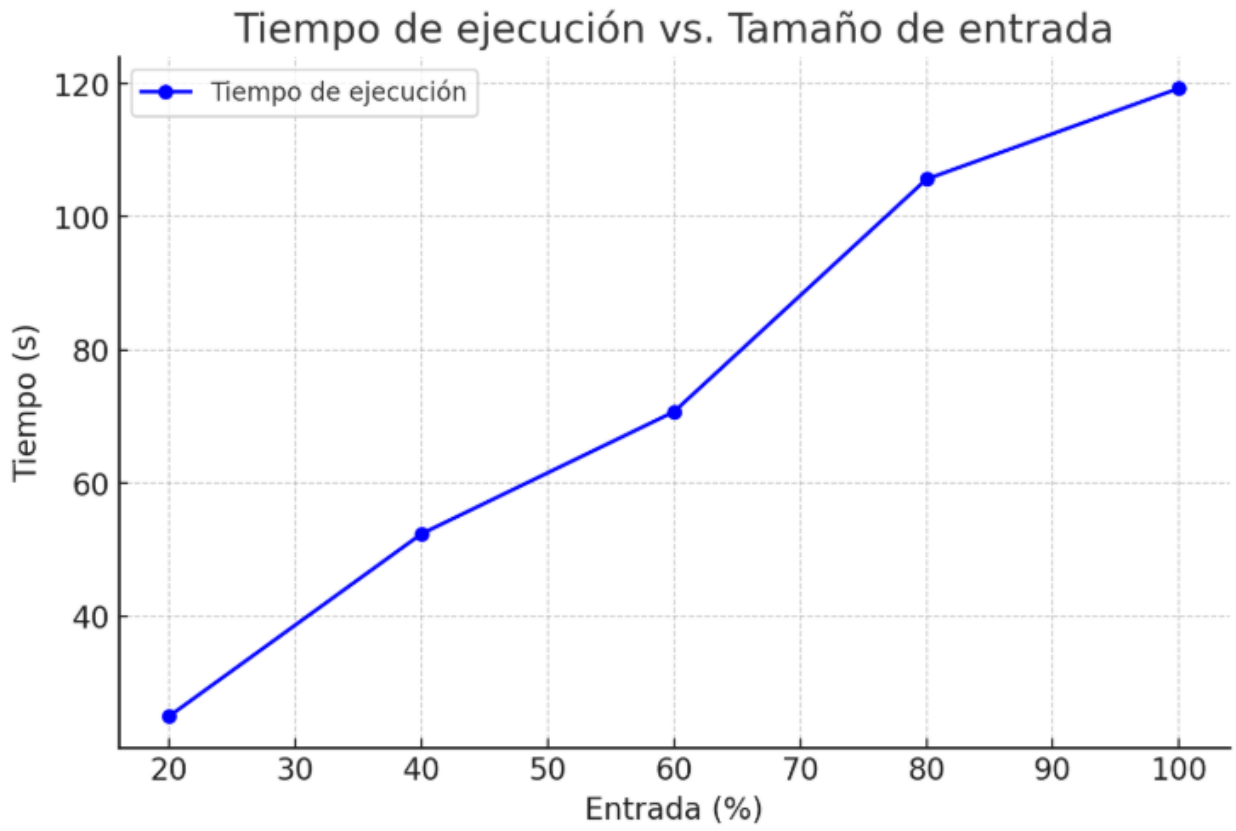
Las tablas con la recopilación de datos de las pruebas.

Los datos con los que se probó fueron:
Departamento: Iowa
Año inicial: 2000
Año final: 2010

Muestra	Salida	Tiempo (ms)
20%	Dato1	24.957
40%	Dato2	52.378
60%	Dato3	70.718
80%	Dato4	105.664
100%	Dato5	119.317

Graficas

Las gráficas con la representación de las pruebas realizadas.



Análisis

A pesar de que la inserción de un elemento en una lista enlazada tiene una complejidad constante O(1), la implementación de este requerimiento presenta un orden de complejidad lineal O(n). Esto se debe a que el algoritmo recorre toda la lista de registros para verificar cuáles cumplen con las condiciones de filtrado. En el peor de los casos, cuando ningún elemento es descartado de inmediato, es necesario iterar sobre la totalidad de los datos, lo que implica una complejidad lineal.

