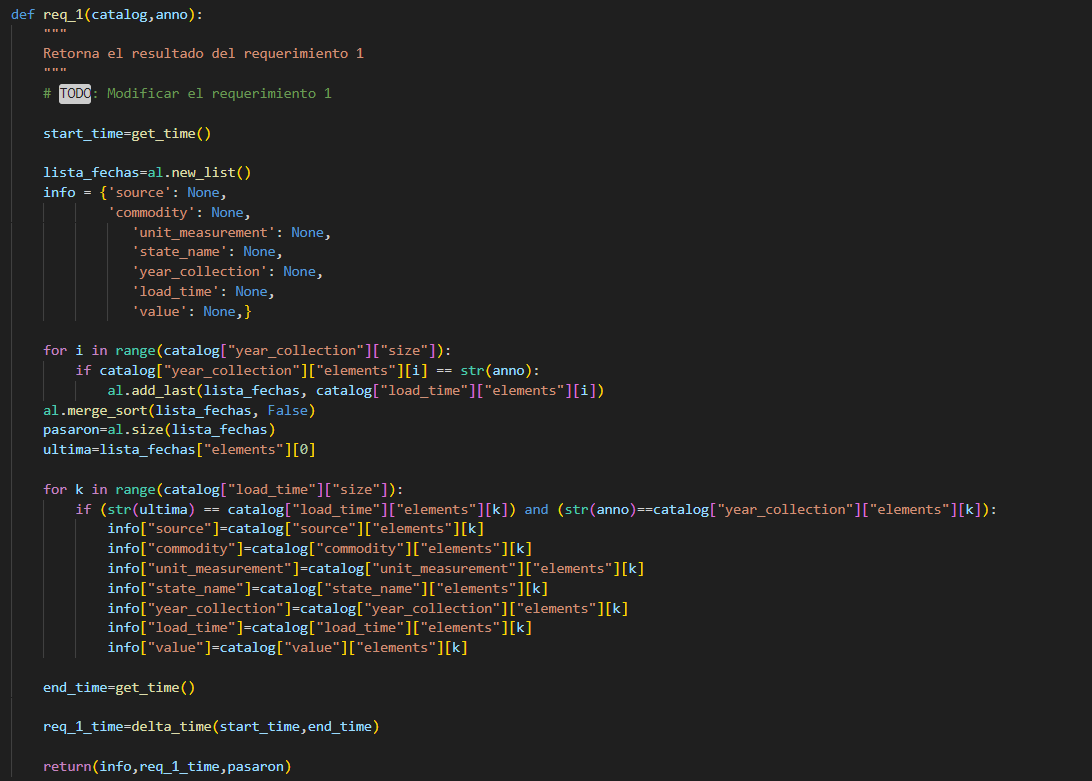
**ANÁLISIS DEL RETO**

Tomas Aponte, 202420148, t.aponte@uniandes.edu.co

# Juan Diego García, 202423575jd.garcia12@uniandes.edu.co

# Requerimiento <<1>>



## Descripción

Este requerimiento se encarga de buscar y mostrar al usuario el último registro encontrado durante un año específico. Además, retorna sus características base. Lo que hace es crear un listado de fechas en donde se insertará la información de los registros que cumplan el filtro de búsqueda. Para encontrar dichos registros se ejecuta un ciclo el cual compara la información necesaria con la de todos los registros para así añadir los que cumplen el filtro. Una vez terminado el ciclo se ordena la lista de manera ascendente, de tal modo que el primer año en la lista sea el último que se encontró dentro del primer filtro. Luego se asigna la longitud de la lista ordenada para el retorno de la totalidad de registros que pasaron el filtro de búsqueda, y por último se hace un nuevo ciclo para encontrar la información del registro que se debe retornar.

|  |  |
| --- | --- |
| **Entrada** | Catálogo con los datos del archivo csv, año deseado |
| **Salidas** | Información del último registro encontrado, totalidad de registros que cumplieron el filtro de búsqueda, tiempo de ejecución |
| **Implementado (Sí/No)** | Si, implementado por Juan García |

## Análisis de complejidad

|  |  |
| --- | --- |
| **Pasos** | **Complejidad** |
| Asignaciones variables | O(1) |
| Recorrido total del catálogo (for i in range …) por filtro para insertar datos en **lista\_fechas** | O(n) |
| Ordenamiento **lista\_fechas** | O(n log n) |
| Nuevas variables asignadas | O(1) |
| Búsqueda del último registro para su retorno | O(n) |
| ***TOTAL*** | ***O(n log n)*** |

