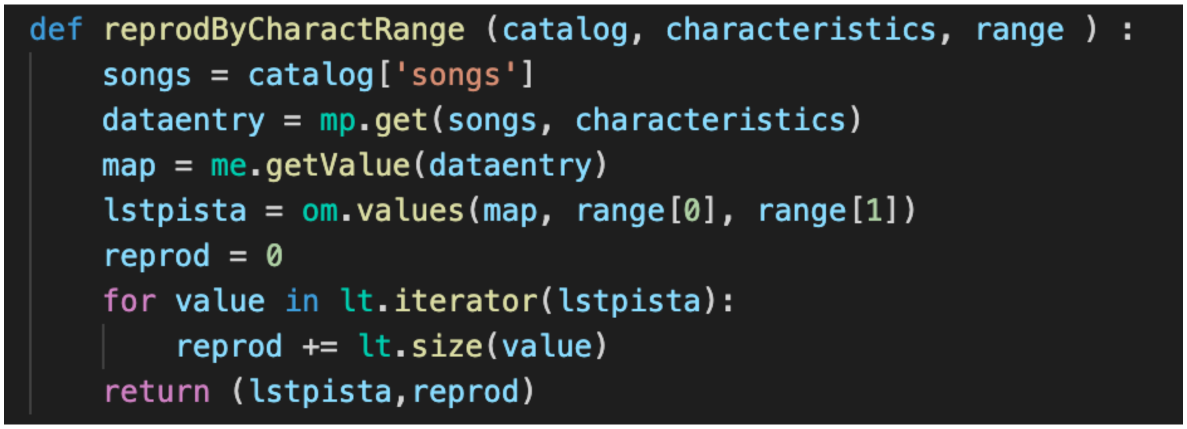
Santiago Pardo
Juan Ramírez

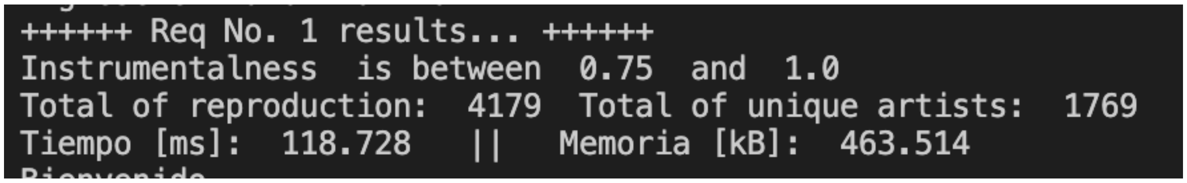
RETO 3

ANÁLISIS

REQ 1



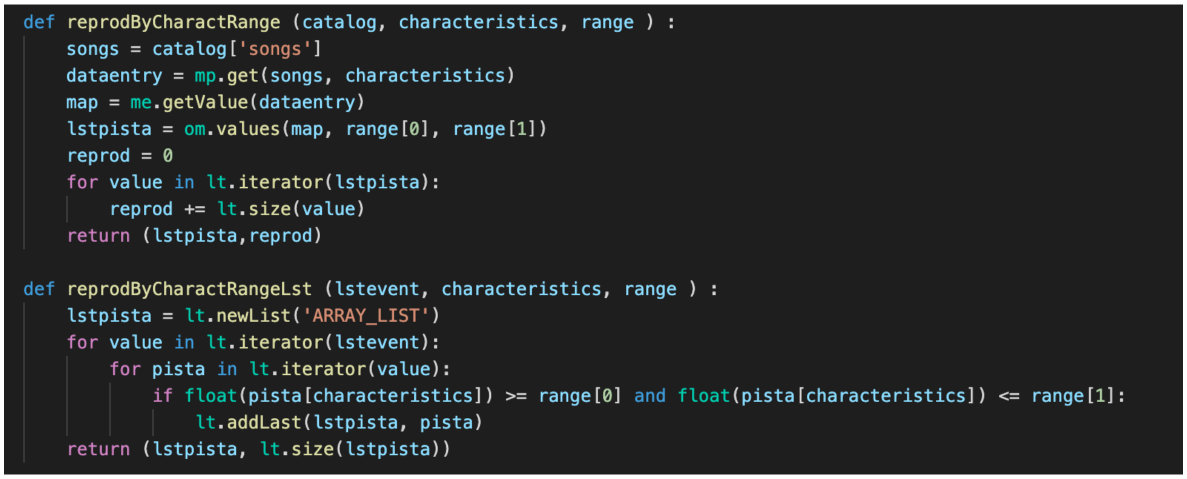
```python
def reprodByCharactRange (catalog, characteristics, range ) :
    songs = catalog['songs']
    dataentry = mp.get(songs, characteristics)
    map = me.getValue(dataentry)
    lstpista = om.values(map, range[0], range[1])
    reprod = 0
    for value in lt.iterator(lstpista):
        reprod += lt.size(value)
    return (lstpista,reprod)
```



