Mateo Cote Canal m.cotec@uniandes.edu.co 202022609

Diego Acosta Corredor da.acostac1@uniandes.edu.co 202110516

DOCUMENTO DE ANALISIS

Requerimiento #3: Mateo Cote Canal

Requerimiento #4: Diego Acosta Corredor

1. Análisis de complejidad:

    - Requerimiento #1

```
def artistasNacidosEnRango(catalogo,fechaInicio,fechaFin):
    artistas=lt.newList("ARRAY_LIST")
    llaves=mp.keySet(catalogo['fechaNacimiento'])
    for fecha in lt.iterator(llaves):
        if fechaFin>=int(fecha)>=fechaInicio:
            ids=me.getValue(mp.get(catalogo['fechaNacimiento'],fecha))
            for IDe in lt.iterator(ids):
                datosAutor=me.getValue(mp.get(catalogo['artistas'],IDe))
                lt.addLast(artistas,datosAutor)
    artistas=ms.sort(artistas,cmpArtistasBeginDate)
    return artistas
```

En este caso la complejidad temporal de la búsqueda seria O(N), ya que al no estar ordenado del map, se tendría que recorrer todo el map para hallar a los artistas que entran en el rango. Adicional a esto, la complejidad del merge sort seria de

- Requerimiento #2

```
def obrasPorDateAcquired(catalogo,fechaInicio,fechaFin):
    fechaInicio=int(fechaInicio.replace("-",""))
    fechaFin=int(fechaFin.replace("-",""))
    obras=lt.newList("ARRAY_LIST")
    llaves=mp.keySet(catalogo['fechaAdquisicion'])
    for fecha in lt.iterator(llaves):
        if fecha!="":
            fechaAdq=int(fecha.replace("-",""))
        else:
            fechaAdq=0
        if fechaInicio<=fechaAdq<=fechaFin:
            ids=me.getValue(mp.get(catalogo['fechaAdquisicion'],fecha))
            for obra in lt.iterator(ids):
                datosObra=me.getValue(mp.get(catalogo['obras'],obra))
                lt.addLast(obras,datosObra)
    obras=ms.sort(obras,cmpObrasDateAquired)
    return obras
```

Mateo Cote Canal m.cotec@uniandes.edu.co 202022609

Diego Acosta Corredor da.acostac1@uniandes.edu.co 202110516

En este caso la complejidad, al igual que en el requerimiento 1, seria O(N), ya que al no estar ordenado el map se tendría que recorrer todo para hallar las obras que entran al rango

- Requerimiento #3
- Requerimiento #4

```
def obrasNacionalidades(catalogo):
    nacionalidades=mp.newMap(maptype="CHAINING",loadfactor=8.0)
    keysNac=mp.keySet(catalogo['obras'])
    for idObra in lt.iterator(keysNac):
        obraIDS=me.getValue(mp.get(catalogo['obras'],idObra))['ConstituentID'].replace("[","").replace("]","").re
        obraNac=[]
        for x in obraIDS:
            nacio=me.getValue(mp.get(catalogo['artistas'],x))['Nationality']
            obraNac.append(nacio)
        for j in obraNac:
            if j=="" or j =="Nationality unknown":
                j="Nacionalidad desconocida"
            if not mp.contains(nacionalidades,j):
                nacionalidadesObras=lt.newList("ARRAY_LIST")
                lt.addLast(nacionalidadesObras,me.getValue(mp.get(catalogo['obras'],idObra)))
                mp.put(nacionalidades,j,nacionalidadesObras)
            else:
                nacioObras=me.getValue(mp.get(nacionalidades,j))
                lt.addLast(nacioObras,me.getValue(mp.get(catalogo['obras'],idObra)))
    natioKeys=mp.keySet(nacionalidades)
    natioNum=lt.newList("SINGLE_LINKED")
    for key in lt.iterator(natioKeys):
        num=lt.size(me.getValue(mp.get(nacionalidades,key)))
        tup=(key,num)
        lt.addLast(natioNum,tup)
    natioNum=ms.sort(natioNum,cmpNacionalidades)
    mas=lt.getElement(natioNum,1)[0]
    listaMas=me.getValue(mp.get(nacionalidades,mas))
    listaMas=sa.sort(listaMas,cmpObrasPorTitulo)
    return nacionalidades,natioNum
```

- Requerimiento #5

2. Pruebas de tiempos de ejecución:

- Requerimiento #1 (archivo large con diferentes rangos de fechas. Valores promedio tomados en milisegundos)

Rango	Tiempo (ms)
1940 - 1960	140.63
1900 - 1970	460.93
1850 - 1980	614.58
1800 - 1990	648.4375
1700 - 2000	687.5

Mateo Cote Canal m.cotec@uniandes.edu.co 202022609

Diego Acosta Corredor da.acostac1@uniandes.edu.co 202110516

- Requerimiento #2 (archivo large con diferentes rangos de fechas. Valores promedio tomados en milisegundos)

Rango	Tiempo (ms)
1980-06-06/1999-09-09	1421.87
1970-06-06/2000-11-09	2807.29
1944-06-06/1989-11-09	4898.43
1960-03-03/2010-03-03	7229.16
1950-01-01/2020-11-11	9968.75

- Requerimiento #3 (archivo large con diferentes artistas. Valores promedio tomados en milisegundos)

	Requerimiento 3
Nombre	
Nombre	
Nombre	
Nombre	
Nombre	

- Requerimiento #4 (varios archivos de prueba)

Archivo	Tiempo (ms)
10%	511.68
30%	1453.21
50%	2468.23
80%	3980.11
100%	5162.51

- Requerimiento #5 (Archivo Large con diferentes departamentos. Valores promedio de 5 datos.)

Departamento	Tiempo (ms)
Photography	1452.69
Drawings & Prints	3659.25
Architecture & Design	889.27
Painting & Sculpture	192.29
Media & Performance	151.79

Mateo Cote Canal m.cotec@uniandes.edu.co 202022609

Diego Acosta Corredor da.acostac1@uniandes.edu.co 202110516

3. Complejidad temporal Reto 1 vs Complejidad temporal Reto 2

En este reto la complejidad temporal fue menor a la del reto 1 ya que, aunque en algunos casos era necesario hacer una búsqueda con un recorrido total, pasando por todos los elementos, era más rápido ya que se accedía a ellos en cualquier momento en O(1), lo que no siempre era posible con un TAD lista. Por esto, al estar referenciados para acceder más fácil a ellos, con un map fue mas sencillo y eficiente la búsqueda y los requerimientos