ANALISIS DE COMPLEJIDAD RETO 2

Juliana Sofia Ahumada Arcos – 201921471

j.ahumadaa@uniandes.edu.co

Daniela Parra Martínez - 202013036

d.parram2@uniandes.edu.co

Requerimiento 1

```
def listar_artist_date(A_I, A_FN, catalog):
    artista = lt.newList('ARRAY_LIST')   O(1)
    fechas_tot = catalog['BeginDate']   O(1)
    fechas = mp.keySet(fechas_tot)   O(1)
    orden = ordenamiento_artist_AI(fechas)   O(1)
    for fecha in lt.iterator(orden):   O(n)
        if (fecha >= A_I) and (fecha <= A_FN):   O(1)
            entry = mp.get(fechas_tot, fecha)   O(1)
            valor = me.getValue(entry)   O(1)
            for artist in valor['elements']:   O(log n)
                lt.addLast(artista, artist)   O(1)

    total = lt.size(artista)   O(1)

    primeros = lt.subList(artista, 1, 3)   O(1)
    ultimos = lt.subList(artista, lt.size(artista) - 2, 3)   O(1)

    return total, primeros['elements'], ultimos['elements']
```

Complejidad total
O(n log n)

Tiempo de ejecución(msg):
- **62.5**

Memoria utilizada(GB):
- **3.44/3.88**

Requerimiento 2

```
def listar_artwork_date(F_I, F_FN, catalog):
    contador = 0
    obra = lt.newList('ARRAY_LIST') O(1)
    fechas_tot = catalog['DateAcquired'] O(1)
    fechas = mp.keySet(fechas_tot)   O(1)
    orden = ordenamiento_artist_AI(fechas)  O(1)
    for fecha in lt.iterator(orden): O(n)
        if (fecha >= F_I) and (fecha <= F_FN):  O(1)
            entry = mp.get(fechas_tot, fecha)   O(1)
            valor = me.getValue(entry)  O(1)
            for obr in valor['elements']: O( log n)
                lt.addLast(obra, obr)  O(1)

    total = lt.size(obra)   O(1)

    primeros = lt.subList(obra, 1, 3)   O(1)
    ultimos = lt.subList(obra, lt.size(obra) - 2, 3)   O(1)

    for obr in lt.iterator(obra):  O(n)
        if 'Purchase' in obr['CreditLine'] or 'purchase' in obr['CreditLine']:
            contador += 1

    return total, primeros['elements'], ultimos['elements'], contador
```

Complejidad total
O(n log n)

Tiempo de ejecución(msg):
- **1243.75**

Memoria utilizada(GB):
- **3,47/3.88**

Requerimiento 3: Daniela Parra Martínez

```
def clasificacion_medio_t_obra(Name, catalog):
    todos = lt.newList('ARRAY_LIST')                    O(1)
    medios = lt.newList('ARRAY_LIST')                   O(1)
    artistas = catalog['artista_obra']                  O(1)


    artista = mp.get(artistas, Name)                    O(1)
    obras_artista = me.getValue(artista)                O(1)
    for obras in lt.iterator(obras_artista):            O(n)
        medio = obras['Medium']                         O(1)
        tec_incluida= False                             O(1)
        for obra_i in lt.iterator(medios):              O(log n)
            if medio == obra_i:                         O(1)
                tec_incluida = True                     O(1)
        if not tec_incluida:                            O(1)
            lt.addLast(medios, medio)                   O(1)
                                # O(n logn)
```

```
contador_mayor = 0                                      O(1)
tecnica_mayor = ""                                      O(1)
for obra_medio in lt.iterator(medios):                  O(n)
    contador = 0                                        O(1)
    for obras in lt.iterator(obras_artista):            O(log n)
        medio = obras['Medium']                         O(1)
        if obra_medio == medio:                         O(1)
            contador += 1                               O(1)
    if contador > contador_mayor:                       O(1)
        contador_mayor = contador                       O(1)
        tecnica_mayor = obra_medio                      O(1)
    tupla = obra_medio, contador                        O(1)
    lt.addLast(todos,tupla)                             O(1)
t_medio=lt.size(todos)                                  O(1)
t_obras = lt.size(obras_artista)                        O(1)
lst_tecnicamayor= lt.newList('ARRAY_LIST')              O(1)
for obras in lt.iterator(obras_artista):                O(n)
    medio = obras['Medium']                             O(1)
    if tecnica_mayor == medio:                          O(1)
        lt.addLast(lst_tecnicamayor, obras)             O(1)
primeros_3 = lt.subList(lst_tecnicamayor, 1, 3)         O(1)
ultimas = lt.subList(lst_tecnicamayor, lt.size(lst_tecnicamayor) - 2, 3)  O(1)

return t_obras, t_medio, tecnica_mayor, lst_tecnicamayor, primeros_3['elements'], ultimas['elements']
```