Este comportamiento se confirma experimentalmente en la gráfica. Dado que los tiempos de ejecución crecen proporcionalmente con el tamaño de la entrada y los datos obtenidos siguen de cerca la línea de tendencia, se evidencia que el algoritmo se ajusta al comportamiento lineal esperado.

Requerimiento 4

Descripción

Este requerimiento filtra registros de una lista según un tipo de producto es decir el commodity y un rango de años.

Primero, se inicializan el tiempo de ejecución, una lista para los datos filtrados y contadores para **SURVEY** y **CENSUS**. Luego, el programa recorre los registros, verificando si el **commodity** buscado está

en las categorías del registro y si el año de recopilación está dentro del rango dado. Si cumple con los criterios, el registro se agrega a la lista y se actualizan los contadores según su origen.

Finalmente, se retorna el tiempo de ejecución, el total de registros filtrados y las cantidades de **SURVEY** y **CENSUS**.

Entrada	Catalog que contiene los registros, commodity de interés, año inicial de interés y año final de interés
Salidas	Tupla con el tiempo de ejecución, número total de registros que cumplieron con el filtro, total de registros con 'source' SURVEY, total de registros con 'source' CENSUS y listado de registros filtrados resultantes.
Implementado (Sí/No)	Si, implementado por Nicolas Sánchez

Análisis de complejidad

Análisis de complejidad de cada uno de los pasos del algoritmo

Pasos	**Complejidad**
Inicializar tiempo	O(1)
Crear una lista vacía	O(1)
Contador de registros survey	O(1)
Contador de registros census	O(1)
Bucle para acceder a cada registro	O(n)
Se asigna un valor a categories	O(1)
Condicional del departamento y año	O(1)
Agrega el registro filtrado a la lista.	O(1)
Condicional de source en survey	O(1)
Contador survey	O(1)
Condicional de source en census	O(1)
Contador census	O(1)
Obtener el tamaño de la lista	O(1)
Finalizar el tiempo	O(1)
Obtener el tiempo final y calcular la diferencia	O(1)
Retornar los valores	O(1).
TOTAL	***O(n)***

Pruebas Realizadas

Descripción de las pruebas de tiempos de ejecución y memoria utilizada. Incluir descripción del procedimiento, las condiciones, las herramientas y recursos utilizados (librerías, computadores donde se ejecutan las pruebas, entre otros).

Procesadores	AMD Ryzen 7 4800HS with Radeon Graphics
Memoria RAM	16 GB
Sistema Operativo	Windows 11