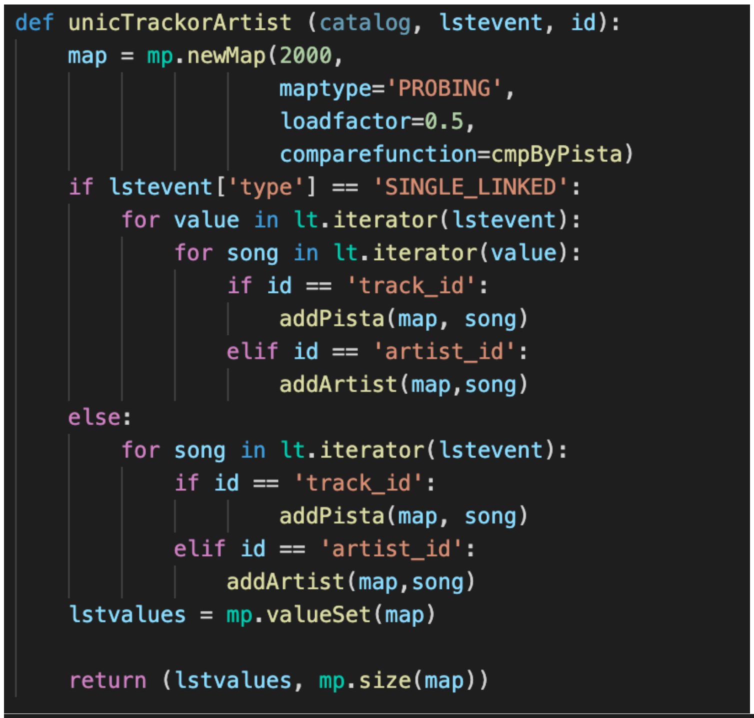
```
++++++ Req No. 1 results... ++++++
Instrumentalness  is between  0.75  and  1.0
Total of reproduction:  4179  Total of unique artists:  1769
Tiempo [ms]:  118.728   ||   Memoria [kB]:  463.514
```

COMPLEJIDAD: O(N), se consume poca memoria, un tiempo relativamente bajo

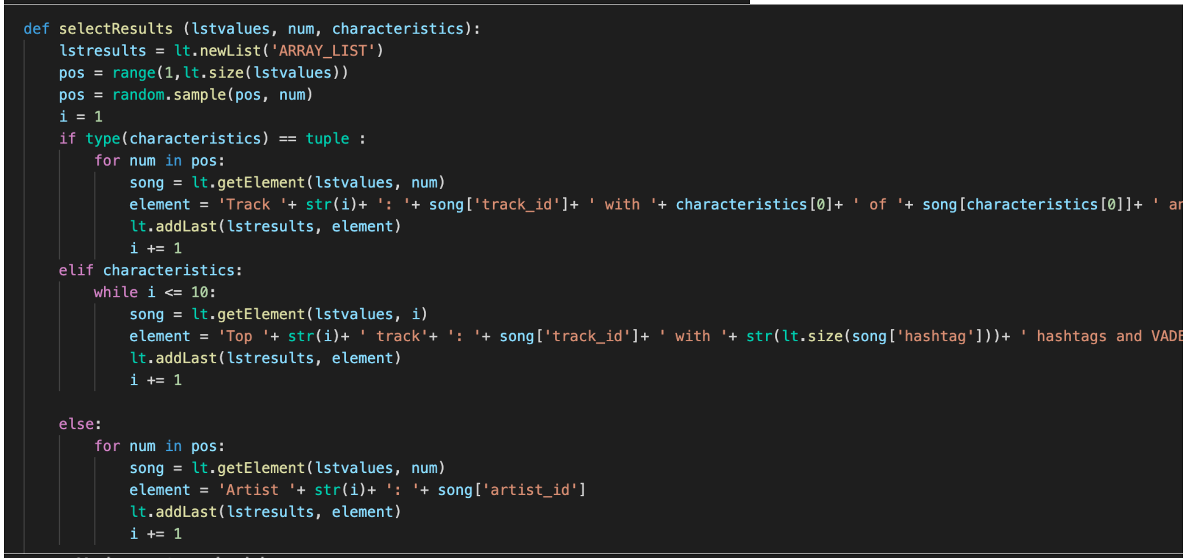
REQ 2 – SANTIAGO PARDO



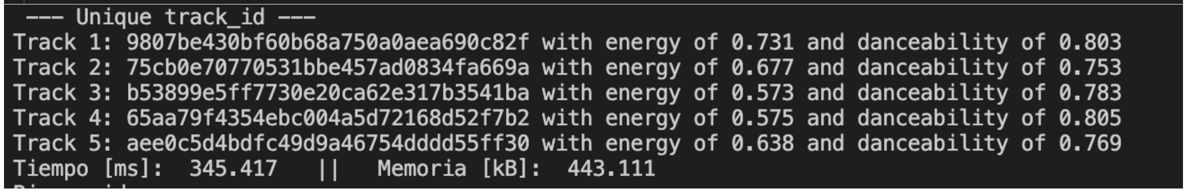
```python
def reprodByCharactRange (catalog, characteristics, range ) :
    songs = catalog['songs']
    dataentry = mp.get(songs, characteristics)
    map = me.getValue(dataentry)
    lstpista = om.values(map, range[0], range[1])
    reprod = 0
    for value in lt.iterator(lstpista):
        reprod += lt.size(value)
    return (lstpista,reprod)

def reprodByCharactRangeLst (lstevent, characteristics, range ) :
    lstpista = lt.newList('ARRAY_LIST')
    for value in lt.iterator(lstevent):
        for pista in lt.iterator(value):
            if float(pista[characteristics]) >= range[0] and float(pista[characteristics]) <= range[1]:
                lt.addLast(lstpista, pista)
    return (lstpista, lt.size(lstpista))
```



```
def unicTrackorArtist (catalog, lstevent, id):
    map = mp.newMap(2000,
                    maptype='PROBING',
                    loadfactor=0.5,
                    comparefunction=cmpByPista)
    if lstevent['type'] == 'SINGLE_LINKED':
        for value in lt.iterator(lstevent):
            for song in lt.iterator(value):
                if id == 'track_id':
                    addPista(map, song)
                elif id == 'artist_id':
                    addArtist(map,song)
    else:
        for song in lt.iterator(lstevent):
            if id == 'track_id':
                addPista(map, song)
            elif id == 'artist_id':
                addArtist(map,song)
    lstvalues = mp.valueSet(map)

    return (lstvalues, mp.size(map))
```



