**ANÁLISIS DEL RETO**

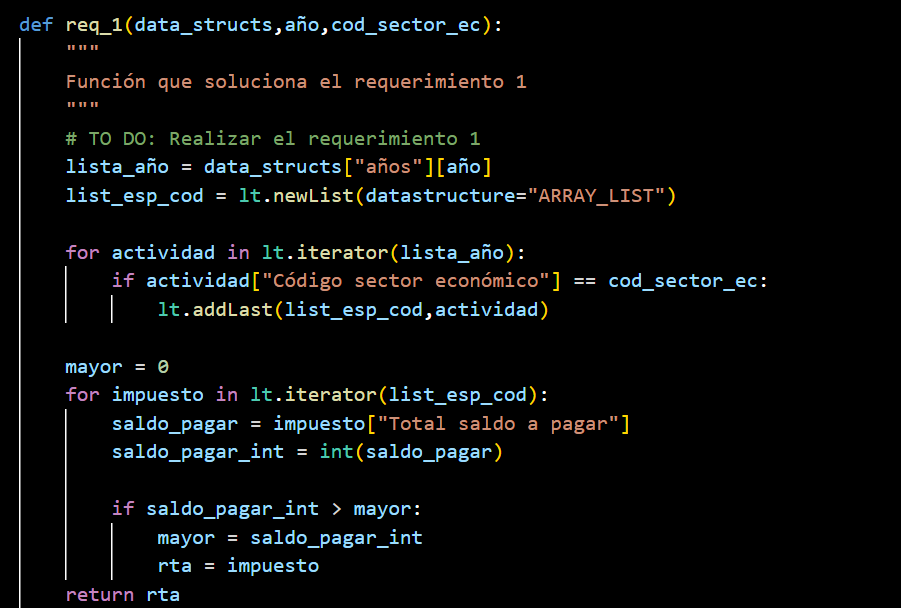
Jacobo Zarruk, 202223913, j.zarruk@uniandes.edu.co

María José Amorocho, 202220179, m.amorocho@uniandes.edu.co

# **Requerimiento 1**

Obtener la actividad económica con mayor saldo a pagar para un sector económico y un año específico

## **Descripción**



Para cumplir con el requerimiento primero se obtiene la lista de actividades de ese año. Luego se recorre esta lista para buscar las actividades económicas que coinciden con el sector dado por parámetro, y posteriormente se comparan por su saldo a pagar para obtener la actividad con mayor valor.

|  |  |
| --- | --- |
| **Entrada** | Año, código sector económico, estructura de datos |
| **Salidas** | Actividad económica con mayor saldo a pagar perteneciente al año y código económico establecidos |
| **Implementado (Sí/No)** | Implementado por María José |

## **Análisis de complejidad**

Análisis de complejidad de cada uno de los pasos del algoritmo

|  |  |
| --- | --- |
| **Pasos** | **Complejidad** |
| Acceder a la lista de actividades económicas del año específico | O(1) |
| Crear una lista para guardar las actividades económicas de ese año con el código del sector económico buscado | O(1) |
| Recorrer la lista de actividades económicas del año. Si su código de sector económico es el mismo que el código dado por parámetro, se agrega a la lista creada para almacenar las actividades económicas con el sector económico buscado. | O(n) |
| Se recorre la lista que tiene las actividades con el mismo código del sector económico buscado por parámetro para comprar cada impuesto por su [“Total saldo a pagar”]. Si el saldo a pagar es mayor a la variable “mayor”, se establece esa actividad como respuesta (y se actualiza el valor de “mayor”). | O(n) |
| ***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).

Las pruebas realizadas fueron realizadas en una maquina con las siguientes especificaciones. El año dado por parámetro fue 2016 usando el sector 1.

|  |  |
| --- | --- |
| Procesadores | Intel(R) Core(TM) i5-10210U CPU @1.60GHz 2.10 GHz |
| Memoria RAM | 8 GB |
| Sistema Operativo | Windows 10 |

|  |  |  |
| --- | --- | --- |
| **Entrada** | **Tiempo (ms)** | **Memoria (kB)** |
| small | 0,18 | 1,48 |
| 5 pct | 0,22 | 1,48 |
| 10 pct | 0,19 | 1,48 |
| 20 pct | 0,24 | 1,48 |
| 30 pct | 0,30 | 1,48 |
| 50 pct | 0,37 | 1,48 |
| 80 pct | 0,45 | 1,48 |
| large | 0,79 | 1,48 |

### **Tablas de datos**

Las tablas con la recopilación de datos de las pruebas.

|  |  |  |
| --- | --- | --- |
| **Muestra** | **Salida (código actividad económica)** | **Tiempo (ms)** |
| small | 130 | 0,18 |
| 5 pct | 125 | 0,22 |
| 10 pct | 125 | 0,19 |
| 20 pct | 125 | 0,24 |
| 30 pct | 125 | 0,30 |
| 50 pct | 125 | 0,37 |
| 80 pct | 125 | 0,45 |
| large | 125 | 0,79 |

### **Graficas**

Las gráficas con la representación de las pruebas realizadas.