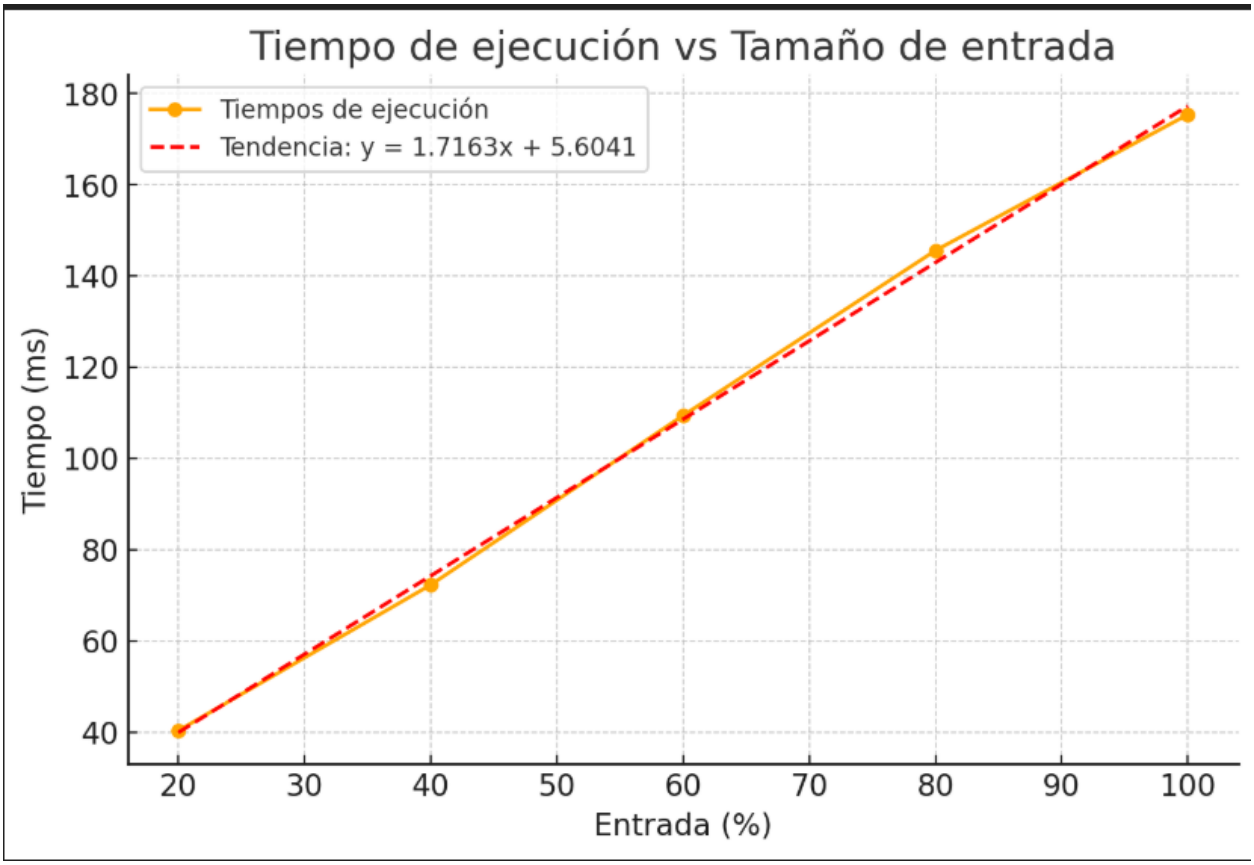
Entrada	**Tiempo (ms)**
20%	40.342
40%	72.285
60%	109.387
80%	145.546
100 %	175.337

Tablas de datos

Las tablas con la recopilación de datos de las pruebas.

Muestra	**Salida**	**Tiempo (ms)**
20%	Dato1	40.342
40%	Dato2	72.285
60%	Dato3	109.387
80%	Dato4	145.546
100%	Dato5	175.337

Graficas



Análisis

La implementación presenta una complejidad **O(n)**, ya que recorre toda la lista de datos para procesar la entrada. En el peor de los casos, analiza todos los elementos, lo que hace que el tiempo de ejecución aumente de manera **proporcional** al tamaño de la entrada.

La tendencia observada en la tabla y la gráfica confirma este comportamiento **lineal**, donde el tiempo de ejecución crece a medida que la cantidad de datos procesados se incrementa.

Requerimiento 5

Descripción

Este requerimiento filtra los registros según una categoría estadística y un rango de años. Para ello, recorre la lista de registros y verifica si la categoría del registro coincide con la ingresada y si el año de carga está dentro del rango especificado. Los registros que cumplen estas condiciones se almacenan en una nueva lista y se contabilizan según su fuente (SURVEY o CENSUS). Finalmente, retorna la cantidad de registros filtrados, su distribución por fuente y el tiempo de ejecución del proceso.

Entrada	Catalogo con los registros, categoría estadística a filtrar, año inicial del periodo a consultar y año final del periodo a consultar
Salidas	Tupla con tiempo de ejecución, numero total de registros filtrados, número total de registros con fuente SURVEY, numero total de registros con fuente CENSUS y array con los registros filtrados.
Implementado (Sí/No)	Sí. Implementado por Gabriela Martinez

Análisis de complejidad

Análisis de complejidad de cada uno de los pasos del algoritmo

Pasos	**Complejidad**
Asignación de variables	O(1)
Recorrido registros	O(n)
Comparaciones y filtrado	O(1)
Adición a la lista	O(1)
Cálculo de tiempo	O(1)
TOTAL	***O(n)***

Pruebas Realizadas

Las pruebas fueron realizadas en una maquina con las siguientes especificaciones. Los datos de entrada fueron INVENTORY en categoría estadística y el periodo fue de 2007 a 2020.