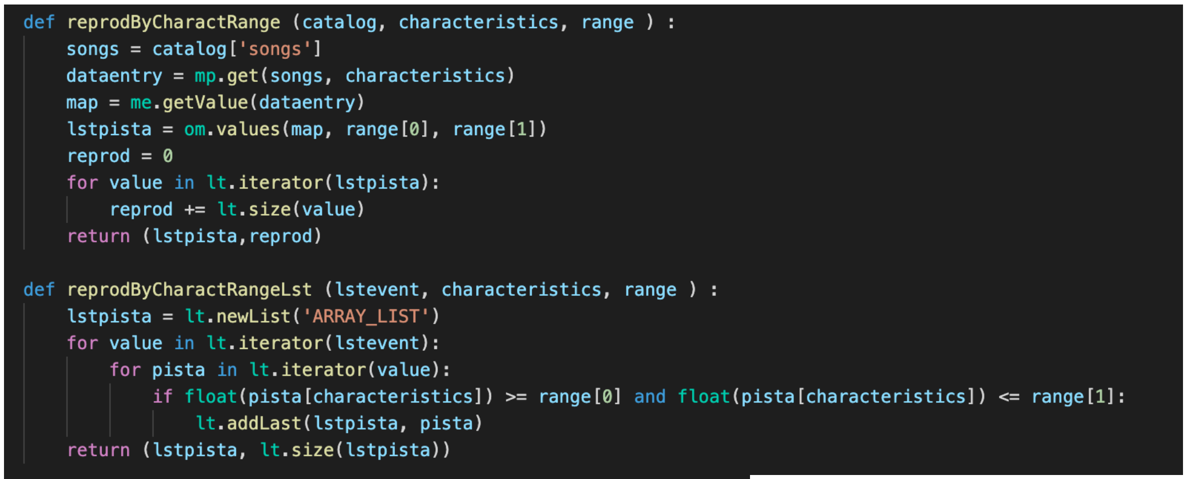
```
def selectResults (lstvalues, num, characteristics):
    lstresults = lt.newList('ARRAY_LIST')
    pos = range(1,lt.size(lstvalues))
    pos = random.sample(pos, num)
    i = 1
    if type(characteristics) == tuple :
        for num in pos:
            song = lt.getElement(lstvalues, num)
            element = 'Track '+ str(i)+ ': '+ song['track_id']+ ' with '+ characteristics[0]+ ' of '+ song[characteristics[0]]+ ' ar
            lt.addLast(lstresults, element)
            i += 1
    elif characteristics:
        while i <= 10:
            song = lt.getElement(lstvalues, i)
            element = 'Top '+ str(i)+ ' track'+ ': '+ song['track_id']+ ' with '+ str(lt.size(song['hashtag']))+ ' hashtags and VADE
            lt.addLast(lstresults, element)
            i += 1

    else:
        for num in pos:
            song = lt.getElement(lstvalues, num)
            element = 'Artist '+ str(i)+ ': '+ song['artist_id']
            lt.addLast(lstresults, element)
            i += 1
```



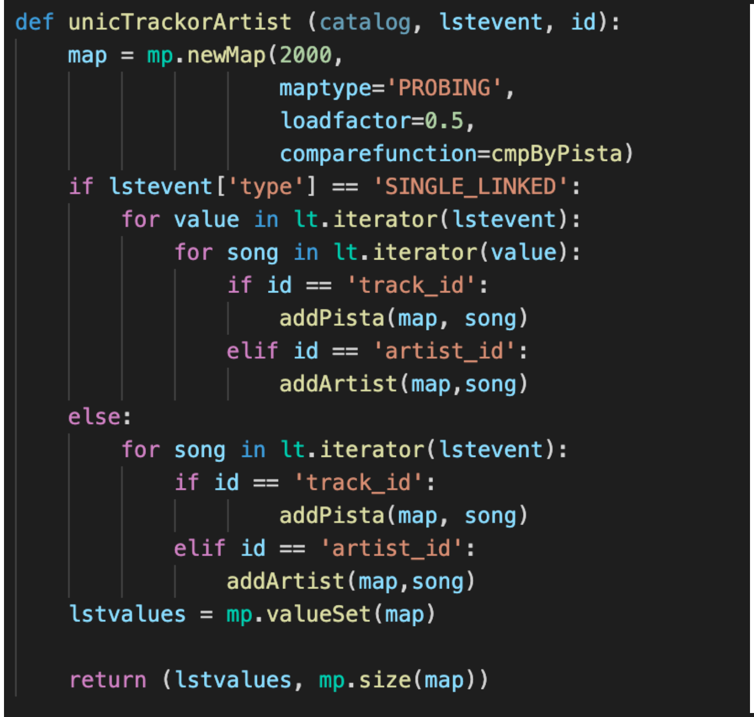
```
 --- Unique track_id ---
Track 1: 9807be430bf60b68a750a0aea690c82f with energy of 0.731 and danceability of 0.803
Track 2: 75cb0e70770531bbe457ad0834fa669a with energy of 0.677 and danceability of 0.753
Track 3: b53899e5ff7730e20ca62e317b3541ba with energy of 0.573 and danceability of 0.783
Track 4: 65aa79f4354ebc004a5d72168d52f7b2 with energy of 0.575 and danceability of 0.805
Track 5: aee0c5d4bdfc49d9a46754dddd55ff30 with energy of 0.638 and danceability of 0.769
Tiempo [ms]:  345.417   ||   Memoria [kB]:  443.111
```