## **Análisis**

Se puede observar que el algoritmo tiene una complejidad de O(n) en cuanto al tiempo de ejecución, pues en la gráfica se observa que los datos siguen aproximadamente una tendencia lineal. Lo anterior se debe a que, para un año en específico, se deben recorrer las n actividades asociadas a ese año en búsqueda de aquellas actividades con el subsector deseado para ser introducidas en una lista. Esta lista también es recorrida buscando una actividad económica específica. En el peor caso, todas las actividades económicas asociadas a ese año en particular pertenecerán al mismo sector y la actividad económica buscada se hallará en la última posición de la lista, por lo que se tendrán que recorrer todas las anteriores actividades.

En cuanto al uso de memoria, no se encuentra distinción y este permanece constante sin importar la cantidad de datos de entrada.

# **Requerimiento 3**

Encontrar el subsector económico con el menor total de retenciones para un año específico

## **Descripción**



Para cumplir con el requerimiento se busca la lista de actividades económicas correspondiente a un año específico. La información de esta lista se ordena en un mapa que tiene como llaves el numero de un subsector económico, y como valor una lista de actividades económicas de ese subsector. De cada lista se obtiene la suma de retenciones de las actividades económicas y se extrae suma de menor valor junto con el número del sector económico al que pertenece esta cantidad. Posteriormente se obtienen los diferentes totales del subsector que son solicitados y se toman las actividades de mayor y menor aporte al total de retenciones.

|  |  |
| --- | --- |
| **Entrada** | Estructura de datos, año |
| **Salidas** | Tupla con la información del subsector económico con menores retenciones en el año dado y las actividades económicas con mayores y menores aportes al total de retenciones. |
| **Implementado (Sí/No)** | Si. Implementado por María José |

## **Análisis de complejidad**

Análisis de complejidad de cada uno de los pasos del algoritmo

|  |  |
| --- | --- |
| **Pasos** | **Complejidad** |
| Obtener la lista de actividades económicas pertenecientes a un año específico. | O(1) |
| Se llama a la función Sort\_list\_year\_subsector que recorre la lista de actividades económicas en un año y las organiza en un mapa (map\_sort\_subsector). Las llaves de estos mapas corresponden al número de un subsector económico y como valor tienen una lista de las actividades económicas con ese mismo subsector. | O(1) |
| Se crea el mapa map\_subsectores\_y\_totales que tendrá como llave el código de un subsector económico y como valor el total de retenciones para ese subsector. | O(1) |
| Se recorre el mapa map\_sort\_subsector. Por cada subsector se toma su valor (es decir, una lista con las actividades económicas de este subsector) y se llama a la función sumar\_elementos, que recorre una lista dada sumando un criterio brindado por parámetro. En este caso se suman el total de retenciones de cada actividad económica por cada subsector en el mapa. Cuando se obtiene el total de retenciones de un subsector, se añade al mapa map\_subsectores\_y\_totales. Como llave, este valor tendrá el número del subsector económico que se recorrió. | O(n) |
| Se llama a la función mayor\_o\_menor\_mapa. Esta función retorna el mayor o el menor valor de un mapa (según lo que se indique) y su llave correspondiente (en una tupla). La función mayor\_o\_menor\_mapa se aplica sobre map\_subsectores\_y\_totales para obtener el código del subsector económico con menor total de retenciones y el valor numérico de esta cantidad. | O(1) |
| Habiendo obtenido el código del subsector económico con menor retenciones, se busca la lista de actividades asociadas a este en el mapa map\_sort\_subsector, usando las funciones mp.get() y me.getValue(). La lista de actividades económicas se ordena ascendentemente con respecto al total de retenciones de cada una. | O(1) |
| Se llama a la función obtener\_totales\_subsector para calcular los totales solicitados del subsector con menor total de retenciones. obtener\_totales\_subsector se encarga de recorrer las actividades económicas de una lista y sumar el total de ingresos netos, costos y gastos, saldo por pagar y saldo a favor de cada una de estas. Estos totales son guardados en un diccionario, que tiene como llaves el título del parámetro calculado (p.ej. “Total ingresos netos”) y como valor la suma obtenida de este criterio. El diccionario con esta información se guarda en la variable info\_totales\_subsect. | O(1) |
| Para obtener las actividades con mayores y menores aportes al total de retenciones se toma la lista de actividades económicas ordenada (lista\_act\_sub\_sec\_menor\_ret) que contiene las actividades económicas pertenecientes al subsector con menor total de retenciones. Si el tamaño de la lista es menor a 6, se retorna la lista ordenada. Si tiene más de 6 elementos se llama a la función get\_3\_last\_and\_first\_list que retorna una lista con los tres primeros y 3 últimos elementos de una lista dada por parámetro. | O(1) |
| Se retorna el diccionario con la información de totales del sector económico con menos retenciones y una lista de las actividades económicas con mayores y menores aportes, en una tupla. | O(n) |
| ***TOTAL*** | ***O(n)*** |