## Pruebas Realizadas

Año de interés: 2010

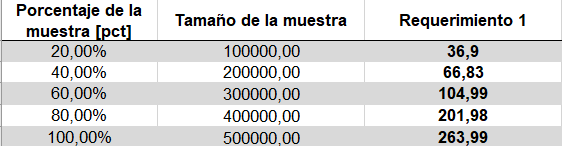
Procesador: Ryzen 7 7730U

RAM: 16GB

Sistema Operativo: Windows 11

|  |  |
| --- | --- |
| **Entrada** | **Tiempo (s)** |
| 20% | **36,9** |
| 40% | **66,83** |
| 60% | **104,99** |
| 80% | **201,98** |
| 100% | **263,99** |

### Tablas de datos

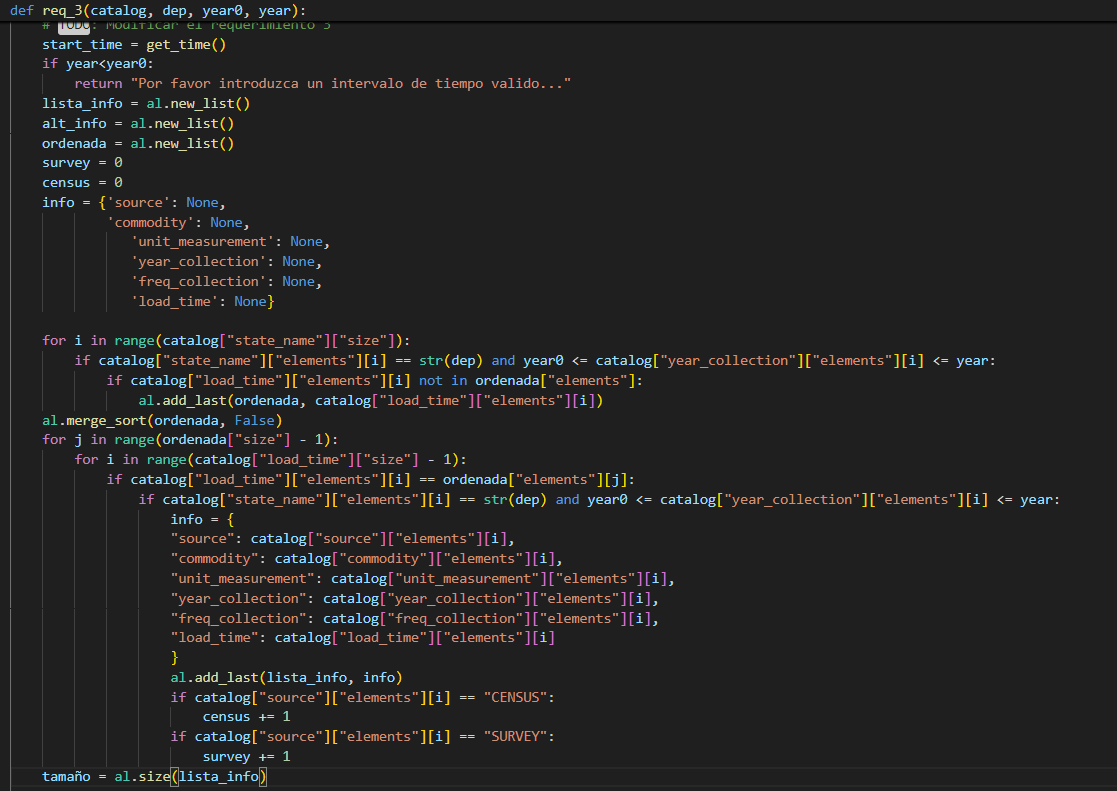


### Gráficas

## Análisis

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

# **Requerimiento <<3>>**



## Descripción

Este requerimiento se encarga de recopilar todos los registros con su respectiva información en una lista ordenada, dentro de un margen de tiempo para un departamento especificado por el usuario, indicando también cuántos de los orígenes de los registros son de tipo “*survey*” o *“census*”. Si esta lista supera los 20 elementos, se retorna solo los primeros y últimos 5. Además, si el margen de tiempo no es coherente, se le indica al usuario que intente la búsqueda con un margen de tiempo adecuado.

|  |  |
| --- | --- |
| **Entrada** | Catálogo con los datos del archivo csv, el departamento deseado y el intervalo de tiempo deseado |
| **Salidas** | Lista con los datos de los registros encontrados, total de datos de tipo “*survey*”, total de datos de tipo *“census*”, total de datos que pasaron el filtro de búsqueda y el tiempo de ejecución del requisito en milisegundos |
| **Implementado (Sí/No)** | Si, implementado por Juan García |