COMPLEJIDAD: O(N^2), se consume memoria similar al requerimiento 1, y es un tiempo mayor.

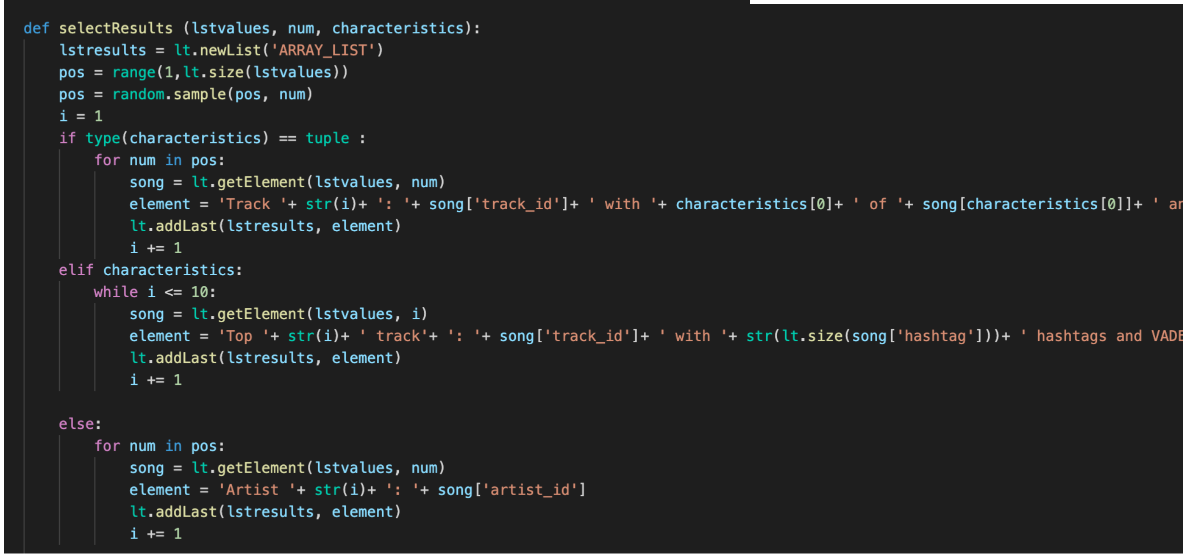
REQ 3 – JUAN RAMÍREZ



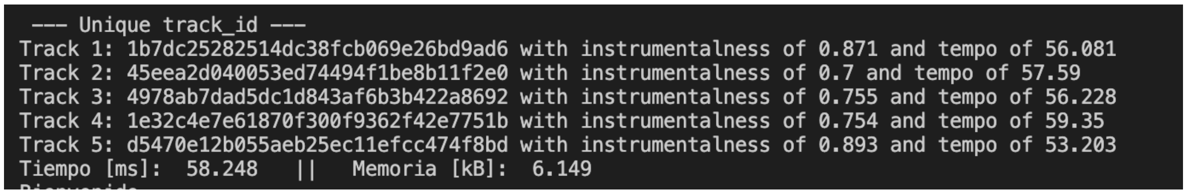
```python
def reprodByCharactRange (catalog, characteristics, range ) :
    songs = catalog['songs']
    dataentry = mp.get(songs, characteristics)
    map = me.getValue(dataentry)
    lstpista = om.values(map, range[0], range[1])
    reprod = 0
    for value in lt.iterator(lstpista):
        reprod += lt.size(value)
    return (lstpista,reprod)

def reprodByCharactRangeLst (lstevent, characteristics, range ) :
    lstpista = lt.newList('ARRAY_LIST')
    for value in lt.iterator(lstevent):
        for pista in lt.iterator(value):
            if float(pista[characteristics]) >= range[0] and float(pista[characteristics]) <= range[1]:
                lt.addLast(lstpista, pista)
    return (lstpista, lt.size(lstpista))
```



```python
def unicTrackorArtist (catalog, lstevent, id):
    map = mp.newMap(2000,
                    maptype='PROBING',
                    loadfactor=0.5,
                    comparefunction=cmpByPista)
    if lstevent['type'] == 'SINGLE_LINKED':
        for value in lt.iterator(lstevent):
            for song in lt.iterator(value):
                if id == 'track_id':
                    addPista(map, song)
                elif id == 'artist_id':
                    addArtist(map,song)
    else:
        for song in lt.iterator(lstevent):
            if id == 'track_id':
                addPista(map, song)
            elif id == 'artist_id':
                addArtist(map,song)
    lstvalues = mp.valueSet(map)

    return (lstvalues, mp.size(map))
```