## **Pruebas Realizadas**

Las pruebas realizadas fueron realizadas en una maquina con las siguientes especificaciones. Como año se tomó el 2013

|  |  |
| --- | --- |
| Procesadores | Intel(R) Core(TM) i5-10210U CPU @ 1.60GHz 2.10 GHz |
| Memoria RAM | 8 GB |
| Sistema Operativo | Windows 10 |

|  |  |  |
| --- | --- | --- |
| **Entrada** | **Tiempo (ms)** | **Memoria (kB)** |
| small | 6,80 | 145,47 |
| 5 pct | 13,06 | 147,52 |
| 10 pct | 11,46 | 147,54 |
| 20 pct | 15,83 | 150,52 |
| 30 pct | 19,54 | 151,22 |
| 50 pct | 30,82 | 152,14 |
| 80 pct | 54,80 | 152,67 |
| large | 45,13 | 155,51 |

### **Tablas de datos**

Las tablas con la recopilación de datos de las pruebas.

|  |  |  |
| --- | --- | --- |
| **Muestra** | **Salida (total retenciones subsector)** | **Tiempo (ms)** |
| small | 1190 | 6,80 |
| 5 pct | 13 | 13,06 |
| 10 pct | 13 | 11,46 |
| 20 pct | 13 | 15,83 |
| 30 pct | 13 | 19,54 |
| 50 pct | 3734 | 30,82 |
| 80 pct | 17 | 54,80 |
| large | 7 | 45,13 |

### **Graficas**

## **Análisis**

La complejidad temporal para el requerimiento 3 se asemeja a O(n), pues los datos no se encuentran muy dispersos respecto a la tendencia lineal graficada. No obstante, hay dos aspectos importantes a considerar: primeramente, aunque el algoritmo en sí tiene un orden aproximado de O(n) cabe resaltar que las funciones que se llaman dentro del mismo (como sort\_list\_year\_subsector, find\_menor\_o\_mayor\_mapa, obtener\_totales\_subsector y demás) tienen internamente una complejidad de O(n), por lo que, si se toma en conjunto, el orden total del algoritmo sería de O(); Estas funciones “auxiliares” se crearon con el fin de facilitar y optimizar el código, pues se identificó que serían útiles en más de un caso. En segundo lugar, como se observa en la gráfica de Tiempo de ejecución vs tamaño datos, hay una variación significativa entre el procesamiento del 80% de los datos y el total de datos, pues toma menor tiempo procesar el archivo de mayor tamaño.

Respecto al uso de memoria, se puede apreciar que se sigue, en gran parte, una tendencia lineal O(n), lo que es coherente teniendo en cuenta las iteraciones que deben hacerse y el volumen de datos manejado.

# **Requerimiento 4**

Encontrar el subsector económico con los mayores costos y gastos de nómina para un año específico

## **Descripción**

Texto

Descripción generada automáticamente

Para cumplir con el requerimiento se busca la lista de actividades económicas correspondiente a un año específico. La información de esta lista se ordena en un mapa que tiene como llaves el numero de un subsector económico, y como valor una lista de actividades económicas de ese subsector. De cada lista se obtiene la suma de costos y gastos de nómina de las actividades económicas y se extrae la suma de mayor valor junto con el número del sector económico al que pertenece esta cantidad. Posteriormente se obtienen los diferentes totales del subsector que son solicitados y se toman las actividades de mayor y menor aporte al total de costos y gastos de nómina.

|  |  |
| --- | --- |
| **Entrada** | Estructura de datos, año |
| **Salidas** | Tupla con la información del subsector económico con mayores costos y gastos de nómina en el año dado y las actividades económicas con mayores y menores aportes al total de costos y gastos de nómina. |
| **Implementado (Sí/No)** | Si. Implementado por María José |

