

OBSERVACIONES DEL RETO 1

Tabla 1. Parámetros Máquinas

	Máquina 1	Máquina 2
Procesadores	Intel® Core™ i7-8550U CPU @ 1.8GHZ, 2.00GHZ	Intel® Core™ i5-8250U CPU @ 1.6GHZ, 1800 Mhz
Memoria RAM (GB)	16GB	8.0 GB
Sistema Operativo	Windows 10 Home-64-bits	Windows 10 Home-64-bits

En la Tabla 1 se observan algunos parámetros de las máquinas utilizadas que pueden llegar a afectar el tiempo de ejecución y el tamaño de la muestra hasta la cual se alcanzan a cargar los datos. Para el análisis del reto 1 se realizará por los cuatro requerimientos:

1. **Encontrar buenos videos por categoría y país (Grupal)**

Parámetros Entrada:

Categoría: Se selecciona una categoría de acuerdo a su name del category-id.csv

País (country): Se selecciona un país (country) del videos-large.csv o videos-small.csv

Número de videos para listar (n): Se selecciona int(n) como la cantidad de videos que le gustaría al usuario visualizar.

Return: Debe retornar la cantidad n seleccionada de los videos con los parámetros: Nombre del video (title) – trending_date – Nombre del canal (channel_title) – publish_time – Reproducciones (views) – likes – dislikes.

```
def sortvideospais(catalog, size, pais, categoria):
    videos = catalog['videos']
    videospais = lt.newList()
    conet = 1
    start_time = time.process_time()
    for catg in lt.iterator(catalog['category']):
        if catg["name"] == categoria:
            break
        conet += 1
    categoriafinal = lt.getElement(catalog["category"], conet)
    for cont in range(1, lt.size(catalog["videos"])):
        video = lt.getElement(videos, cont)
        if video["country"] == pais and video["category_id"] == categoriafinal["tag_id"]:
            lt.addLast(videospais, video)
    videospais = mt.sort(videospais, cmpVideosByViews)
    videospaisfinal = lt.subList(videospais, 0, size)
    stop_time = time.process_time()
    elapsed_time_mseg = (stop_time - start_time)*1000
    return (videospaisfinal, elapsed_time_mseg)
```

Figura 1. Código implementado requerimiento 1

2. Encontrar video tendencia por país (Estudiante 2)

Parámetros Entrada:

País (country): Se selecciona un país (country) del videos-large.csv o videos-small.csv

Return: Esta función debe retornar, de acuerdo al parámetro país (country), el vídeo que más días lleva trending con la información: Nombre del video (title) – Nombre del canal (channel_title) – Pais (country) – Número de días como tendencia.

```
def number_one_video(catalog, country):  
  
    videos = catalog["videos"]  
    sublist_country = lt.newList("ARRAY_LIST")  
    start_time = time.process_time()  
  
    for x in range(0, lt.size(catalog["videos"])):  
        pays = lt.getElement(videos, x)["country"]  
        if pays == country.lower():  
            lt.addLast(sublist_country, lt.getElement(catalog['videos'], x))  
    lt_country = sublist_country  
    lt_country = mt.sort(lt_country, cmpfunctionByVideoid)  
  
    sublist_title = lt.newList("ARRAY_LIST")  
    j = lt_country["elements"]  
  
    listu = [x["video_id"] for x in j]  
  
    contador = 0  
    initial_title = listu[0]  
    for elemento in listu:  
        freq = listu.count(elemento)  
        if (freq > contador):  
            contador = freq  
            initial_title = elemento  
    video_id = initial_title  
    count = contador  
  
    for x in range(1, lt.size(videos)):  
        ct = lt.getElement(videos, x)["country"]  
        vid_id = lt.getElement(videos, x)["video_id"]  
        if ct == country and vid_id == video_id:  
            lt.addLast(sublist_title, lt.getElement(catalog['videos'], x))  
    videoTrend = mt.sort(sublist_title, cmpVideosByViews)  
    videoTrend = lt.subList(videoTrend, 0, 1) |  
    stop_time = time.process_time()  
    elapsed_time_mseg = (stop_time - start_time)*1000  
  
    dic = {'title': videoTrend["elements"][0]["title"], "channel_title": videoTrend["elements"][0]["channel_title"],  
          "country": country, "number_days": count, "elapsed_time_mseg": elapsed_time_mseg}  
    return dic
```