Procesadores	**Intel(R) Core(TM) i5-1235U**
Memoria RAM	16 GB
Sistema Operativo	Windows 11

Entrada	**Tiempo (ms)**
20%	57.16
40%	92.39
60%	125.221
80%	173.426
100%	213.33

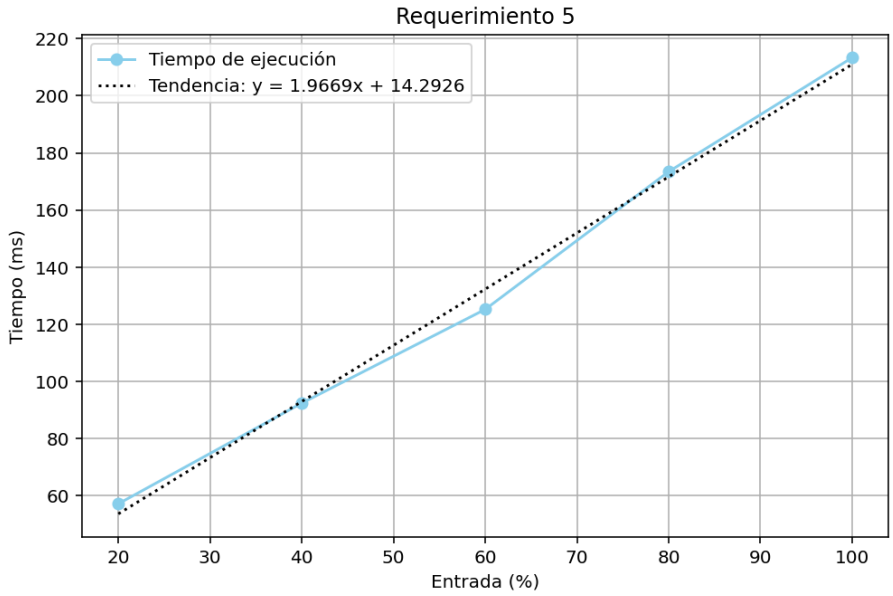
Tablas de datos

Las tablas con la recopilación de datos de las pruebas.

Muestra	**Salida**	**Tiempo (ms)**
20%	Dato1	57.16
40%	Dato2	92.39
60%	Dato3	125.221
80%	Dato4	173.426
100%	Dato5	213.33

Graficas

Las gráficas con la representación de las pruebas realizadas.



Análisis

La implementación tiene una complejidad O(n), ya que recorre todos los registros para filtrar por categoría y rango de años. Además, realiza conversiones de cadenas y comparaciones, pero su impacto es mínimo frente al recorrido lineal.

Experimentalmente, el tiempo de ejecución crece de forma proporcional al número de registros, siguiendo un comportamiento lineal esperado.

Requerimiento 6

Descripción

Este requerimiento retorna un listado de los registros que cumplieron con los filtros, estos siendo el departamento de interés, y un periodo de tiempo definido. Cuando los filtros se cumplen se recuento el total de los registros, cuantos fueron tipo censo y cuantos fueron tipo encuesta. Para el listado resultante se muestra el tipo de fuente, el año de recopilación, el nombre del departamento, la unidad de medición y el tipo de producto.

Entrada	Catálogo, departamento para filtrar, año inicial y final del periodo de interés
Salidas	Tiempo de ejecución, número total de registros, número total de tipo de fuente censo o encuesta, y para el listado resultante, el tipo de fuente, año de recopilación, nombre del departamento, unidad de medición y tipo de producto

Implementado (Sí/No)	Si.

Análisis de complejidad

Análisis de complejidad de cada uno de los pasos del algoritmo

Pasos	**Complejidad**
Asignación de variables	O(1)
Recorrido completo	O(n)
Conversión de fecha	O(1)
Comparación de fechas y departamento	O(1)
Comparación tipo de fuente	O(1)
Cálculo de tiempo	O(1)
TOTAL	***O(n)***

Pruebas Realizadas

Descripción de las pruebas de tiempos de ejecución y memoria utilizada. Incluir descripción del procedimiento, las condiciones, las herramientas y recursos utilizados (librerías, computadores donde se ejecutan las pruebas, entre otros).