## **Análisis de complejidad**

|  |  |
| --- | --- |
| **Pasos** | **Complejidad** |
| Validación del intervalo (if year<year0) | O(1) |
| Asignación de variables | O(1) |
| Recorrido total del catálogo (for i in range …) | O(n) |
| Filtros de la primera búsqueda | O(n) |
| Ordenamiento **ordenada** | O(n log n) |
| Búsqueda coincidencias **ordenada** y catálogo | O(n2) |
| Asignación info | O(1) |
| Añadir a **lista\_info** | O(n) |
| Verificación del tamaño | O(1) |
| Posible agregado a **alt\_list** | O(n) |
| ***TOTAL*** | ***O(n2)*** |

## **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).

Año de inicio: 2010

Año de fin: 2018

Departamento: SOUTH CAROLINA

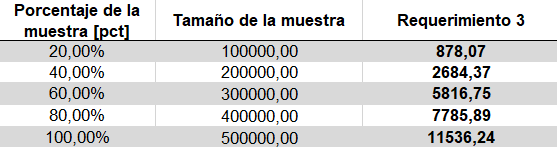
Procesador: Ryzen 7 7730U

RAM: 16GB

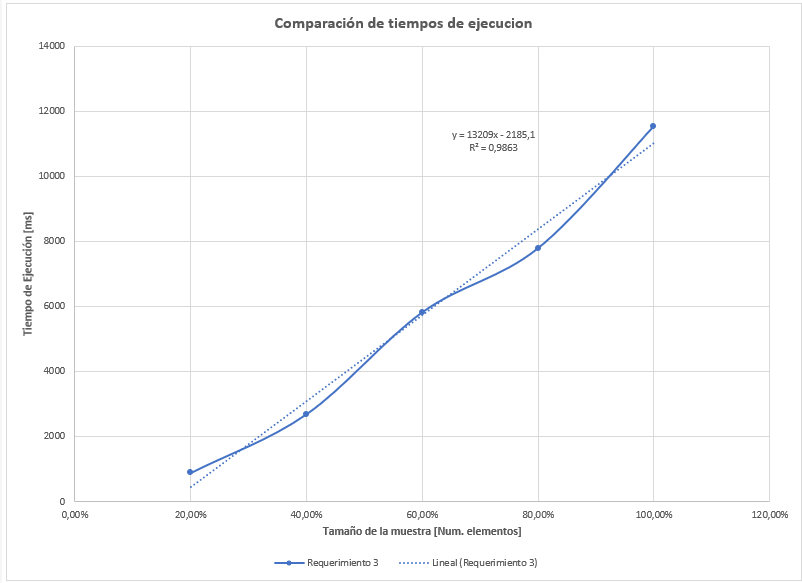
Sistema Operativo: Windows 11

|  |  |
| --- | --- |
| **Entrada** | **Tiempo (s)** |
| 20% | **878,07** |
| 40% | **2684,37** |
| 60% | **5816,75** |
| 80% | **7785,89** |
| 100% | **11536,24** |