## **Análisis de complejidad**

Análisis de complejidad de cada uno de los pasos del algoritmo

|  |  |
| --- | --- |
| **Pasos** | **Complejidad** |
| Obtener la lista de actividades económicas pertenecientes a un año específico. | O(1) |
| Se llama a la función Sort\_list\_year\_subsector que recorre la lista de actividades económicas en un año y las organiza en un mapa (map\_sort\_subsector). Las llaves de estos mapas corresponden al número de un subsector económico y como valor tienen una lista de las actividades económicas con ese mismo subsector. | O(1) |
| Se crea el mapa map\_subsectores\_y\_totales que tendrá como llave el código de un subsector económico y como valor el total de costos y gastos de nómina para ese subsector. | O(1) |
| Se recorre el mapa map\_sort\_subsector. Por cada subsector se toma su valor (es decir, una lista con las actividades económicas de este subsector) y se llama a la función sumar\_elementos, que recorre una lista dada sumando un criterio brindado por parámetro. En este caso se suman el total de costos y gastos de nómina de cada actividad económica por cada subsector en el mapa. Cuando se obtiene el total de retenciones de un subsector, se añade al mapa map\_subsectores\_y\_totales. Como llave, este valor tendrá el número del subsector económico que se recorrió. | O(n) |
| Se llama a la función mayor\_o\_menor\_mapa. Esta función retorna el mayor o el menor valor de un mapa (según lo que se indique) y su llave correspondiente (en una tupla). La función mayor\_o\_menor\_mapa se aplica sobre map\_subsectores\_y\_totales para obtener el código del subsector económico con mayor total de costos y gastos de nómina y el valor numérico de esta cantidad. | O(1) |
| Habiendo obtenido el código del subsector económico con mayor total de costos y gastos de nómina, se busca la lista de actividades asociadas a este en el mapa map\_sort\_subsector, usando las funciones mp.get() y me.getValue(). La lista de actividades económicas se ordena ascendentemente con respecto al total de costos y gastos de nómina de cada una. | O(1) |
| Se llama a la función obtener\_totales\_subsector para calcular los totales solicitados del subsector con mayor total de costos y gastos de nómina. obtener\_totales\_subsector se encarga de recorrer las actividades económicas de una lista y sumar el total de ingresos netos, costos y gastos, saldo por pagar y saldo a favor de cada una de estas. Estos totales son guardados en un diccionario, que tiene como llaves el título del parámetro calculado (p.ej. “Total ingresos netos”) y como valor la suma obtenida de este criterio. El diccionario con esta información se guarda en la variable info\_totales\_subsect. | O(1) |
| Para obtener las actividades con mayores y menores aportes al total de retenciones se toma la lista de actividades económicas ordenada (lista\_act\_sub\_sec\_menor\_ret) que contiene las actividades económicas pertenecientes al subsector con mayor total de costos y gastos de nómina. Si el tamaño de la lista es menor a 6, se retorna la lista ordenada. Si tiene más de 6 elementos se llama a la función get\_3\_last\_and\_first\_list que retorna una lista con los tres primeros y 3 últimos elementos de una lista dada por parámetro. | O(1) |
| Se retorna el diccionario con la información de totales del sector económico con mayor total de costos y gastos de nómina y una lista de las actividades económicas con mayores y menores aportes, en una tupla. | O(n) |
| ***TOTAL*** | ***O(n)*** |

## **Pruebas Realizadas**

Las pruebas realizadas fueron realizadas en una maquina con las siguientes especificaciones. Como año se tomó el 2021

|  |  |
| --- | --- |
| Procesadores | Intel(R) Core(TM) i5-10210U CPU @ 1.60GHz 2.10 GHz |
| Memoria RAM | 8 GB |
| Sistema Operativo | Windows 10 |

|  |  |  |
| --- | --- | --- |
| **Entrada** | **Tiempo (ms)** | **Memoria (kB)** |
| small | 7,56 | 147,02 |
| 5 pct | 13,56 | 148,94 |
| 10 pct | 18,13 | 152,71 |
| 20 pct | 21,46 | 153,00 |
| 30 pct | 23,66 | 153,5 |
| 50 pct | 31,98 | 155,59 |
| 80 pct | 39,66 | 156,29 |
| large | 46,29 | 158,59 |

### **Tablas de datos**

Las tablas con la recopilación de datos de las pruebas.

|  |  |  |
| --- | --- | --- |
| **Muestra** | **Salida (total costos y gastos de nómina subsector)** | **Tiempo (ms)** |
| small | 783223 | 7,56 |
| 5 pct | 2200949 | 13,56 |
| 10 pct | 8210362 | 18,13 |
| 20 pct | 10299389 | 21,46 |
| 30 pct | 15114866 | 23,66 |
| 50 pct | 20328834 | 31,98 |
| 80 pct | 30967827 | 39,66 |
| large | 35755662 | 46,29 |