Procesadores	**AMD Ryzen 7 5700 with Radeon Graphics**
Memoria RAM	16 GB
Sistema Operativo	Windows 11

Entrada	**Tiempo (s)**
20%	0.631
40%	1.289
60%	1.925
80%	2.636
100%	3.141

Tablas de datos

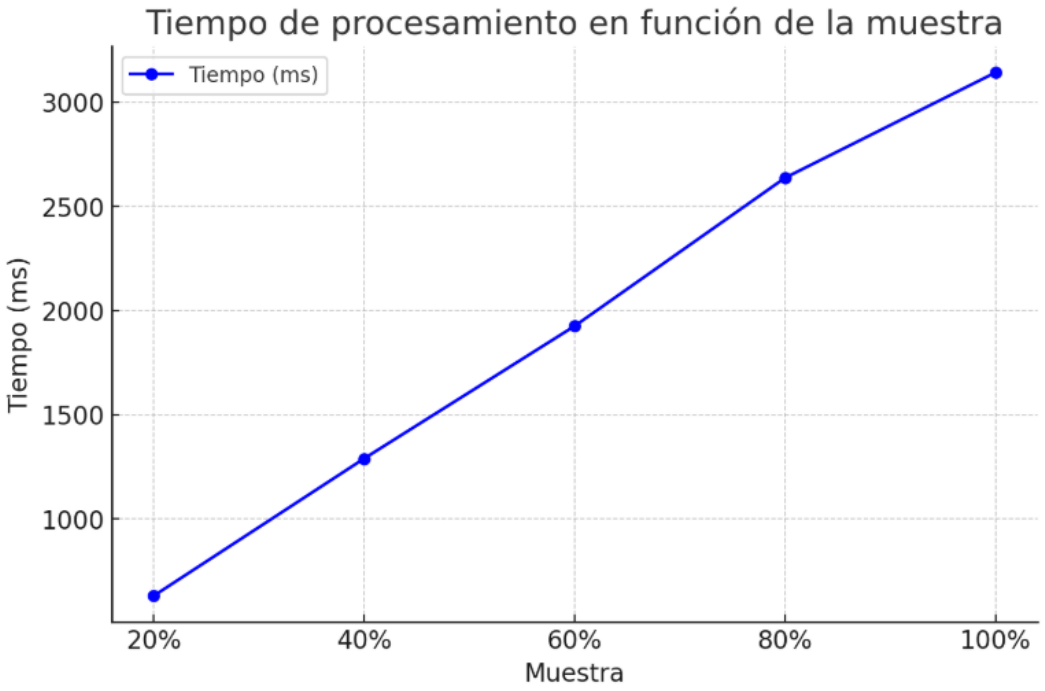
Las tablas con la recopilación de datos de las pruebas.

Todas las pruebas se realizaron con estos parámetros: Iowa, 2013-01-01, 2024-12-31

Muestra	**Salida**	**Tiempo (ms)**
20%	Dato1	631.051
40%	Dato2	1289.860
60%	Dato3	1925.452
80%	Dato4	2636.646
100%	Dato5	3141.967

Graficas

Las gráficas con la representación de las pruebas realizadas.



Análisis

La implementación es correcta y las pruebas indican que a mayor tamaño de muestras mayor tiempo se demora, esto probando que la complejidad temporal tiende a O(n)

Requerimiento 7

Descripción

El requerimiento nos pide recorrer todos los registros del departamento de interés en un periodo de tiempo definido para encontrar los registros de menor y mayor valor. También contar el total de registros que cumplieron el filtro, el total de registros cuyo valor no es válido, y el total de tipo censo y de tipo encuesta.

Entrada	Catálogo, nombre del departamento, año inicial y final del periodo
Salidas	Tiempo de ejecución, número total de registros, y para los de mayores y menores ingresos, año de recopilación, indicación de si es el mayor o menor, valor de ingresos, total de registros, registros cuyo valor no es válido, y total de registros de tipo censo y tipo entrevista.
Implementado (Sí/No)	Si.