### **Tablas de datos**

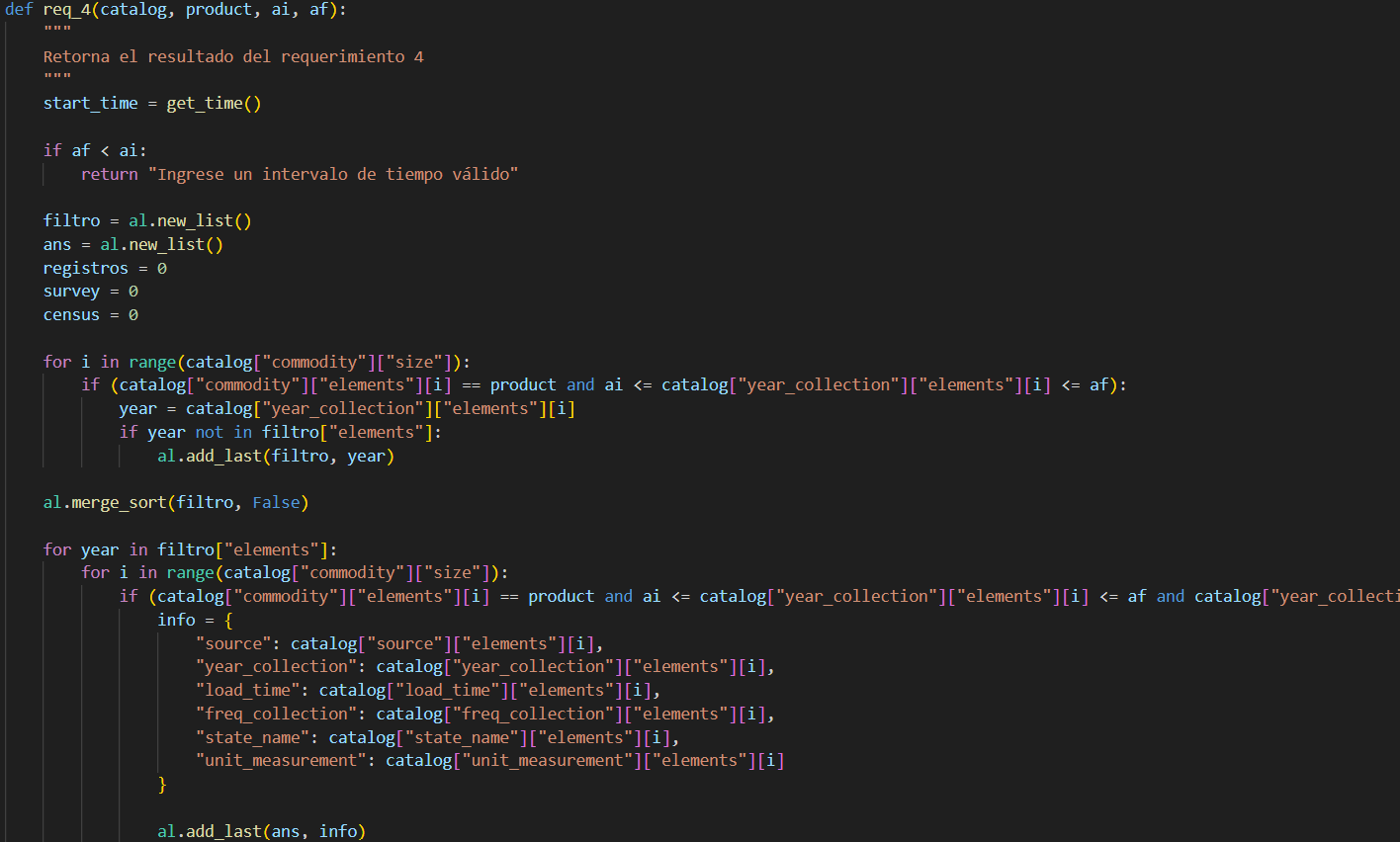


### **Graficas**



## **Análisis**

# **Requerimiento <<4>>**



## **Descripción**

Este requerimiento se encarga de buscar los registros dentro del catálogo que tienen un producto y tiempo de recolección específico dado por el usuario. Primero descarta intervalos de tiempo inválidos, luego crea las listas y contadores que serán empleados, recorre la lista por primera vez para guardar y organizar los años validos en una lista, se recorre la lista una segunda vez para cada elemento de la lista ordenada y se encuentran y guardan los registros que coinciden ya en orden, también incrementan los contadores en el proceso, se comprueba si la lista deba ser acortada y retorna el resultado.

|  |  |
| --- | --- |
| **Entrada** | Catálogo, producto, año inicial, año final |
| **Salidas** | Listado de registros, cantidad de tipo census, cantidad de tipo survey, cantidad de registros, tiempo de ejecución. |
| **Implementado (Sí/No)** | Si, Tomas Aponte |

## **Análisis de complejidad**

|  |  |
| --- | --- |
| **Pasos** | **Complejidad** |
| Se crean contadores y listas requeridas | O(1) |
| Se recorre la lista para años validos | O(n) |
| Se añaden los años validos a una lista | O(1) |
| Se organiza la lista | O(nlog(n)) |
| Se recorre el catálogo para cada año valido | O(n\*\*2) |
| Se añaden lo registros validos a una lista | O(1) |
| Incrementan los contadores necesarios | O(1) |
| Se acorta la lista si es necesario | O(1) |
| Se retornan los resultados | O(1) |
| **TOTAL** | O(n\*\*2) |