Figura 2. Código implementado requerimiento 2

3. Encontrar video tendencia por categoría (Estudiante 1)

Parámetros Entrada:

Categoría(category_name): se ingresa la categoría de video que se quiere filtrar la cual debe estar en el archivo category-id.csv para encontrar el video con más días en trending dentro de esa categoría.

```
def sortvideobyctrending(catalog, categoria):
    videos = catalog['videos']
    videospais = lt.newList("ARRAY_LIST")
    start_time = time.process_time()
    conet = 1
    for catg in lt.iterator(catalog['category']):
        if catg["name"] == categoria:
            break
        conet += 1
    categoriafinal = lt.getElement(catalog["category"], conet)
    for cont in range(1, lt.size(catalog["videos"])):
        video = lt.getElement(videos, cont)
        if video["category_id"] == categoriafinal["tag_id"]:
            lt.addLast(videospais, video)
    lista = []
    dic = {
    }
    max = 0
    titulo = ""
    for pos in range(1, lt.size(videospais)):
        videoid = lt.getElement(videospais, pos)
        if videoid["video_id"] != "#NAME?":
            lista.append(videoid["video_id"])
            dic[videoid["video_id"]] = videoid

    for x in lista:
        if lista.count(x) > max:
            max = lista.count(x)
            titulo = x

    print("días : " + str(max))
    stop_time = time.process_time()
    elapsed_time_mseg = (stop_time - start_time)*1000
    print("Elapsed Time ms:", elapsed_time_mseg)
    return (dic[titulo])
```

Figura 3. Código implementado requerimiento 3

Return:

Debe de retornar el título de video, el nombre del canal del video, su categoría, y el número de días que estuvo

4. Buscar los videos con más likes (Grupal)

Parámetros Entrada:

País (country): Se selecciona un país (country) del videos-large.csv o videos-small.csv

Número de videos para listar (n): Se selecciona int(n) como la cantidad de videos que le gustaría al usuario visualizar.

Etiqueta del video (Tag): palabra relacionada con la que se va a buscar el los tags del vídeo obtenidos del parámetro tags de videos-large.csv o videos-small.csv

Return: Regresa los vídeos ordenados del mayor número de likes al menor de acuerdo a los parámetros insertados. Retorna Nombre del video (title) – Nombre del canal (cannel_title) – publish_time – Reproducciones (views) – likes – dislikes – tags.

```
def VideoByTagLikes(catalog, country, size, tag):  
  
    videosct = lt.newList("ARRAY_LIST")  
    start_time = time.process_time()  
  
    for x in range(0, lt.size(catalog["videos"])):  
        ct = lt.getElement(catalog['videos'], x)["country"]  
        tag_c = lt.getElement(catalog['videos'], x)["tags"]  
        if ct == country and (tag in str(tag_c)) == True:  
            lt.addLast(videosct, lt.getElement(catalog['videos'], x))  
  
    videosct = mt.sort(videosct, cmpVideosByLikes)  
    final_lt = lt.subList(videosct, 1, size)  
  
    stop_time = time.process_time()  
    elapsed_time_mseg = (stop_time - start_time)*1000  
    print("Elapsed Time:", elapsed_time_mseg)  
    return (final_lt)
```

Figura 4. Código implementado requerimiento 4

Consideraciones Preliminares: Dado el análisis visto en el laboratorio 4 y 5, en conjunto con la teoría presentada en clase, se optó por utilizar el algoritmo de ordenamiento recursivo mergesort (peor caso $O(n \log n)$). Asimismo, aunque seleccionando la opción 1 se puede cambiar el tipo de EDA, se va a utilizar "ARRAY_LIST" ya que tiene un $O(1)$ en todas las funciones que se implementaron con el TAD lista (newlist, size, getelement, firstelement).

Resultados Obtenidos por Requerimiento

Para obtener el tiempo de ejecución total por requerimiento se escogieron parámetros bases para comparar entre ambas máquinas. Esto se hizo con el fin de ver el distinto funcionamiento entre ambas máquinas y su complejidad.