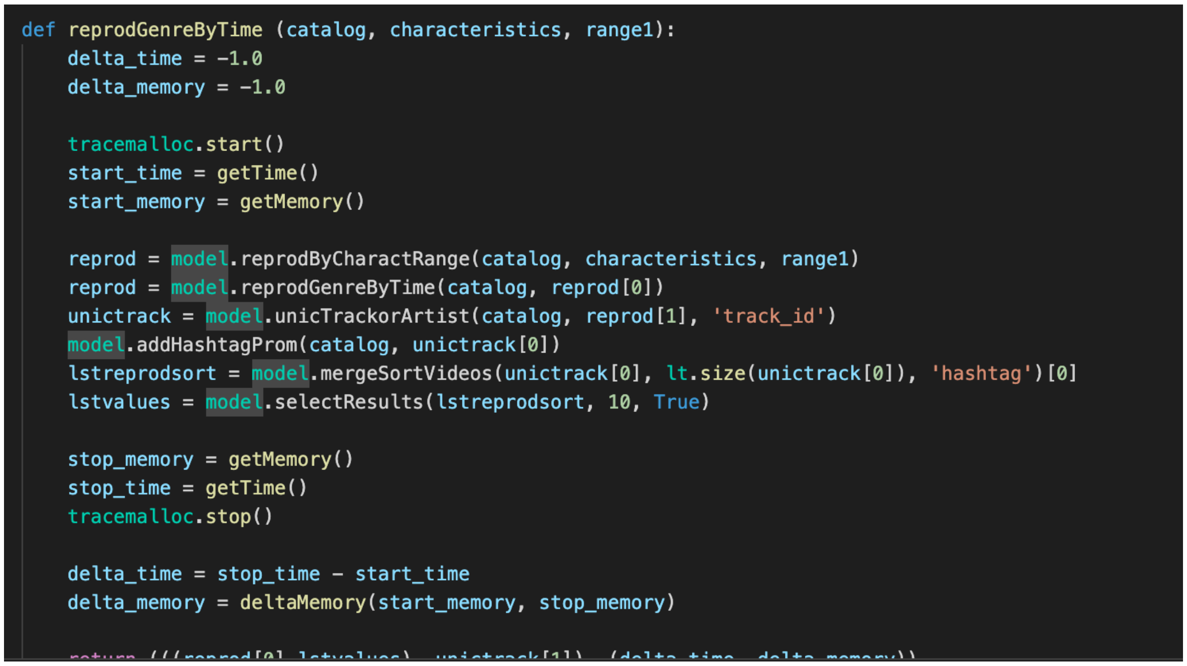
```python
def selectResults (lstvalues, num, characteristics):
    lstresults = lt.newList('ARRAY_LIST')
    pos = range(1,lt.size(lstvalues))
    pos = random.sample(pos, num)
    i = 1
    if type(characteristics) == tuple :
        for num in pos:
            song = lt.getElement(lstvalues, num)
            element = 'Track '+ str(i)+ ': '+ song['track_id']+ ' with '+ characteristics[0]+ ' of '+ song[characteristics[0]]+ ' ar
            lt.addLast(lstresults, element)
            i += 1
    elif characteristics:
        while i <= 10:
            song = lt.getElement(lstvalues, i)
            element = 'Top '+ str(i)+ ' track'+ ': '+ song['track_id']+ ' with '+ str(lt.size(song['hashtag']))+ ' hashtags and VADE
            lt.addLast(lstresults, element)
            i += 1

    else:
        for num in pos:
            song = lt.getElement(lstvalues, num)
            element = 'Artist '+ str(i)+ ': '+ song['artist_id']
            lt.addLast(lstresults, element)
            i += 1
```



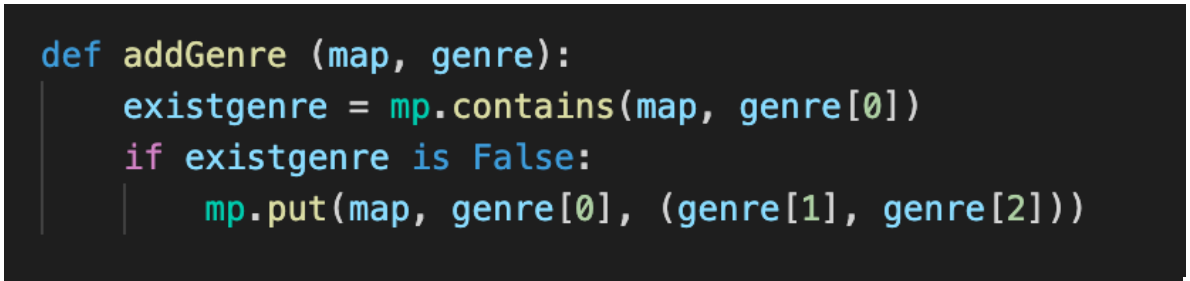
```
 --- Unique track_id ---
Track 1: 1b7dc25282514dc38fcb069e26bd9ad6 with instrumentalness of 0.871 and tempo of 56.081
Track 2: 45eea2d040053ed74494f1be8b11f2e0 with instrumentalness of 0.7 and tempo of 57.59
Track 3: 4978ab7dad5dc1d843af6b3b422a8692 with instrumentalness of 0.755 and tempo of 56.228
Track 4: 1e32c4e7e61870f300f9362f42e7751b with instrumentalness of 0.754 and tempo of 59.35
Track 5: d5470e12b055aeb25ec11efcc474f8bd with instrumentalness of 0.893 and tempo of 53.203
Tiempo [ms]:  58.248   ||   Memoria [kB]:  6.149
```