### **Graficas**

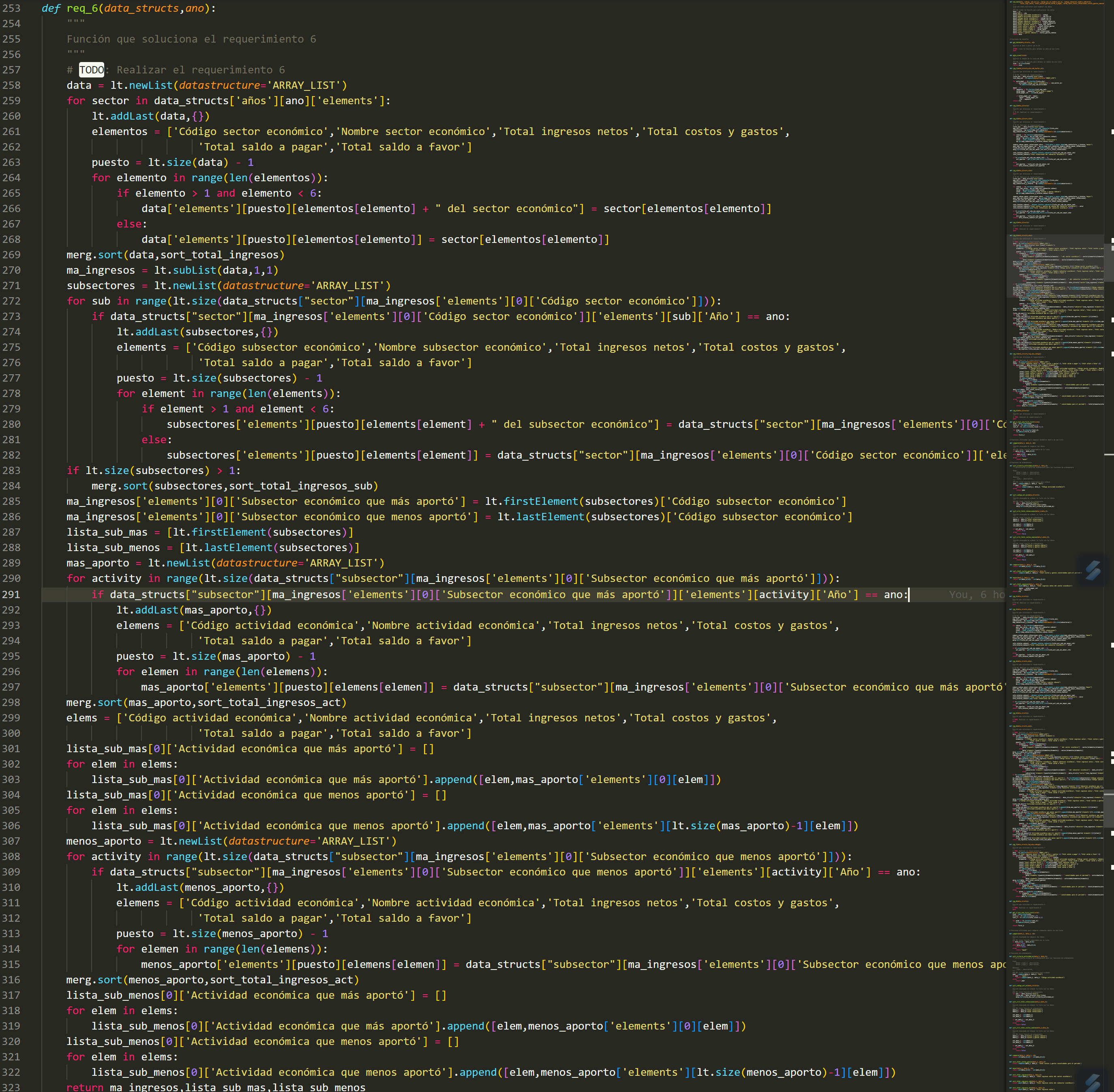
## **Análisis**

La complejidad temporal para el requerimiento 4 se asemeja a O(n), pues los datos se encuentran significativamente alineados respecto a la tendencia lineal graficada. No obstante, vale recalcar que, como en el requerimiento anterior, aunque el algoritmo en sí tiene un orden aproximado de O(n), las funciones “auxiliares” que se llaman dentro del mismo (como sort\_list\_year\_subsector, find\_menor\_o\_mayor\_mapa, obtener\_totales\_subsector y demás) tienen internamente una complejidad de O(n), por lo que, si se toma en conjunto, el orden total sería de O(). Respecto al consumo de memoria, también se observa una tendencia lineal. Se puede apreciar que las gráficas hechas para este requerimiento comparten varias similitudes con las del requerimiento 3, de manera que las ecuaciones para las líneas de tendencia son parecidas; lo anterior debido a que, prácticamente se usa el mismo método de resolución, sólo que con parámetros diferentes.