Complejidad total
O(n log n)

Tiempo de ejecución(mseg):
- **1250.0**

Memoria utilizada(GB):
- **3.36/3.88**

Requerimiento 4: Juliana Sofia Ahumada Arcos

```
def Obras_Nacionalidad(catalog):
    cuantos = 0
    valores = lt.newList('ARRAY_LIST')  O(1)
    nacionalidades = catalog['artistNationality']  O(1)
    cada_una = mp.keySet(nacionalidades)  O(1)
    for nacionalidad in lt.iterator(cada_una):  O(n)
        if nacionalidad == '' or nacionalidad == 'Nationality unknown':  O(1)
            entry = mp.get(nacionalidades, nacionalidad)  O(1)
            valor = me.getValue(entry)['size']  O(1)
            cuantos += valor  O(1)
            dic = (cuantos, 'Nationality unknown')  O(1)
        else:
            entry = mp.get(nacionalidades, nacionalidad)  O(1)
            valor = me.getValue(entry)['size']  O(1)
            dic = (valor, nacionalidad)  O(1)

        lt.addLast(valores, dic)  O(1)

    orden = ordenamiento(valores)  O(1)
    mayores = lt.subList(orden, 1, 10)  O(1)

    primeros = mp.get(nacionalidades, lt.getElement(mayores, 1)[1])  O(1)
    primeros_3 = lt.subList(me.getValue(primeros), 1, 3)  O(1)
    ultimas = lt.subList(me.getValue(primeros), lt.size(me.getValue(primeros)) - 2, 3)
                                                                        O(1)
    return mayores['elements'], primeros_3['elements'], ultimas['elements']
```

Complejidad total O(n)

Tiempo de ejecución(msg):
- **15.625**

Memoria utilizada(GB):
- **3.35/3.88**

Requerimiento 5

```
def Costo_departamento(department, catalog):
    costo = 0 O(1)
    lst_fechas_o = lt.newList('ARRAY_LIST') O(1)
    lst_costo_i = lt.newList('ARRAY_LIST') O(1)
    departamentos = catalog['Department'] O(1)
    departamento = mp.get(departamentos, department) O(1)
    obras = me.getValue(departamento) O(1)
    total_obras = lt.size(obras) O(1)
    for obra in lt.iterator(obras): O(n)
        costo_i = 0 O(1)
        depth = obra['Depth (cm)'] O(1)
        diameter = obra['Diameter (cm)'] O(1)
        height = obra['Height (cm)'] O(1)
        length = obra['Length (cm)'] O(1)
        width = obra['Width (cm)'] O(1)
        date = obra['Date'] O(1)
        if date != '': O(1)
            tupla_date = date, obra O(1)
            lt.addLast(lst_fechas_o, tupla_date) O(1)

        x = Dimensiones(depth, diameter, height, length, width) O(1)
        if x == -1: O(1)
            costo += 48 O(1)
            costo_i = 48 O(1)
        else:
            costo_1 = x * 72 O(1)
            costo += costo_1 O(1)
            costo_i = x * 72 O(1)
        tupla = costo_i, obra O(1)
        lt.addLast(lst_costo_i, tupla) O(1)

    costo_total = round(costo, 3) O(1)

    orden_1 = ordenamientos(lst_costo_i) O(1)
    orden_2 = ordenamientos(lst_fechas_o) O(1)

    primeros = lt.subList(orden_2, 1, 5) O(1)
    ultimos = lt.subList(orden_1, lt.size(orden_1) - 4, 5) O(1)

    lista_f = Valores(primeros, ultimos) O(1)

    return total_obras, costo_total, lista_f[0]['elements'], lista_f[1]['elements']

def Valores(primeros, ultimos):
    primeros_5 = lt.newList('ARRAY_LIST') O(1)
    ultimos_5 = lt.newList('ARRAY_LIST') O(1)
    for valor in lt.iterator(primeros): O(log n)
        obra = valor[1] O(1)
        lt.addLast(primeros_5, obra) O(1)
    for value in lt.iterator(ultimos): O(log n)
        obras = value[1] O(1)
        lt.addLast(ultimos_5, obras) O(1)

    return primeros_5, ultimos_5
```

