**1**

**import** requests**,** json**,** urllib.parse

base\_url **=** 'https://itunes.apple.com/search'

artist **=** 'Слот'

params **=** {

'term': artist,

'limit': 200,

'media': 'music',

'entity': 'song',

'attribute': 'artistTerm'

}

response **=** requests**.**get(f"{base\_url}?{urllib**.**parse**.**urlencode(params)}")

data **=** json**.**loads(response**.**text)

data **=** data['results']

2

data **=** map(**lambda** x: {

'artistName': x['artistName'],

'kind' : x['kind'],

'artistId': x['artistId'],

'collectionId': x['collectionId'],

'trackId': x['trackId'],

'artistName': x['artistName'],

'collectionName': x['collectionName'],

'trackName': x['trackName'],

'trackPrice': x['trackPrice'],

'releaseDate': x['releaseDate'],

'trackCount': x['trackCount'],

'trackTimeMillis': x['trackTimeMillis'],

'primaryGenreName': x['primaryGenreName']

}, data)

data **=** list(data)

3

**def** convert\_millis(millis: int) **->** str:

total\_seconds **=** millis **//** 1000

seconds **=** total\_seconds **%** 60

minutes **=** (total\_seconds **//** 60) **%** 60

hours **=** (total\_seconds **//** (60 **\*** 60))

**return** f"{hours}:Hours,{minutes}:Minutes,{seconds}:Seconds"

**def** filter\_list(items, value):

has\_more **=** items**.**count(value) **>** 1

**return** [x **for** x **in** items **if** **not** (x **==** value **and** has\_more)]

4

**import** statistics

countOfSongs **=** len(list(map(**lambda** x: x['trackName'], data)))

dataGenre **=** (list((map(**lambda** x: x['primaryGenreName'], data))))

dataAlbums **=** len(list(set(map(**lambda** x: x['collectionId'], data))))

songsDuration **=** sum(list(map(**lambda** x : x['trackTimeMillis'],data)))

Average\_time **=** statistics**.**mean(list(map(**lambda** x : x['trackTimeMillis'],data)))

median\_time **=** list(map(**lambda** x : x['trackTimeMillis'],data))

median\_time**.**sort()

median\_time **=** statistics**.**median(median\_time)

songsCost **=** sum(filter\_list(list(map(**lambda** x: x['trackPrice'], data)),**-**1))

Average\_cost **=** statistics**.**mean(filter\_list(list(map(**lambda** x: x['trackPrice'], data)),**-**1))

median\_cost **=** filter\_list(list(map(**lambda** x: x['trackPrice'], data)),**-**1)

median\_cost**.**sort()

median\_cost **=** statistics**.**median(median\_cost)

time\_extractor **=** **lambda** x: x['trackTimeMillis']

longest\_song **=** max(data, key**=**time\_extractor)

shortest\_song **=** min(data, key**=**time\_extractor)

track\_price\_extractor **=** **lambda** x: x['trackPrice']

dearest\_song **=** max(data, key**=**track\_price\_extractor)

сheapest\_song **=** min(data, key**=**track\_price\_extractor)

5

print('Artist:',artist)

print('Number of songs:',countOfSongs)

print('Popular genre:','Rock -',dataGenre**.**count('Pop'))

print('Album count:',dataAlbums)

print('Total duration of songs:',convert\_millis(songsDuration))

print('Average duration of songs:',convert\_millis(Average\_time))

print('Median duration of songs:',convert\_millis(median\_time))

print('Longest song:', longest\_song['trackName'], convert\_millis(longest\_song['trackTimeMillis']))

print('Shortest song:', shortest\_song['trackName'], convert\_millis(shortest\_song['trackTimeMillis']))

print('Total cost of songs ≈ ',round(songsCost),'$')

print('Average cost of songs:',Average\_cost,'$')

print('Median cost of songs:',median\_cost,'$')

print('The most expensive song:', dearest\_song['trackName'],'Trackrpice:',dearest\_song['trackPrice'])

**if** сheapest\_song['trackPrice'] **==** **-**1 :

print('Сheapest song:', сheapest\_song['trackName'],'Track price:','Free')

**else**:

print('Сheapest song:', сheapest\_song['trackName'],'Track price:',сheapest\_song['trackPrice'])

