Documento de análisis

- **Nombres:**

Estudiante 1: Samuel Josué Freire Tarazona, 202111460, s.freire@uniandes.edu.co ---------->

Requerimiento 3 (Individual) = Samuel Josué Freire Tarazona

Estudiante 2: José David Martínez Oliveros, 202116677, jd.martinezo1@uniandes.edu.co---------->

Requerimiento 4 (Individual)= José David Martínez Oliveros

- **Análisis de complejidad:**
  - **Requerimiento 1:**
    - Primera parte grande:

```
años = mp.keySet(catalogo['years'])
```

      - O(n)
    - Segunda parte grande:

```
for c in lt.iterator(años):
    if int(c) >= int(año1) and int(c) <= int(año2):
        valor = mp.get(catalogo['years'],c)
        for i in lt.iterator(me.getValue(valor)['books']):
            lt.addLast(nueva,i)
orden = sortnacidos(nueva)
```

      - O(n^2)
    - Tercera parte grande:

```
orden = sortnacidos(nueva)
```

      - O(nlog$n$)
    - COMPLEJIDAD GENERAL: O(n^2)

- **Requerimiento 2:**
  - Primera parte grande:

```
llaves = (mp.keySet(catalog['AdquisicionFecha']))
```

    - O(n)
  - Segunda parte grande:

```
for c in lt.iterator(llaves):
    if c >= fecha1 and c <= fecha2:
        valor = mp.get(catalog['AdquisicionFecha'],c)
        lista = me.getValue(valor)['books']
        for j in lt.iterator(lista):
            lt.addLast(nueva,j)
```

    - O(n^2)
  - Tercera parte grande:

```
orden = sortobras(nueva)
```

    - O(nlog$n$)
  - COMPLEJIDAD GENERAL: O(n^2)

- **Requerimiento 3:**
  - Primera parte grande:

```
valores = mp.get(catalog['Nombres_Artistas'],Artista)
valores_especificos = mp.get(catalog['Codigos_Artistas'],me.getValue(valores))
```

    O (k)

  - Segunda parte grande:

```
def cantidad_tecnicas(artistas):

    cantidad_de_tecnicas_veces = lt.newList('AR
    tecnicas_final = lt.newList('ARRAY_LIST')
    for partes in lt.iterator(artistas['obras']
        lt.addLast(tecnicas_final,partes['Mediu

    for i in lt.iterator(tecnicas_final):
        posauthor = lt.isPresent(cantidad_de_te
        if posauthor > 0:
            artista = lt.getElement(cantidad_de_
            artista['Cantidad'] += 1
        else:
            artista = newTecnica(i)
            artista['Cantidad'] = 1
            lt.addLast(cantidad_de_tecnicas_vec

    k = 0
    for p in lt.iterator(cantidad_de_tecnicas_v
        if int(p['Cantidad']) > k:
            k = p['Cantidad']
            maximo = p['Tecnica']

    return maximo,cantidad_de_tecnicas_veces
```

    - O(n)

  - Tercera Parte grande:

```
tecnicas_orden = sortCantidades(tecnicas[1])
```

    - O(nlog$n$)

  - Cuarta parte grande:

```
for obra in lt.iterator(me.getValue(valores_especificos)['obras']):
    if obra['Medium'] == tecnicas[0]:
        lt.addLast(obras,obra)
```

    - O(n)

  - COMPLEJIDAD GENERAL: O(n)

- **Requerimiento 5:**
  - Primera parte grande:

```
total_departamento = mp.get(catalog['Departamento'],departamento)
obras_artista = me.getValue(total_departamento)['books']
```

    - O(k)
  - Segunda parte grande:

```
def calculo_de_transporte(catalog):

    obras = lt.newList('ARRAY_LIST')
    for obra in lt.iterator(catalog):

        peso = obra['Weight (kg)']
        altura = obra['Height (cm)']
        ancho = obra['Width (cm)']
        profundidad = obra['Depth (cm)']
        longitud = obra['Length (cm)']
        diametro = obra['Diameter (cm)']
```

    - O(n)
  - Tercera parte grande:

```
def sortcostos(catalog):
    orden = merge.sort(catalog, comparacostos)
    return orden
```

    - O(nlog$n$)
  - Cuarta parte grande:

```
def suma_costo(catalog):

    suma = 0
    for p in lt.iterator(catalog):
        suma += p['costo']

    return float(suma)
```

    - O(n)

- Quinta parte grande:

```
def suma_peso(catalog):

    suma = 0
    for p in lt.iterator(catalog):
        suma += float(p['peso'])

    return float(suma)
```

  - O(n)

