**Integrantes:**

Req 2 - Esteban Leiva, 202021368, e.leivam@uniandes.edu.co

Req 3 - Michelle Vargas, 201914771, [bm.vargas@uniandes.edu.co](mailto:bm.vargas@uniandes.edu.co)

**Análisis de complejidad, tiempo y memoria:**

*REQ 1*:

def req1(*menor*,*mayor*,*feature*,*catalog*):

    events = om.values(catalog[feature],menor,mayor) #1 O(lgN)

    num\_events = model.numevents(events) #2 O(N)

    lista\_artistas = model.list\_art(events) #3 O(N)

    mapa = mp.newMap(*maptype*="PROBING",*loadfactor*=0.5)

    i = it.newIterator(lista\_artistas)

*while* it.hasNext(i): #4 O(N)

        artist = it.next(i)

        mp.put(mapa, artist, None)

    artists = mp.size(mapa)

    print('\n'+feature+' is between '+str(menor)+' and '+str(mayor)+'\nTotal of reproduction: '+str(num\_events)+'\nTotal of unique artists: '+str(artists))

Conclusión:

Teniendo en cuenta lo anterior, la complejidad del Requerimiento 1 es O(N).

Tiempo y consumo de memoria:

|  |  |
| --- | --- |
| Tiempo (ms) | Memoria (kB) |
| 591.198 | 1454.500 |

*REQ 2*:

def req2(*catalog*,*min\_en*,*max\_en*,*min\_dan*,*max\_dan*):

    keys1 = om.values(catalog["energy"],min\_en,max\_en) #1 O(lgN)

    lista1 = model.listaeventos(catalog["energy"],keys1) #2 O(N)

    mapa = om.newMap(*omaptype*="RBT")

    i1 = it.newIterator(lista1)

*while* it.hasNext(i1): #3 O(N)

        event = it.next(i1)

        model.addtomap2(mapa, event,"danceability")

    keys2 = om.values(mapa,min\_dan,max\_dan) #4 O

    lista2 = model.listaeventos(mapa,keys2) #5 O

    mapafin = mp.newMap(*maptype*="PROBING",*loadfactor*=0.5)

    i2 = it.newIterator(lista2)

*while* it.hasNext(i2): #6 O(N)

        event = it.next(i2)

        mp.put(mapafin, event["track\_id"],event)

    print('\Energy is between '+str(min\_en)+' and '+str(max\_en)+'\Danceability is between '+str(min\_dan)+' and '+str(max\_dan)+'\nTotal of unique tracks in events: '+str(mp.size(mapafin))+'\n')

    n = 1

    listakeys = mp.keySet(mapafin)

*while* n<=5: #7 O(1)

        llave = lt.getElement(listakeys, n)

        event = mp.get(mapafin,llave)["value"]

        print('Track '+str(n)+': '+ event["track\_id"]+' with energy of '+str(event["energy"])+' and danceability of '+str(event["danceability"]))

        n +=1

    print('\n')

Conclusión:

Teniendo en cuenta lo anterior, la complejidad del Requerimiento 2 es O(N).

Tiempo y consumo de memoria:

|  |  |
| --- | --- |
| Tiempo (ms) | Memoria (kB) |
| 411.847 | 3016.613 |

*REQ 3*:

*REQ 4*:

def req4(*catalog*,*genre*,*minimo*,*maximo*,*req*,*mapa*):

*if* req == "req5":

        mapa = mapa

*else*:

        mapa = catalog["tempo"]

    menor = None

    mayor = None

*if* minimo==None and maximo==None:

*if* genre=='reggae':

            menor=60.0

            mayor=90.0

*elif* genre=='down-tempo':

            menor=70.0

            mayor=100.0

*elif* genre=='chill-out':

            menor=90.0

            mayor=120.0

*elif* genre=='hip-hop':

            menor=85.0

            mayor=115.0

*elif* genre=='jazz and funk':

            menor=120.0

            mayor=125.0

*elif* genre=='pop':

            menor=100.0

            mayor=130.0

*elif* genre=='r&b':

            menor=60.0

            mayor=80.0

*elif* genre=='rock':

            menor=110.0

            mayor=140.0

*elif* genre=='metal':

            menor=100.0

            mayor=160.0

*else*:

        menor=minimo

        mayor=maximo

    events = om.values(mapa,menor,mayor) #1 O(lgN)

    num\_events = model.numevents(events) #2 O(N)

    lista\_artistas = model.list\_art(events) #3 O(N)

    mapa = mp.newMap(*maptype*="PROBING",*loadfactor*=0.5)

    i = it.newIterator(lista\_artistas)

*while* it.hasNext(i): #4 O(N)

        artist = it.next(i)

        mp.put(mapa, artist, None)

    artists = mp.size(mapa)

    listartists=mp.keySet(mapa) #5 O(N)

    print('\n======= '+genre.upper()+' ========'+'\nFor '+genre+' the tempo is between '+str(menor)+' and '+str(mayor)+'\n'+genre+' reproductions: '+str(num\_events)+' with '+str(artists)+' different artists'+'\n\n---- Some artists for '+genre+' -----\n')

    i=it.newIterator(listartists)

    n=1

*while* it.hasNext(i) and n<=5: #6 O(1)

        artist=it.next(i)

        print('Artist '+str(n)+': '+artist)

        n+=1

Conclusión:

Teniendo en cuenta lo anterior, la complejidad del Requerimiento 3 es O(N).

Tiempo y consumo de memoria:

|  |  |
| --- | --- |
| Tiempo (ms) | Memoria (kB) |
| 14304.934 | 48.750 |