Artist: Слот

Count of songs: 68

Most popular genre count: Rock - 53

Count of albums: 15

Songs Duration: 5:Hours,28:Minutes,49:Seconds

Average time: 0.0:Hours,4.0:Minutes,3.0:Seconds

Median duration songs: 0.0:Hours,4.0:Minutes,5.0:Seconds

Longest song: Круги на воде 0:Hours,4:Minutes,41:Seconds

Shortest song: Мёртвые звезды (feat. Info) 0:Hours,:Minutes,0:Seconds

Total cost of songs ≈ 76 $

Average cost: 1.6 $

Median cost songs: 1.6 $

The most expensive song: Лего Trackrpice: 1.29

Сheapest song: Лего Track price: 1.29

6

**from** itertools **import** groupby

collection\_id\_extractor **=** **lambda** x: x['collectionId']

sorted\_by\_album **=** sorted(data, key**=**collection\_id\_extractor)

album\_groups **=** {}

album\_names **=** {}

**for** album\_id, songs **in** groupby(sorted\_by\_album, key**=**collection\_id\_extractor):

songs **=** list(songs)

album\_groups[album\_id] **=** songs

album\_names[album\_id] **=** songs[0]['collectionName']

**del** sorted\_by\_album

print(list(album\_names**.**values()))

print(list(album\_names**.**items()))

7

album\_name **=** album\_names[1592722910]

album\_values**=** list(album\_groups[1592722910])

album\_songs\_count **=** len(list(set(map(**lambda** x: x['trackId'], album\_values))))

dataGenre\_Album **=** (list((map(**lambda** x: x['primaryGenreName'], album\_values))))

releaseDate\_Album **=** list(set(map(**lambda** x: x['releaseDate'], album\_values)))

songsDuration\_Album **=** sum(list(map(**lambda** x : x['trackTimeMillis'],album\_values)))

Average\_time\_Album **=** statistics**.**mean(list(map(**lambda** x : x['trackTimeMillis'],album\_values)))

median\_time\_Album **=** list(map(**lambda** x : x['trackTimeMillis'],album\_values))

median\_time\_Album**.**sort()

median\_time\_Album **=** statistics**.**median(median\_time\_Album)

longest\_song\_Album **=** max(album\_values, key**=**time\_extractor)

shortest\_song\_Album **=** min(album\_values, key**=**time\_extractor)

dearest\_song\_Album **=** max(album\_values, key**=**track\_price\_extractor)

сheapest\_song\_Album **=** min(album\_values, key**=**track\_price\_extractor)

songsCost\_Album **=** sum(filter\_list(list(map(**lambda** x: x['trackPrice'], album\_values)),**-**1))

Average\_cost\_Album **=** statistics**.**mean(filter\_list(list(map(**lambda** x: x['trackPrice'], album\_values)),**-**1))

median\_cost\_Album **=** filter\_list(list(map(**lambda** x: x['trackPrice'], album\_values)),**-**1)

median\_cost\_Album**.**sort()

median\_cost\_Album **=** statistics**.**median(median\_cost\_Album)

8

print('Album name:',album\_name)

print('Count of songs:',album\_songs\_count)

print('Most popular genre count:','Pop -',dataGenre\_Album**.**count('Pop'))

print('Album release date : ',**\***releaseDate\_Album)

print('Songs Duration:',convert\_millis(songsDuration\_Album))

print('Average time:',convert\_millis(Average\_time\_Album))

print('Median duration songs:',convert\_millis(median\_time\_Album))

print('Longest song:', longest\_song\_Album['trackName'], convert\_millis(longest\_song\_Album['trackTimeMillis']))

print('Shortest song:', shortest\_song\_Album['trackName'], convert\_millis(shortest\_song\_Album['trackTimeMillis']))

print('Total cost of songs ≈ ',round(songsCost\_Album),'$')

print('Average cost:',Average\_cost\_Album,'$')

print('Median cost songs:',median\_cost\_Album,'$')

print('The most expensive song:', dearest\_song\_Album['trackName'],'Trackrpice:',dearest\_song\_Album['trackPrice'])

**if** сheapest\_song['trackPrice'] **==** **-**1 :

print('Сheapest song:', сheapest\_song\_Album['trackName'],'Track price:','Free')

**else**:

print('Сheapest song:', сheapest\_song\_Album['trackName'],'Track price:',сheapest\_song\_Album['trackPrice'])