# **Requerimiento 6**

## Encontrar el sector económico con el mayor total de ingresos netos para un año específico.

## **Descripción**



Para cumplir con el requerimiento se empieza creando una nueva lista, luego se mira cada uno de los sectores que se encuentran en el año preestablecido, luego se añaden a la lista previamente creada con solo los datos necesarios para la impresión de la tabla. Después de esto se ordena esta lista de mayor a menor teniendo en cuenta el total de ingresos netos de cada sector, a continuación, procedemos a crear una sub lista con solo el primer sector que se encuentra en la lista anterior pues este es el sector con mayor total de ingresos netos, lo cual es lo que piden. De la misma manera, creo otra lista nueva, posteriormente miro cada uno de los subsectores que se encuentran en el sector previamente encontrado, más adelante hago lo mismo de agregarlos en la lista solo teniendo en cuenta los datos necesarios para la posterior impresión de la tabla, ordeno todos los datos de esta lista, si esta es tiene más de un dato dentro, de mayor a menor dado el total de ingresos netos, dado así que el subsector que más aportó es el que se encuentra en la primera posición y el que menos aportó en la ultima posición. Posterior a esto añado el código de ambos subsectores a la anterior lista del sector, siguiente a esto se crean dos nuevas listas dadas por distintas listas, en vez de ser un diccionario con distintas llaves, para poder hacer la correcta impresión posteriormente, en estas listas se añade lo necesario de cada uno de las actividades económicas que más y menos aportaron a cada uno de los subsectores para luego esto ser añadido a la listas propias de los subsectores que más aportó y que menos aportó.

|  |  |
| --- | --- |
| **Entrada** | Año, estructura de datos |
| **Salidas** | Sector económico con mayor total de ingresos netos para el año establecido, subsector económico que más aportó, subsector económico que menos aportó, actividad económica que más aportó para cada uno de los subsectores y actividad económica que menos aportó para cada uno de los subsectores. |
| **Implementado (Sí/No)** | Si. Implementado por Jacobo |

## **Análisis de complejidad**

Análisis de complejidad de cada uno de los pasos del algoritmo

|  |  |
| --- | --- |
| **Pasos** | **Complejidad** |
| Crear una lista para guardar los sectores económicos de ese año. | O(1) |
| Recorrer la lista de sectores económicos del año. Se agrega únicamente los datos necesarios para la correcta impresión de la tabla. | O(n) |
| Se ordena la lista de mayor a menor teniendo en cuenta el total de ingresos netos de cada sector económico. | O(n log n) |
| Crear una sub lista para guardar el sector económico con mayor total de ingresos netos. | O(1) |
| Crear una lista para guardar los subsectores económicos del sector. | O(1) |
| Recorrer la lista de subsectores económicas del sector económico. Se agrega únicamente los datos necesarios para la correcta impresión de la tabla. | O(n) |
| Se ordena la lista de mayor a menor teniendo en cuenta el total de ingresos netos de cada subsector económico. | O(n log n) |
| Se agrega el código del subsector que más aportó y del que menos aportó al sector económico | O(n) |
| Crear una lista guardando el subsector económico que más aportó. | O(1) |
| Crear una lista guardando el subsector económico que menos aportó. | O(1) |
| Crear una lista para guardar las actividades económicas del subsector que más aportó. | O(1) |
| Recorrer la lista de actividades económicas del subsector económico que más aporto. Se agrega únicamente los datos necesarios para la correcta impresión de la tabla. | O(n) |
| Se ordena la lista de mayor a menor teniendo en cuenta el total de ingresos netos de cada actividad económica. | O(n log n) |
| Se pasa los datos de la actividad económica que más aportó convirtiéndolos en una lista de listas dadas por lo que previamente era la llave y el valor de esta | O(6) |
| ***TOTAL*** | ***O(n log 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).

Las pruebas realizadas fueron realizadas en una máquina con las siguientes especificaciones. El año dado por parámetro fue 2021.

|  |  |
| --- | --- |
| Procesadores | AMD Ryzen 5 5625U with Radeon Graphics 2.30 GHz |
| Memoria RAM | 16.0 GB |
| Sistema Operativo | Windows 11 Home Single Language |

|  |  |  |
| --- | --- | --- |
| **Entrada** | **Tiempo (ms)** | **Memoria (kB)** |
| small | 0,35 | 4,15 |
| 5 pct | 1,25 | 6,40 |
| 10 pct | 1,71 | 7,78 |
| 20 pct | 4,25 | 8,15 |
| 30 pct | 6,21 | 8,46 |
| 50 pct | 8,29 | 8,65 |
| 80 pct | 12,77 | 8,72 |
| large | 15,15 | 8,72 |

### **Tablas de datos**

Las tablas con la recopilación de datos de las pruebas.

|  |  |  |
| --- | --- | --- |
| **Muestra** | **Salida (código sector económico)** | **Tiempo (ms)** |
| small | 8 | 0,35 |
| 5 pct | 6 | 1,25 |
| 10 pct | 6 | 1,71 |
| 20 pct | 6 | 4,25 |
| 30 pct | 6 | 6,21 |
| 50 pct | 8 | 8,29 |
| 80 pct | 8 | 12,77 |
| large | 8 | 15,15 |

### **Graficas**

Las gráficas con la representación de las pruebas realizadas.