COMPLEJIDAD: O(N^2), similar al requerimiento 2

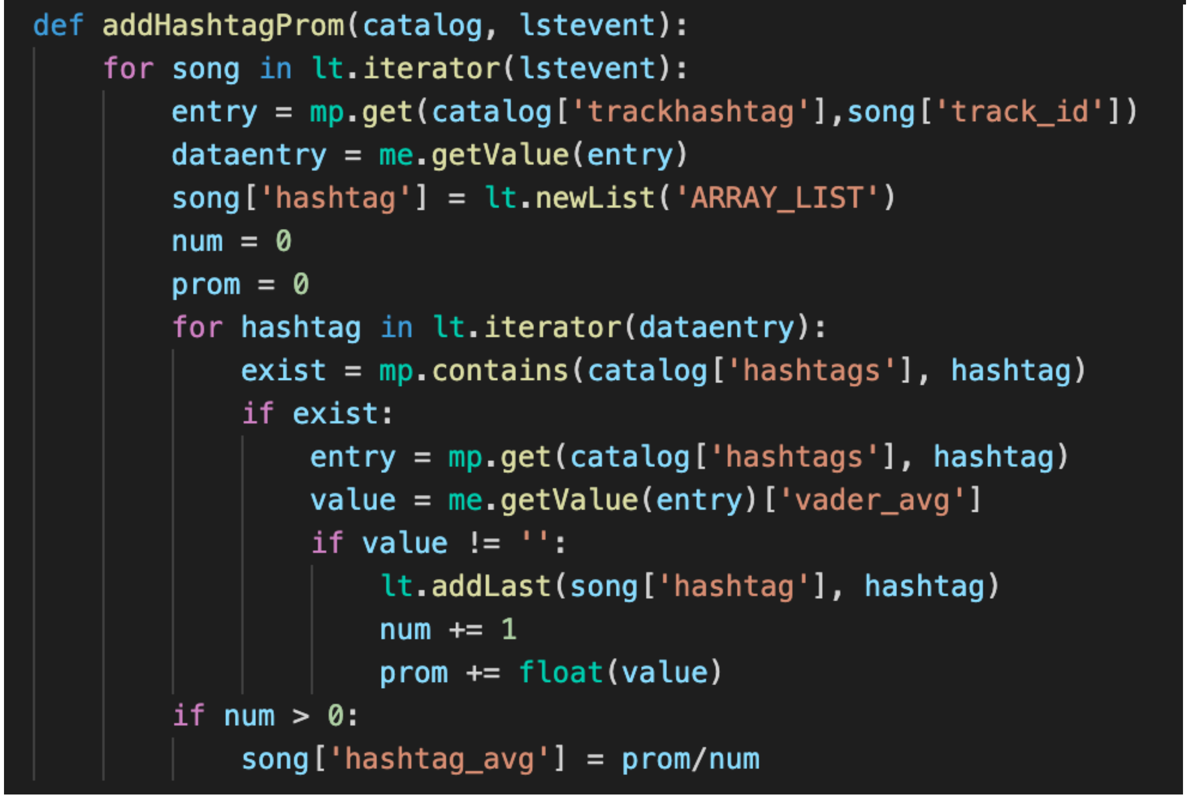
REQ 4



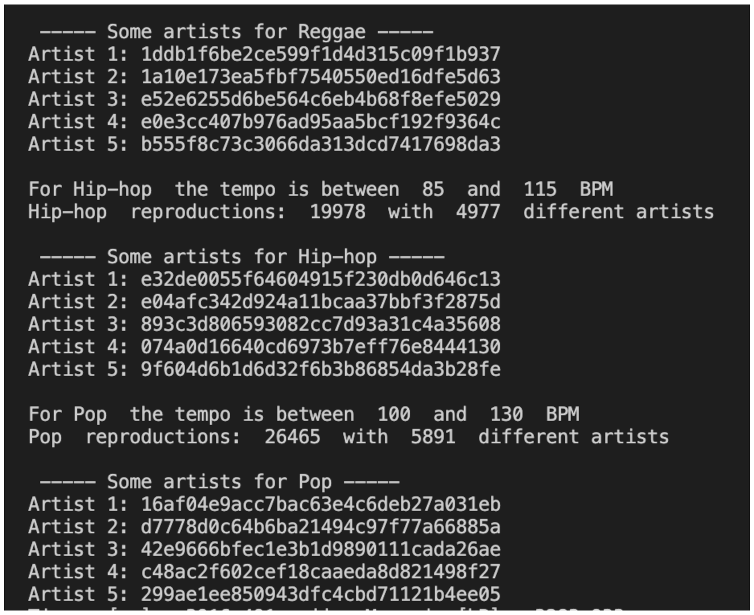
```python
def reprodGenreByTime (catalog, characteristics, range1):
    delta_time = -1.0
    delta_memory = -1.0

    tracemalloc.start()
    start_time = getTime()
    start_memory = getMemory()

    reprod = model.reprodByCharactRange(catalog, characteristics, range1)
    reprod = model.reprodGenreByTime(catalog, reprod[0])
    unictrack = model.unicTrackorArtist(catalog, reprod[1], 'track_id')
    model.addHashtagProm(catalog, unictrack[0])
    lstreprodsort = model.mergeSortVideos(unictrack[0], lt.size(unictrack[0]), 'hashtag')[0]
    lstvalues = model.selectResults(lstreprodsort, 10, True)

    stop_memory = getMemory()
    stop_time = getTime()
    tracemalloc.stop()

    delta_time = stop_time - start_time
    delta_memory = deltaMemory(start_memory, stop_memory)

    return (((reprod[0], lstvalues), unictrack[1]), (delta_time, delta_memory))
```



```python
def addGenre (map, genre):
    existgenre = mp.contains(map, genre[0])
    if existgenre is False:
        mp.put(map, genre[0], (genre[1], genre[2]))
```



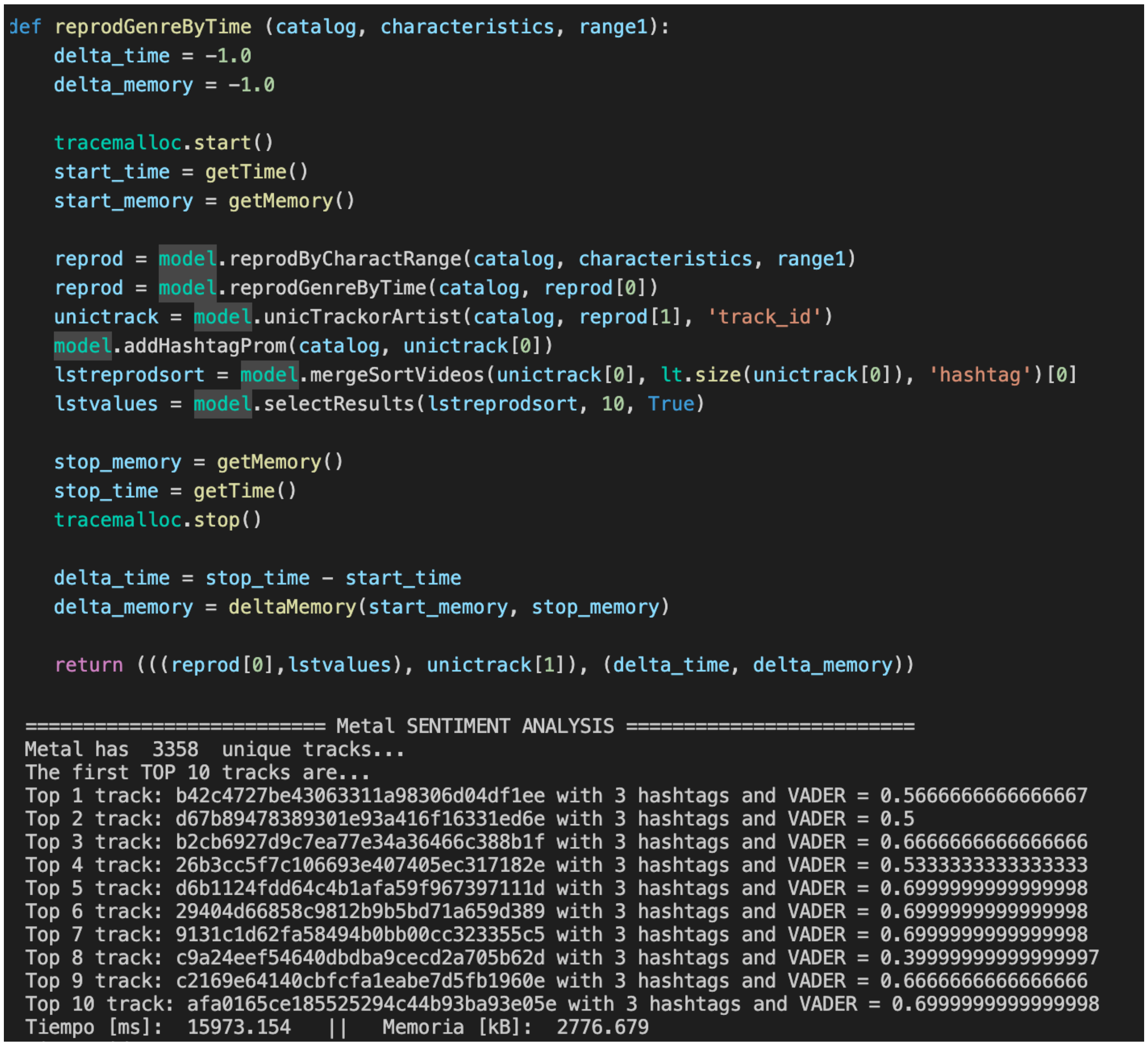
```python
def addHashtagProm(catalog, lstevent):
    for song in lt.iterator(lstevent):
        entry = mp.get(catalog['trackhashtag'],song['track_id'])
        dataentry = me.getValue(entry)
        song['hashtag'] = lt.newList('ARRAY_LIST')
        num = 0