## **Pruebas Realizadas**

Año inicio: 2010

Año final: 2020

Producto: CATTLE

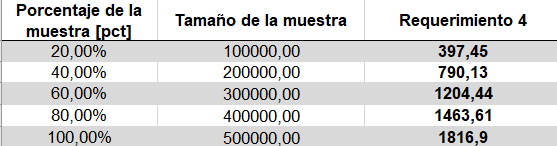
Procesador: Ryzen 7 7730U

RAM: 16GB

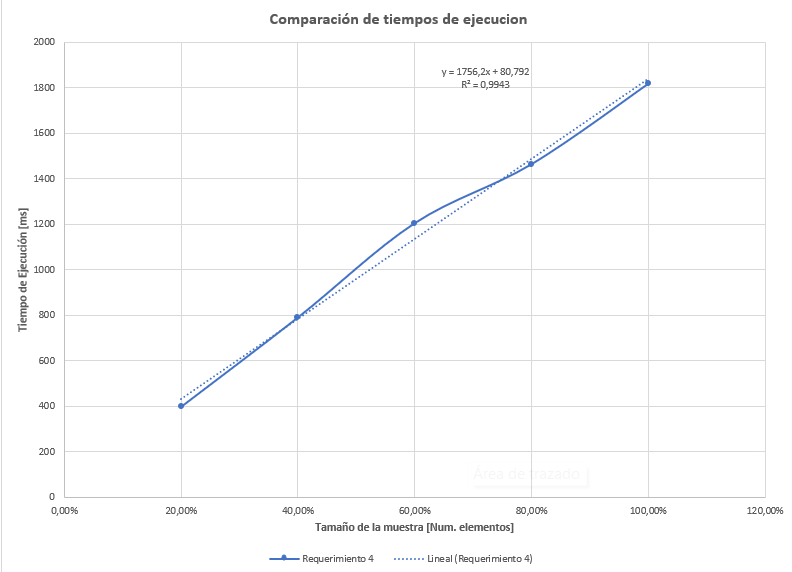
Sistema Operativo: Windows 11

|  |  |
| --- | --- |
| **Entrada** | **Tiempo (s)** |
| 20% | **397,45** |
| 40% | **790,13** |
| 60% | **1204,44** |
| 80% | **1463,61** |
| 100% | **1816,9** |

