RETO #2

Juanita Gil Arango –j.gila2@uniandes.edu.co - 202111556

Gabriela Carvajal – g.carvajal@uniandes.edu.co - 202111058

Complejidad temporal:

Req 1:

```
111 def ListaCiudad(catalog, ciudad):
112     listaciudad= lt.newList()
113     if ciudad in lt.iterator(om.keySet(catalog['city'])):
114         for ciudad in lt.iterator(om.keySet(catalog['city'])):
115             info =lt.newList()
116             lt.addLast(info, ciudad['datetime'])
117             lt.addLast(info, ciudad['city'])
118             lt.addLast(info, ciudad['country'])
119             lt.addLast(info, ciudad['duration (seconds)'])
120             lt.addLast(info, ciudad['shape'])
121             lt.addLast(listaciudad, info)
122     return listaciudad
```

O(n), ya que, en el peor de los casos, se cumple el bloque if y se hace un recorrido con el for.

Req 2:

```
143 def duration(catalog, segmin, segmax):
144     tiempos= om.values(catalog['duration (seconds)'], segmin, segmax)
145     print(tiempos)
146     first= lt.subList(tiempos, 1, 3)
147     last=lt.subList(tiempos, -2, 3)
148     primeros=lt.newList()
149     ultimos=lt.newList()
150     for linea in first:
151         x= lt.newList()
152         lt.addLast(x,linea['datetime'])
153         lt.addLast(x,linea['city'])
154         lt.addLast(x,linea['country'])
155         lt.addLast(x,linea['duration (seconds)'])
156         lt.addLast(x,linea['shape'])
157         lt.addLast(primeros,x)
158     for linea in last:
159         x= lt.newList()
160         lt.addLast(x,linea['datetime'])
161         lt.addLast(x,linea['city'])
162         lt.addLast(x,linea['country'])
163         lt.addLast(x,linea['duration (seconds)'])
164         lt.addLast(x,linea['shape'])
165         lt.addLast(ultimos,x)
166     return (tiempos, primeros, ultimos)
```

2O(n), ya que, en el peor de los casos, se cumplen ambos for

Req 3:

```
def durationHrs_min(catalog, inferior, superior):
    llaves_rango = om.keys(catalog['duration (hours/min)'],inferior, superior)
    size = lt.size(llaves_rango)
    for duration in lt.iterator(llaves_rango):
        entry = om.get(catalog['duration (hours/min)'], duration)
        valor = me.getValue(entry)
        for linea in lt.iterator(valor):
            lst = lt.newList()
            info = lt.newList()
            lt.addLast(info, linea['datetime'])
            lt.addLast(info, linea['city'])
            lt.addLast(info, linea['country'])
            lt.addLast(info, linea['duration (seconds)'])
            lt.addLast(info, linea['shape'])
            lt.addLast(lst,info)
            listaOrdenada = sa.sort(lst, compareDurationH_M)
    return size, listaOrdenada
```

O(n)^2, ya que, en el peor de los casos, se cumplen ambos for, el de afuera y el de adentro

Req 4:

```
def avistRangoFechas(catalog, inferior, superior):
    llaves_rango = om.keys(catalog['datetime'],inferior, superior)
    size = lt.size(llaves_rango)
    for fecha in lt.iterator(llaves_rango):
        entry = om.get(catalog['datetime'], fecha)
        valor = me.getValue(entry)
        for linea in lt.iterator(valor):
            lst = lt.newList()
            info = lt.newList()
            lt.addLast(info, linea['datetime'])
            lt.addLast(info, linea['city'])
            lt.addLast(info, linea['country'])
            lt.addLast(info, linea['duration (seconds)'])
            lt.addLast(info, linea['shape'])
            lt.addLast(lst,info)
    listaOrdenada = sa.sort(lst, compareDates)
    return size, listaOrdenada
```

O(n)^2, ya que, en el peor de los casos, se cumplen ambos for, el de afuera y el de adentro

Req 5:

	2O(n), ya que, en el peor de los casos, se cumplen ambos for

Requerimientos individuales:

Req. 2: Gabriela Carvajal

Req. 3: Juanita Gil