        prom = 0
        for hashtag in lt.iterator(dataentry):
            exist = mp.contains(catalog['hashtags'], hashtag)
            if exist:
                entry = mp.get(catalog['hashtags'], hashtag)
                value = me.getValue(entry)['vader_avg']
                if value != '':
                    lt.addLast(song['hashtag'], hashtag)
                    num += 1
                    prom += float(value)
        if num > 0:
            song['hashtag_avg'] = prom/num
```



```
 ----- Some artists for Reggae -----
Artist 1: 1ddb1f6be2ce599f1d4d315c09f1b937
Artist 2: 1a10e173ea5fbf7540550ed16dfe5d63
Artist 3: e52e6255d6be564c6eb4b68f8efe5029
Artist 4: e0e3cc407b976ad95aa5bcf192f9364c
Artist 5: b555f8c73c3066da313dcd7417698da3

For Hip-hop  the tempo is between  85  and  115  BPM
Hip-hop  reproductions:  19978  with  4977  different artists

 ----- Some artists for Hip-hop -----
Artist 1: e32de0055f64604915f230db0d646c13
Artist 2: e04afc342d924a11bcaa37bbf3f2875d
Artist 3: 893c3d806593082cc7d93a31c4a35608
Artist 4: 074a0d16640cd6973b7eff76e8444130
Artist 5: 9f604d6b1d6d32f6b3b86854da3b28fe

For Pop  the tempo is between  100  and  130  BPM
Pop  reproductions:  26465  with  5891  different artists

 ----- Some artists for Pop -----
Artist 1: 16af04e9acc7bac63e4c6deb27a031eb
Artist 2: d7778d0c64b6ba21494c97f77a66885a
Artist 3: 42e9666bfec1e3b1d9890111cada26ae
Artist 4: c48ac2f602cef18caaeda8d821498f27
Artist 5: 299ae1ee850943dfc4cbd71121b4ee05
```