### **Tablas de datos**



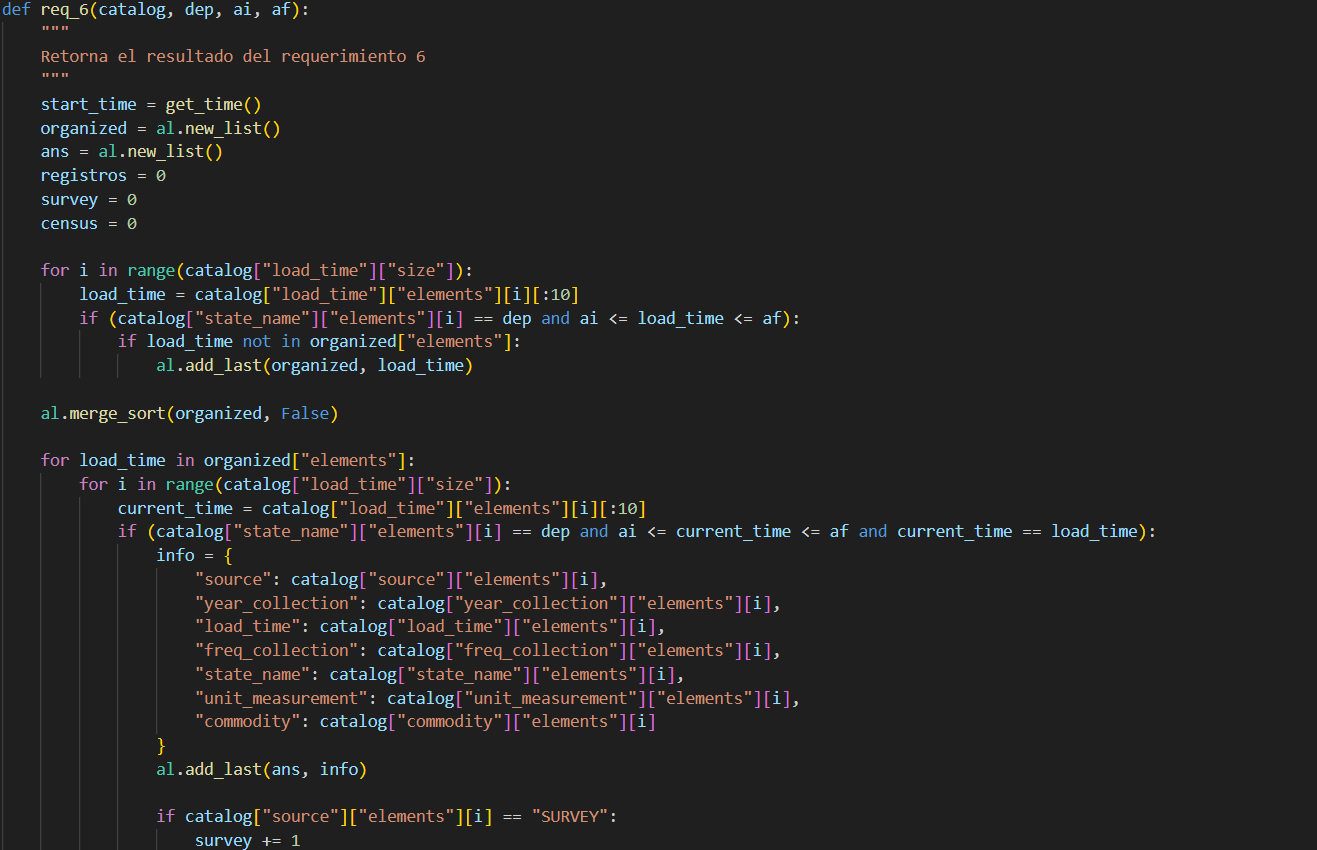
### **Graficas**



## **Análisis**

La grafica es coherente con el análisis ya que la relación entre el tamaño de la muestra y el tiempo de ejecución demuestra un comportamiento cuadrático. Esto se debe a que en la implementación del código hay un ciclo dentro de otro a la hora de seleccionar elementos validos ordenados.

# **Requerimiento <<6>>**



## **Descripción**

Este requerimiento esta encargado de buscar los registros en el catálogo que cumplan con un criterio de búsqueda de tiempo de subida y departamento. Primero crea las listas y contadores necesarias para su ejecución, lego da un recorrido al catalogo para extraer los tiempos de subida validos en una lista diferente, se organiza la lista y se recorre el catalogo para cada uno de los tiempos de carga almacenados. Los registros que coincidan con estas fechas de carga y departamentos se agregan a la lista de resultado. Al añadir un registro a la lista incrementan los contadores. Se comprueba si hay que acortar la lista y se retorna el resultado.

|  |  |
| --- | --- |
| **Entrada** | Catalogo, departamento, fecha inicio, fecha fin |
| **Salidas** | Listado de registros, cantidad de tipo census, cantidad de tipo survey, cantidad de registros, tiempo de ejecución. |
| **Implementado (Sí/No)** | Si, Tomas Aponte |

## **Análisis de complejidad**

|  |  |
| --- | --- |
| **Pasos** | **Complejidad** |
| Crea listas y contadores necesarios | O(1) |
| Se recorre el catálogo para tiempos de carga validos | O(n) |
| Se añaden los tiempos a una lista | O(1) |
| Se organiza la lista | O(nlog(n) |
| Se recorre el catalogo para cada tiempo de carga | O(n\*\*2) |
| Se añaden los registros validos a una lista | O(1) |
| Incrementan los contadores necesarios | O(1) |
| Se acorta la lista si es necesario | O(1) |
| Se retornan los resultados | O(1) |
| **TOTAL** | O(n) |