Análisis de complejidad

Análisis de complejidad de cada uno de los pasos del algoritmo

Pasos	**Complejidad**
Asignación de variables	O(1)
Recorrido completo	O(n)
Comparaciones del filtro	O(1)
Reemplazo de valores de variables	O(1)
Incremento de contadores	O(1)
Comparaciones menor y mayor	O(1)
Reemplazo mayor y menor	O(1)
Cálculo de tiempo de ejecución	O(1)
TOTAL	***O(n)***

Pruebas Realizadas

Descripción de las pruebas de tiempos de ejecución y memoria utilizada. Incluir descripción del procedimiento, las condiciones, las herramientas y recursos utilizados (librerías, computadores donde se ejecutan las pruebas, entre otros).

Procesadores	**AMD Ryzen 7 5700 with Radeon Graphics**
Memoria RAM	16 GB
Sistema Operativo	Windows 11

Entrada	**Tiempo (s)**
20%	0.027
40%	0.048
60%	0.069
80%	0.100
100%	0.114

Tablas de datos

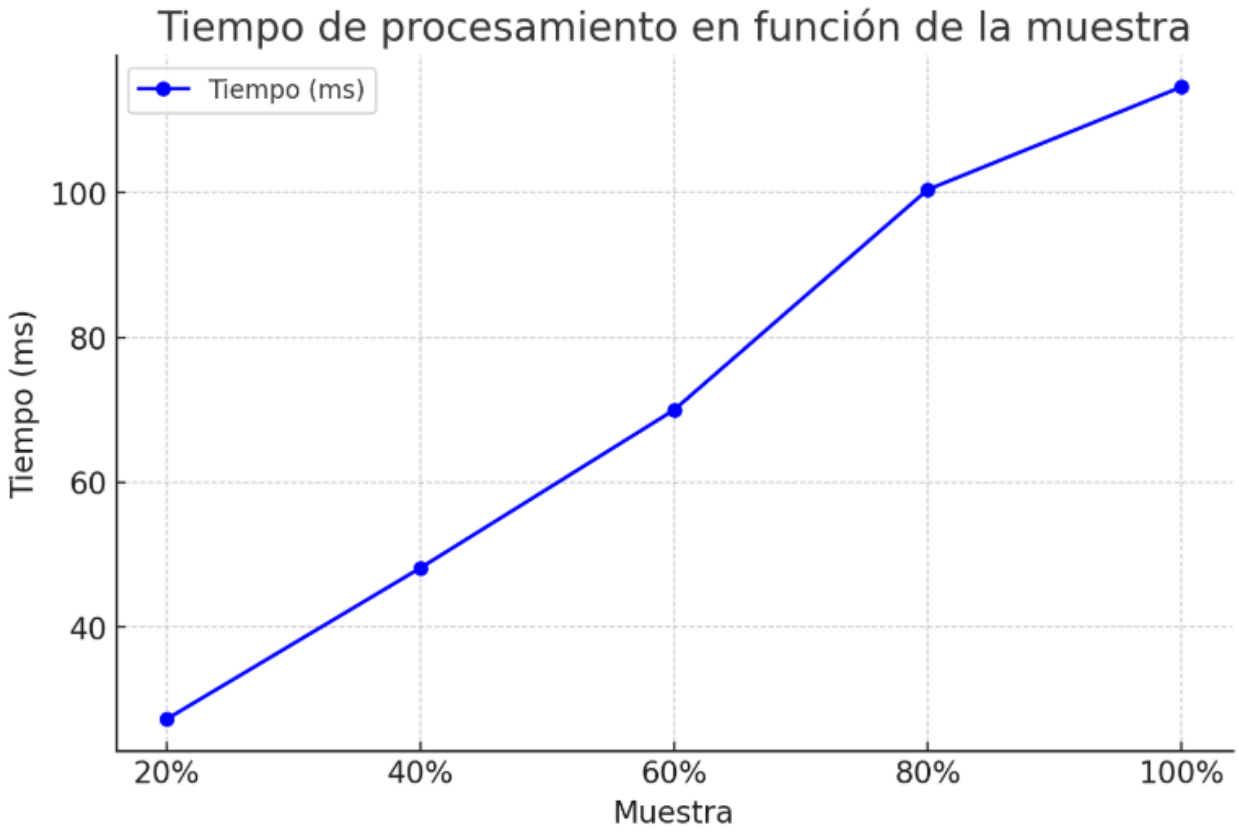
Las tablas con la recopilación de datos de las pruebas.

