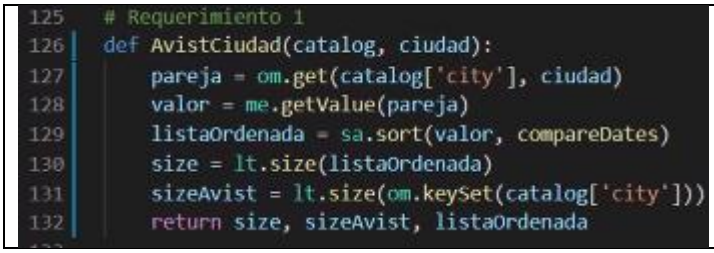
RETO #3

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

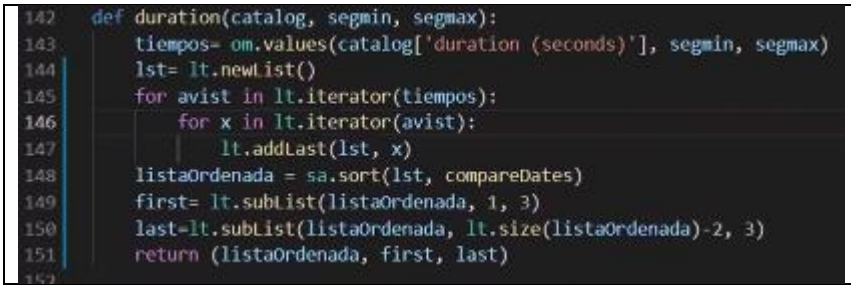
Gabriela Carvajal – g.carvajal@uniandes.edu.co - 202111058

Complejidad temporal:

Req 1:

		O(n(log(n)), ya que es la complejidad temporal del sort

Req 2:

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

Req 3:

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

Req 4:

```python
#Req 4
def avistRangoFechas(catalog, inferior, superior):
    inf = datetime.datetime.strptime(inferior,"%Y-%m-%d")
    inf2 = inf.date()
    sup = datetime.datetime.strptime(superior,"%Y-%m-%d")
    sup2 = sup.date()
    llaves_rango = om.keys(catalog['datetime'],inf2, sup2)
    lista = lt.newList()
    for fecha in lt.iterator(llaves_rango):
        entry = om.get(catalog['datetime'], fecha)
        valor = me.getValue(entry)
        for linea in lt.iterator(valor):
            lt.addLast(lista, linea)
    listaOrdenada = sa.sort(lista, compareDates)
    size = lt.size(listaOrdenada)
    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:

```python
#req 5
def avistZona(catalog, longmin, longmax, latmin, latmax):
    zonas= lt.newList()
    avist= om.values(catalog['longitude'], longmin, longmax)
    for avistamientos in lt.iterator(avist):
        for avistamiento in lt.iterator(avistamientos):
            if float(avistamiento['latitude']) >= latmin and float(avistamiento['latitude']) <= latmax:
                lt.addLast(zonas, avistamiento)
    total= lt.size(zonas)
    first= lt.subList(zonas, 1, 3)
    last=lt.subList(zonas, total-2, 3)
    return (total, first, last)
```

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

Requerimientos individuales:

Req. 2: Gabriela Carvajal

Req. 3: Juanita Gil