## **Pruebas Realizadas**

Fecha inicial: 2010-05-02

Fecha final: 2017-09-20

Departamento de interés: SOUTH CAROLINA

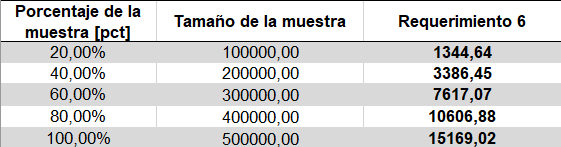
Procesador: Ryzen 7 7730U

RAM: 16GB

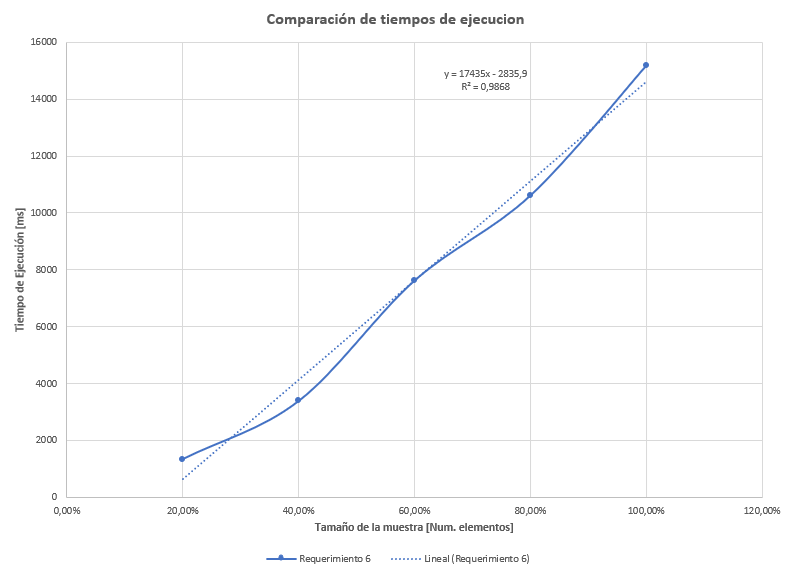
Sistema Operativo: Windows 11

|  |  |
| --- | --- |
| **Entrada** | **Tiempo (s)** |
| 20% | 1344,64 |
| 40% | 3386,45 |
| 60% | 7617,07 |
| 80% | 10606,88 |
| 100% | 15169,02 |

### **Tablas de datos**



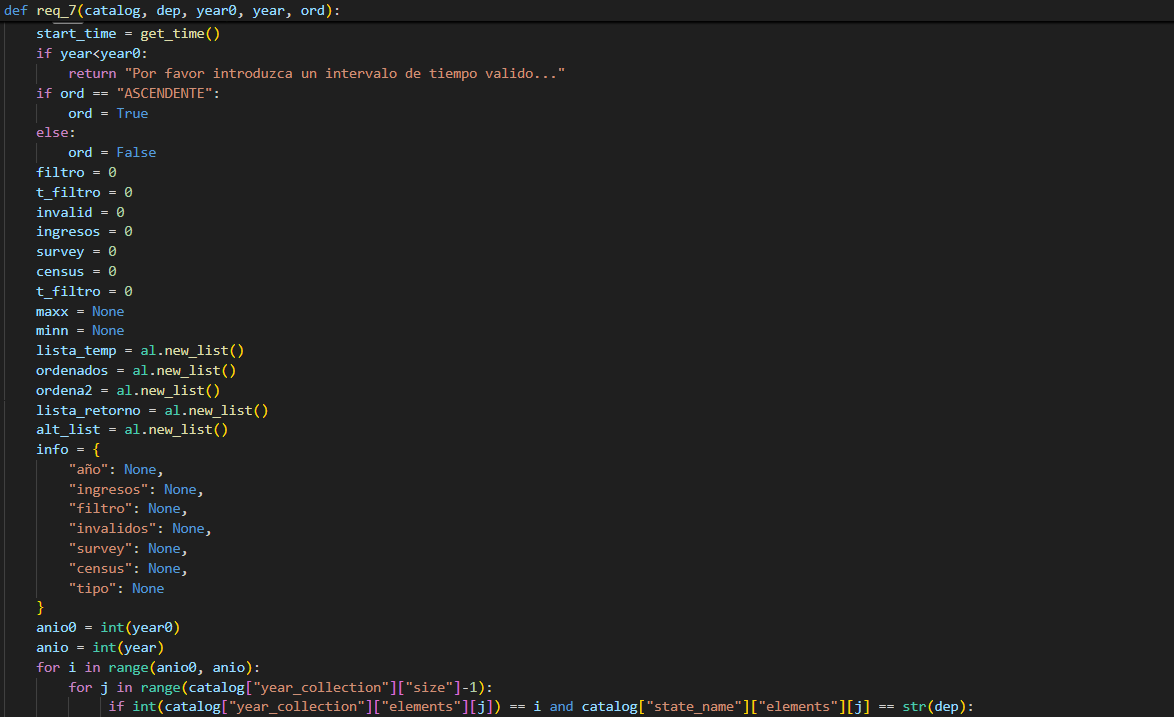
### **Graficas**



## **Análisis**

La relación entre el tamaño de la muestra y el tiempo de ejecución muestra un comportamiento cuadrático, lo cual coincide con lo planteado en el análisis de complejidad. Esto tiene sentido gracias a que dentro de la función hay un ciclo dentro de otro, lo cual causa que el programa tenga que recorrer toda la lista en su completitud una cantidad de veces considerable.