COMPLEJIDAD: O(N^2), en este caso, el mergesort tiene una menor complejidad que N^2, por lo que se ignora

REQ 5



```
def reprodGenreByTime (catalog, characteristics, range1):
    delta_time = -1.0
    delta_memory = -1.0

    tracemalloc.start()
    start_time = getTime()
    start_memory = getMemory()

    reprod = model.reprodByCharactRange(catalog, characteristics, range1)
    reprod = model.reprodGenreByTime(catalog, reprod[0])
    unictrack = model.unicTrackorArtist(catalog, reprod[1], 'track_id')
    model.addHashtagProm(catalog, unictrack[0])
    lstreprodsort = model.mergeSortVideos(unictrack[0], lt.size(unictrack[0]), 'hashtag')[0]
    lstvalues = model.selectResults(lstreprodsort, 10, True)

    stop_memory = getMemory()
    stop_time = getTime()
    tracemalloc.stop()

    delta_time = stop_time - start_time
    delta_memory = deltaMemory(start_memory, stop_memory)

    return (((reprod[0],lstvalues), unictrack[1]), (delta_time, delta_memory))
```

```
========================== Metal SENTIMENT ANALYSIS =========================
Metal has  3358  unique tracks...
The first TOP 10 tracks are...
Top 1 track: b42c4727be43063311a98306d04df1ee with 3 hashtags and VADER = 0.5666666666666667
Top 2 track: d67b89478389301e93a416f16331ed6e with 3 hashtags and VADER = 0.5
Top 3 track: b2cb6927d9c7ea77e34a36466c388b1f with 3 hashtags and VADER = 0.6666666666666666
Top 4 track: 26b3cc5f7c106693e407405ec317182e with 3 hashtags and VADER = 0.5333333333333333
Top 5 track: d6b1124fdd64c4b1afa59f967397111d with 3 hashtags and VADER = 0.6999999999999998
Top 6 track: 29404d66858c9812b9b5bd71a659d389 with 3 hashtags and VADER = 0.6999999999999998
Top 7 track: 9131c1d62fa58494b0bb00cc323355c5 with 3 hashtags and VADER = 0.6999999999999998
Top 8 track: c9a24eef54640dbdba9cecd2a705b62d with 3 hashtags and VADER = 0.39999999999999997
Top 9 track: c2169e64140cbfcfa1eabe7d5fb1960e with 3 hashtags and VADER = 0.6666666666666666
Top 10 track: afa0165ce185525294c44b93ba93e05e with 3 hashtags and VADER = 0.6999999999999998
Tiempo [ms]:  15973.154   ||   Memoria [kB]:  2776.679
```

COMPLEJIDAD: O(N^2), en este caso, el mergesort tiene una menor complejidad que N^2, por lo que se ignora

SE USARON LOS DATOS SMALL, 5PT Y 10PT