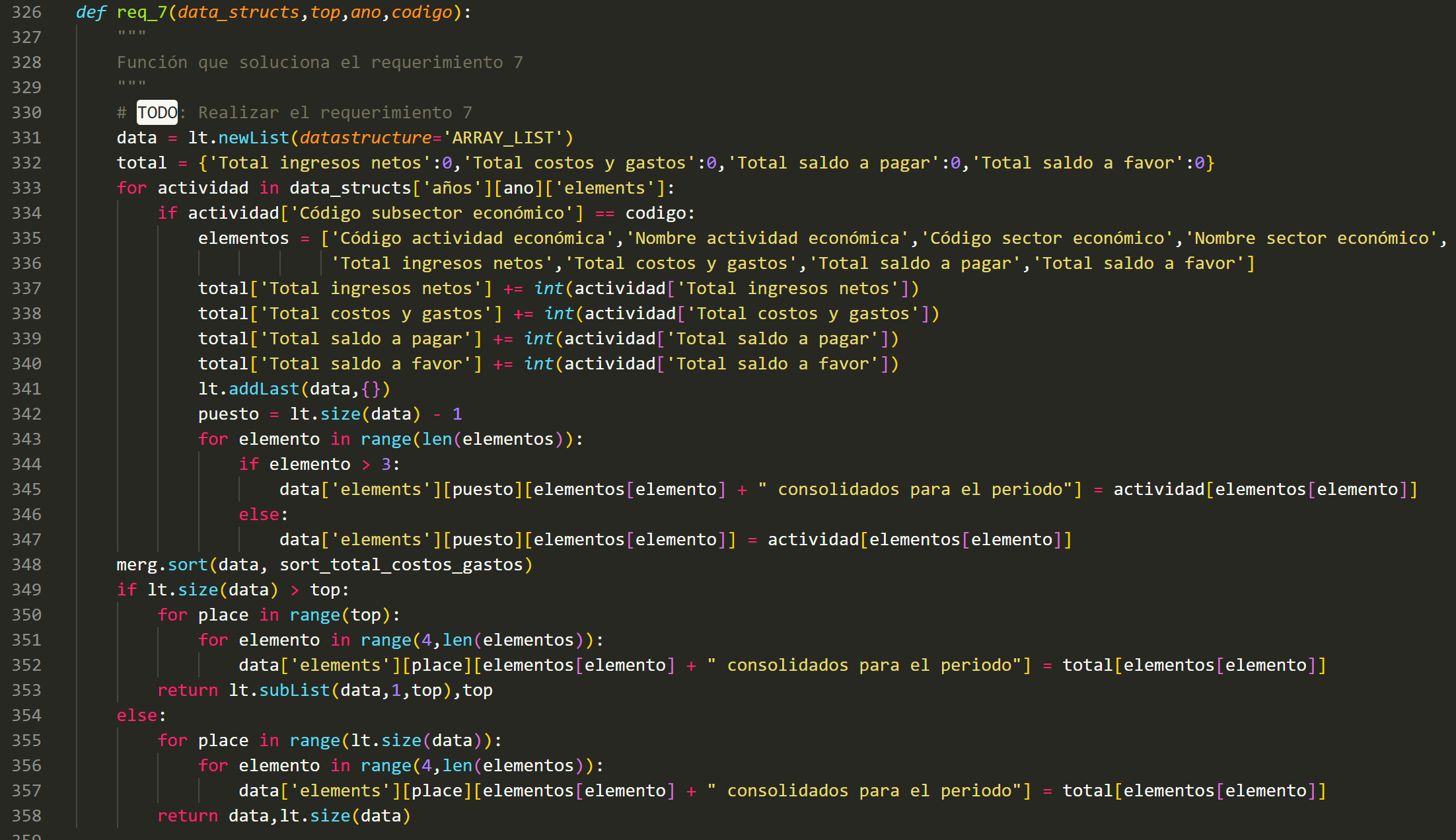
## **Análisis**

Se puede observar que el algoritmo tiene una complejidad de O(n log n), para el tiempo de ejecución, en la gráfica se observa una tendencia prácticamente logarítmica, pues dado a la complejidad del algoritmo, O(N\*log(N)), a medida que aumentan los datos la complejidad aumentan de una forma parecida a la logarítmica, pues no es solo el logaritmo, sino que también este se multiplica por la cantidad de datos. En cuanto al uso de memoria, es prácticamente radical, pues es parecida a la gráfica de una raíz.

# **Requerimiento 7**

Listar el TOP (N) de las actividades económicas con el menor total de costos y gastos para un subsector y un año específicos.