- Sexta parte grande:

```
def obtener_antiguas(catalog):
    """
    Retorna los tres ultimos artistas cargados
    """
    ordenadas = sortantiguas(catalog)
    con_fecha = lt.newList()
    orden = lt.newList()
    for obra in lt.iterator(ordenadas):
        if obra['fecha'] != '':
            lt.addLast(con_fecha, obra)
    for cont in range(1, 6):
        arte = lt.getElement(con_fecha, cont)
        lt.addLast(orden, arte)
    return orden
```

  - O(n)

- COMPLEJIDAD GENERAL: O(n)

- **Pruebas de requerimiento:**

Reto 2				
Pruebas requerimiento 1				
	1800-1900	1900-1920	1945-1980	Promedio
Maquina 1	29,28	40,23	28,63	32,7133333
Maquina 2				0
Pruebas requerimiento 2				
	1900/01/01-2	1929/11/19-1	1950/12/01-1	Promedio
Maquina 1	5078,69	4698,36	6125,69	5300,91333
Maquina 2				0
Pruebas requerimiento 3				
	Libero badii	Chip Lord	Vladimir Bur	Promedio
Maquina 1	0	0	0	0
Maquina 2				0
Pruebas requerimiento 4				
	obras total	obras total	obras total	Promedio
Maquina 1	1686043,75	1465449,36	1654467,39	1601986,83
Maquina 2				0
Pruebas requerimiento 5				
	Drawings & I	Photography	Painting & Sc	Promedio
Maquina 1	8069,85	3150,69	328,16	3849,56667
Maquina 2				0

- **Comparación de tiempo de ejecución (Promedio):**

	Comparacion	
	Reto 1	Reto 2
Maquina 1	20,8366667	32,7133333
Maquina 2		
Maquina 1	2656,25	5300,91333
Maquina 2		
Maquina 1	20,8366667	0
Maquina 2		
Maquina 1	1601986,83	0
Maquina 2		
Maquina 1	3968,75333	3849,56667
Maquina 2		

- **Comparación de complejidades:**
  - **Requerimiento 1:**
    - Para el primer requerimiento, en el reto 1, dio una complejidad de O(n). Luego, para el reto 2 salió una complejidad de O(n^2). Para este caso, la complejidad del segundo reto salió mucho más compleja que la del primer reto. Esto se puede dar por la organización de los datos, ya que al tener una estructura de mapas buscar un rango en las llaves y sacra los valores se vuelve más complejo. Puesto que, para la función de búsqueda del rango, para el primer reto y su orden eran mucho más sencillos de manejar. Sin embargo, el hecho de poder tener una organización de los datos por medio de llaves tiende a ser mejor. Realmente, la complejidad se aumenta al querer ordenar los datos, ya que se usan los ordenamientos del Tad Lista.
  - **Requerimiento 2:**
    - Para el segundo requerimiento, en el reto 1, dio una complejidad de O(n$\log n$). Luego, para el reto 2, dio una complejidad de O(n^2). En este caso, la complejidad del reto 2, para este requerimiento, es mucho mayor. Puesto que, es recorrer casi los datos nuevamente. Sin embargo, el tiempo tiende a ser menor en el reto 2 en comparación al reto 1. La complejidad en este caso se aumenta, ya que al lograr ordenar las obras se tiende a realizar algo más. Esto de más, es el recorrido para poder ordenar los valores internos en las fechas usadas. Pasa algo parecido a lo mencionado en el párrafo anterior, que al tener una organización de mapas, para buscar un rango, es mucho mayor que en las listas.
  - **Requerimiento 3:**
    - Para el tercer requerimiento, en el reto1, dio una complejidad de O(n). Luego, para el reto 2, dio una complejidad de O(n). En este caso, diferente a los anteriores, ya que aquí las complejidades del reto 1 y 2 son iguales. En este caso, se denota que la búsqueda de los autores, tiende a ser similar. Puesto que, los recorridos tiende a ser iguales para lograr extraer su información. Para poder lograr extraer su información, es muy similar, ya que no toca entrar dos veces a la organización si ambos están relacionados de la misma manera. Realmente, el uso de Tad lista y mapas en este requerimiento es muy similar, y su implementación tiende a ser muy similar.

- **Requerimiento 5:**
  - Para el quinto requerimiento, en el reto 1, dio una complejidad de O(n). Luego, para el reto 2 salió un complejidad de O(n). En este caso las complejidades son iguales, como en el requerimiento 3. Para este caso, se facilitó por un lado la organización de las obras. Pero, el cálculo de su precio es muy similar. También, la complejidad tiende a ser igual, ya que se usan las Tad lista como valores principales. Realmente, cualquiera de las dos estructuras son muy similares. En las complejidades resulta, interesante que la búsqueda de elementos se asimilar. Puesto que, los mapas tiende a ser mejores en ese caso. Sin embargo, en este caso su implementación fue muy similar, en el sentido, de su complejidad.