Se realizo todo con Iowa, 2000, 2024

Muestra	**Salida**	**Tiempo (ms)**
20%	Dato1	27.267
40%	Dato2	48.148
60%	Dato3	69.979
80%	Dato4	100.426
100%	Dato5	114.612

Graficas

Las gráficas con la representación de las pruebas realizadas.



Análisis

La implementación es correcta y las pruebas indican que a mayor tamaño de muestras mayor tiempo se demora, esto probando que la complejidad temporal tiende a O(n).

Requerimiento 8

Descripción

El requerimiento nos pide recorrer todos los registros para encontrar el total de departamentos, el tiempo promedio de carga, año de inicio de recopilación menor y mayor, y el departamento que mayor promedio tenga, nombre del departamento, tiempo promedio, total de registros, año de inicio de recopilación menor y mayor, tiempo entre recopilación y carga menor y mayor, y total de registros de tipo censo y encuesta.

Entrada	Catálogo
Salidas	Tiempo de ejecución, total de departamentos, tiempo promedio de carga, año de inicio de recopilación menor y mayor, y para el departamento con mayor tiempo de carga, nombre del departamento, tiempo promedio, total de registros, año de inicio de recopilación menor y mayor, tiempo entre recopilación y carga menor y mayor, y total de registros de tipo censo y encuesta.
Implementado (Sí/No)	Si.

Análisis de complejidad

Análisis de complejidad de cada uno de los pasos del algoritmo

Pasos	**Complejidad**
Asignación de variables	O(1)
Recorrido total	O(n)
Comparaciones de año, tipo y departamento o creación de un nuevo diccionario del departamento	O(1)
Incrementos en variables	O(1)
Recorrido del diccionario de los departamentos	O(1)
Comparación de promedios	O(1)
Reemplazo y cálculo de promedio final	O(1)
Cálculo de tiempo	O(1)
TOTAL	***O(n)***

Pruebas Realizadas

Descripción de las pruebas de tiempos de ejecución y memoria utilizada. Incluir descripción del procedimiento, las condiciones, las herramientas y recursos utilizados (librerías, computadores donde se ejecutan las pruebas, entre otros).

Procesadores	**AMD Ryzen 7 5700 with Radeon Graphics**
Memoria RAM	16 GB
Sistema Operativo	Windows 11