# Requerimiento <<7>>



## Descripción

Esta función se encarga de retornar una lista con información pertinente para cada año dentro de un rango de tiempo especificado por el usuario, para un departamento dado y ordenado de acuerdo a las preferencias del usuario. La función, para cada año que se encuentre dentro del rango, bisca dentro del catálogo ciertos parámetros para después agregarlos a un diccionario “info” que además se agrega a una lista temporal, para después ordenar los datos de acuerdo a los ingresos que se encontraron, y si se encuentran 2 o más ingresos iguales, estos se ordenan de acuerdo a la cantidad de registros encontrados para el año dado. Posterior a esto, se acomodan los diccionarios ordenadamente en la lista de retorno, y se verifica el tamaño de la lista. Si esta supera los 15 elementos, se retornan los primeros y últimos 5.

|  |  |
| --- | --- |
| **Entrada** | Catálogo con los datos del archivo csv, el departamento deseado, el intervalo de tiempo deseado y el tipo de ordenamiento de los datos |
| **Salidas** | Totalidad de registros que cumplieron el filtro, lista con la información ordenada por año y tiempo de ejecución. |
| **Implementado (Sí/No)** | Si, implementado por Juan García |

## Análisis de complejidad

Análisis de complejidad de cada uno de los pasos del algoritmo

|  |  |
| --- | --- |
| **Pasos** | **Complejidad** |
| Comparaciones preliminares | O(1) |
| Asignación variables | O(1) |
| Recorrido total del catálogo por los años del parámetro | O(n2) |
| Asignación **info** | O(1) |
| Añadir **info**  a **lista\_temp** | O(1) |
| Recorrido de **lista\_temp** para agregar elementos a ordenar a lista **ordenados** | O(n) |
| Ordenamiento de **ordenados** | O(n log n) |
| Determinación **maxx** y **minn** | O(n) |
| Asignar **maxx** y **minn** a sus respectivos **info** | O(n) |
| Determinar si existen duplicados en **ordenados** | O(n) |
| Posible agregado de datos adicionales a **ordena2** | O(n) |
| Posible ordenamiento de **ordena2** | O(n log n) |
| Ciclos para añadir **info** ordenados en **lista\_retorno** | O(n2) |
| Verificación tamaño **lista\_retorno** | O(1) |
| Posible agregado a **alt\_retorno** | O(n) |
| ***TOTAL*** | ***O(n2)*** |

## Pruebas Realizadas

Año de inicio: 2007

Año de fin: 2015

Departamento: SOUTH CAROLINA

Orden: ASCENDENTE

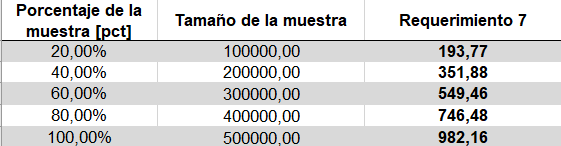
Procesador: Ryzen 7 7730U

RAM: 16GB

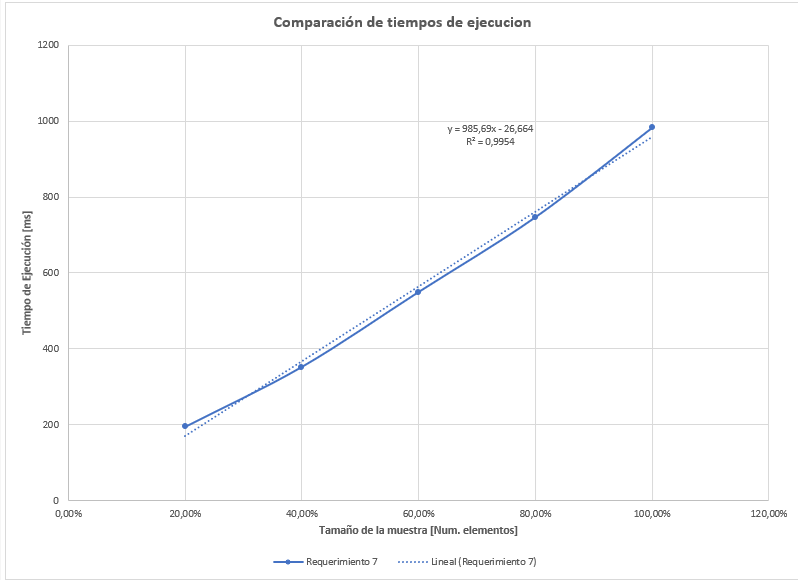
Sistema Operativo: Windows 11

|  |  |
| --- | --- |
| **Entrada** | **Tiempo (s)** |
| 20% | **193,77** |
| 40% | **351,88** |
| 60% | **549,46** |
| 80% | **746,48** |
| 100% | **982,16** |

### Tablas de datos



### Gráficas



## Análisis

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