Requerimiento 5

```
def Dimensiones(depth, diameter, height, length, width):
    contador = 0  O(1)
    no_hay = True  O(1)
    dimensiones = -1  O(1)
    posicion = 1 O(1)
    medidas = lt.newList("ARRAY_LIST")   O(1)
    lt.addLast(medidas, depth)  O(1)
    lt.addLast(medidas, height) O(1)
    lt.addLast(medidas, length) O(1)
    lt.addLast(medidas, width) O(1)
    lt.addLast(medidas, diameter) O(1)
    while posicion <= lt.size(medidas):  O(n)
        dimension = lt.getElement(medidas, posicion)  O(1)
        if (dimension != '') and (dimension != '0'):  O(1)
            lt.changeInfo(medidas, posicion, float(dimension)) O(1)
            contador += 1   O(1)
            no_hay = False   O(1)
        else:
            lt.changeInfo(medidas, posicion, 1) O(1)
        posicion += 1   O(1)
    factor = 10**(-2*contador) O(1)
    if no_hay == False: O(1)
        if diameter != '':  O(1)
            diameter = lt.getElement(medidas, 5) O(1)
            height = lt.getElement(medidas, 2)  O(1)
            dimensiones = 3.1416 * ((diameter/2)**2) * height * factor/100
    else:                                                    O(1)
        depth = lt.getElement(medidas, 1) O(1)
        height = lt.getElement(medidas, 2) O(1)
        length = lt.getElement(medidas, 3)  O(1)
        width = lt.getElement(medidas, 4) O(1)
        dimensiones =  depth * height * length * width * factor
return dimensiones                                           O(1)
```

Complejidad total
O(n log n)

Tiempo de ejecución(msg):
- **15328.125**

Memoria utilizada(GB):
- **3.41/3.88**

REQUERIMIENTO 1

RETO 1

- La complejidad es de o(n)
- Y con el archivo large su tiempo de ejecución es 1187.5 msg

RETO 2

- La complejidad es O(n log n)
- Y con el archivo large su tiempo de ejecución es 1243.75

REQUERIMIENTO 2

RETO 1

- La complejidad es de o(n)
- Y con el archivo large su tiempo de ejecución es 12453.125 msg

RETO 2

- La complejidad es de o(n log n)
- Y con el archivo large su tiempo de ejecución es 1243.75 msg

REQUERIMIENTO 3

RETO 1

- La complejidad es de o(n)
- Y con el archivo large su tiempo de ejecución es 12453.125 msg

RETO 2

- La complejidad es de o(n log n)
- Y con el archivo large su tiempo de ejecución es 1187.5 msg

REQUERIMIENTO 4

RETO 1

- La complejidad es de o(n**4logn**2)
- Y con el archivo large su tiempo de ejecución supero el límite de 5 min

RETO 2

- La complejidad es de o(n)
- Y con el archivo large su tiempo de ejecución es 15.625 msg

REQUERIMIENTO 5

RETO 1

- No se implementó este requerimiento sin embargo calculamos que su complejidad la cual es aproximadamente

o(n**3 log n)

RETO 2

- La complejidad es de o(n log n)
- Y con el archivo large su tiempo de ejecución es 15328.125 msg