## **Descripción**



Para cumplir con el requerimiento se empieza creando una nueva lista, luego se crea un diccionario para guardar el total de ingresos, gastos y costos, salgo a pagar y saldo a favor para el periodo, después se mira cada actividad económica para el año previamente dado por el usuario y, con la ayuda de un condicional, nos aseguramos de que el código del subsector de la actividad económica sea el mismo al código dado por el usuario. Después, se crea una lista para posteriormente poder añadir a la lista creada al inicio solo los datos de la actividad necesarios para la impresión de la tabla, esto se logra mirando cada uno de los elementos dados en la lista, a su vez, se van sumando el total de ingresos, gastos y costos, salgo a pagar y saldo a favor para el periodo. Luego de todo esto se ordena la lista de menor a mayor, seguidamente, gracias a un condicional, miramos si la lista es mayor en tamaño a la cantidad de actividades pedidas. Finalmente nos aseguramos de cambiar el total de ingresos, gastos y costos, salgo a pagar y saldo a favor para cada actividad a la suma total de ingresos, gastos y costos, salgo a pagar y saldo a favor.

|  |  |
| --- | --- |
| **Entrada** | Año, código del subsector económico, cantidad de datos a listar y estructura de datos |
| **Salidas** | Una lista de una cantidad N dada por el Top de actividades económicas con el menor total de costos y gastos para un subsector y un año especifico |
| **Implementado (Sí/No)** | Si. Implementado por Jacobo |

## **Análisis de complejidad**

Análisis de complejidad de cada uno de los pasos del algoritmo

|  |  |
| --- | --- |
| **Pasos** | **Complejidad** |
| Crear una lista para guardar los sectores económicos de ese año. | O(1) |
| Crear un diccionario para guardar el total de ingresos, gastos y costos, salgo a pagar y saldo a favor para el periodo. | O(1) |
| Recorrer la lista de actividades económicas del año. | O(n) |
| Se usa un condicional para saber si el código del subsector económico de cada actividad es igual al código dado por el usuario. | O(n) |
| Se agrega únicamente los datos necesarios para la correcta impresión de la tabla. | O(n) |
| Se ordena la lista de menor a mayor teniendo en cuenta el total de costos y gastos. | O(n log n) |
| Se usa un condicional para saber si la lista es mayor en tamaño a la cantidad de actividades pedidas | O(1) |
| Se pasa los datos de la actividad económica que más aportó convirtiéndolos en una lista de listas dadas por lo que previamente era la llave y el valor de esta | O(4) |
| ***TOTAL*** | ***O(n log 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).

Las pruebas realizadas fueron realizadas en una máquina con las siguientes especificaciones. El año dado por parámetro fue 2020, el código de subsector económico fue 11 y la cantidad de elementos deseados en el top fueron 9.

|  |  |
| --- | --- |
| Procesadores | AMD Ryzen 5 5625U with Radeon Graphics 2.30 GHz |
| Memoria RAM | 16.0 GB |
| Sistema Operativo | Windows 11 Home Single Language |

|  |  |  |
| --- | --- | --- |
| **Entrada** | **Tiempo (ms)** | **Memoria (kB)** |
| small | 0,16 | 1,57 |
| 5 pct | 0,17 | 1,57 |
| 10 pct | 0,27 | 2,15 |
| 20 pct | 0,37 | 3,32 |
| 30 pct | 0,46 | 5,11 |
| 50 pct | 0,87 | 6,60 |
| 80 pct | 0,91 | 7,28 |
| large | 1,11 | 7,65 |

### **Tablas de datos**

Las tablas con la recopilación de datos de las pruebas.

|  |  |  |
| --- | --- | --- |
| **Muestra** | **Salida (código actividad económica)** | **Tiempo (ms)** |
| small | 6422 | 0,16 |
| 5 pct | 6422 | 0,17 |
| 10 pct | 6422, 6532 | 0,27 |
| 20 pct | 6422, 6423, 6521, 6532 | 0,37 |
| 30 pct | 6522, 6424, 6613, 6422, 6423, 6521, 6532 | 0,46 |
| 50 pct | 6522, 6615, 6629, 6424, 6421, 6621, 6613, 6422, 6492 | 0,87 |
| 80 pct | 6513, 6531, 6522, 6615, 6629, 6493, 6424, 6421, 6494 | 0,91 |
| large | 6614, 6513, 6531, 6522, 6491, 6615, 6496, 6629, 6611 | 1,11 |

### **Graficas**

Las gráficas con la representación de las pruebas realizadas.

## **Análisis**

Se puede observar que el algoritmo tiene una complejidad de O(n log n), para tanto el tiempo de ejecución como el uso de memora, en las gráficas se observa una tendencia prácticamente logarítmica, pues dado a la complejidad del algoritmo, O(N\*log(N)), a medida que aumentan los datos la complejidad aumentan de una forma parecida a la logarítmica, pues no es solo el logaritmo, sino que también este se multiplica por la cantidad de datos.