Entrada	**Tiempo (s)**
20%	0.187
40%	0.372
60%	0.552
80%	0.714
100%	0.882

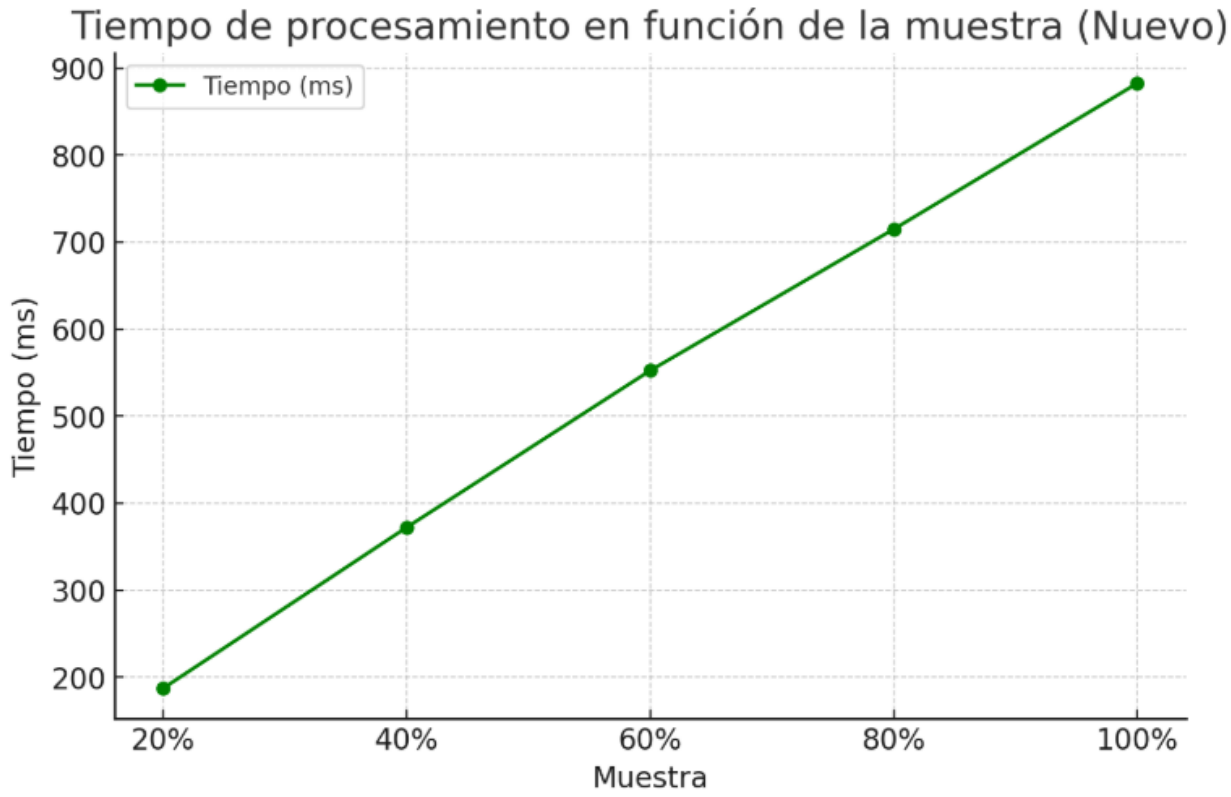
Tablas de datos

Las tablas con la recopilación de datos de las pruebas.

Muestra	Salida	Tiempo (ms)
20%	Dato1	187.093
40%	Dato2	372.097
60%	Dato3	552.371
80%	Dato4	714.853
100%	Dato5	882.532

Graficas

Las gráficas con la representación de las pruebas realizadas.



Análisis

La implementación es correcta y las pruebas indican que a mayor tamaño de muestras mayor tiempo se demora, esto probando que la complejidad temporal tiende a O(n)

Requerimiento Ejemplo

Descripción

```python
def get_data(data_structs, id):
    """
    Retorna un dato a partir de su ID
    """
    pos_data = lt.isPresent(data_structs["data"], id)
    if pos_data > 0:
        data = lt.getElement(data_structs["data"], pos_data)
        return data
    return None
```

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

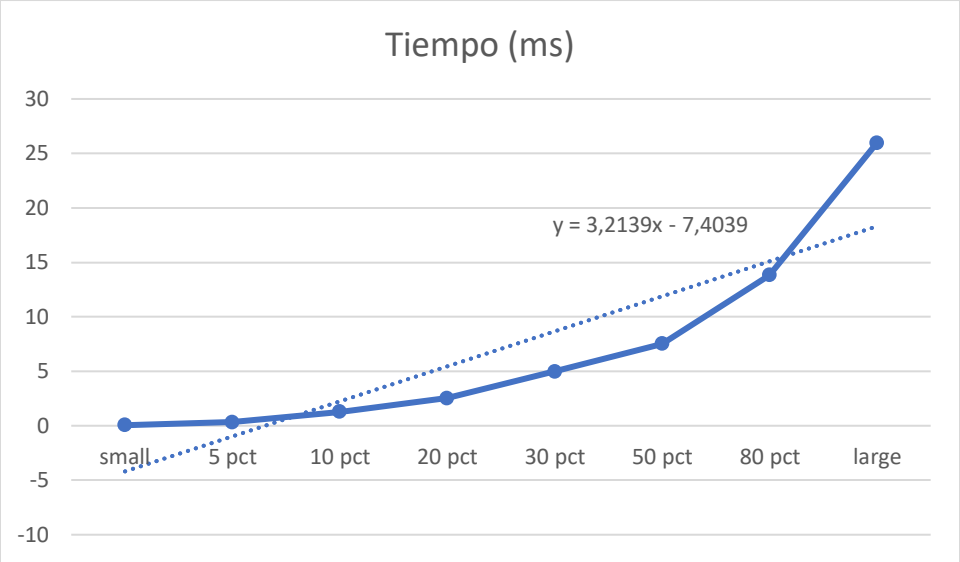
Tablas de datos

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.