Requerimiento 1

Categoría: Music

País: Canadá

N: 3

```
Indique tamaño de la muestra: 3  
Indique el país: canada  
Indique la categoría: Music  
18.28.05 ZAYN - Entertainer (Official Video) 2018-05-23T16:00:03.000Z ZaynVEVO 9501171 498326 12238  
Elapsed Time ms: 0.0  
18.28.05 ZAYN - Entertainer (Official Video) 2018-05-23T16:00:03.000Z ZaynVEVO 9501171 498326 12238  
Elapsed Time ms: 0.0  
17.26.12 Diljit Dosanjh | Raat Di Gedi (Official Video) Neeru Bajwa | Jatinder Shah | Arvindr Khaira 2017-12-23T11:32:18.000Z Speed Records 6973988 219682 6331
```

Figura 5. Resultado Obtenido req 1

Requerimiento 2

País: Canadá

```
Este es el req 2  
Ingrese el país al que le gustaría ver el video con más días como tendencia:  
canada  
{'title': 'Post Malone - rockstar ft. 21 Savage', 'channel_title': 'PostMaloneVEVO', 'country': 'canada', 'number_days': 8, 'elapsed_time_mseg': 133765.625}
```

Figura 6. Resultado Obtenido req 2

Requerimiento 3

Categoría: Music

```
Indique la categoría: Music  
días : 92  
Elapsed Time ms: 135671.875  
Childish Gambino - This Is America (Official Video) ChildishGambinoVEVO 10
```

Figura 7. Resultado requerimiento 3

Requerimiento 4

N: 3

País: Canadá

Tag: 2018

```
{'title': 'VENOM - Official Trailer (HD)', 'channel_title': 'Sony Pictures Entertainment', 'Publish time': '2018-04-24T03:45:03.000Z', 'views': '53071887', 'Likes': '1243479', 'Dislikes': '44414', 'Tags': 'Venom|Venom Movie|Venom (2018)|Marvel|Marvel Comics|Planet of the Symbiotes|Eddie Brock|Tom Hardy|Ruben Fleischer|Spider-man|Spider-man: Homecoming|Michelle Williams|Jenny Slate|Riz Ahmed|Spider-man Spinoff|We Are Venom|Peter Parker|Sony Pictures Entertainment|film|movie|official|official venom movie trailer|Official trailer|sony pictures venom'})  
{'title': 'VENOM - Official Trailer (HD)', 'channel_title': 'Sony Pictures Entertainment', 'Publish time': '2018-04-24T03:45:03.000Z', 'views': '51436997', 'Likes': '1229869', 'Dislikes': '43653', 'Tags': 'Venom|Venom Movie|Venom (2018)|Marvel|Marvel Comics|Planet of the Symbiotes|Eddie Brock|Tom Hardy|Ruben Fleischer|Spider-man|Spider-man: Homecoming|Michelle Williams|Jenny Slate|Riz Ahmed|Spider-man Spinoff|We Are Venom|Peter Parker|Sony Pictures Entertainment|film|movie|official|official venom movie trailer|Official trailer|sony pictures venom'})  
{'title': 'Dua Lipa - IDGAF (Official Music Video)', 'channel_title': 'Dua Lipa', 'Publish time': '2018-01-12T12:00:07.000Z', 'views': '24609441', 'Likes': '1210593', 'Dislikes': '31214', 'Tags': 'dua lipa|idgaf|d11|i don't give a fuck|dua idgaf|dua lipa official|dua new video|dua lipa video|dua leepa|warner bros records|warner Bros|so i cut you off|dua lipa i dont give a fuck|Pop|Dance|2018|Debut Album|i Don't Give A F|Dueling Duas|Dueling Lipas|Double Dua|Double Lipa|Dueling Dua Lipas|Double Dua Lipas'}
```

Figura 8. Resultado requerimiento 4

Resultado Requerimientos

Tabla 2. Parámetros Máquinas			
	Máquina 1	Máquina 2	
Tamaño de la muestra (n) (ARRAY_LIST)	Tiempo (ms)	Tiempo (ms)	Complejidad
Req 1	5718.75	4390,25	O(n)
Req 2	34718.75	129437,5	O(n)
Req 3	4914.625	138187,5	O(n)
Req 4	890.625	765,25	O(n)

Tanto para el código utilizado para el primer y cuarto requerimiento se utiliza un único ciclo para obtener los parámetros requeridos en cada función, además se genera un merge sort de la lista resultante. Debido a esto, en el peor de los casos se deberá recorrer la longitud total de la lista; por lo que, el orden de crecimiento dependerá de la cantidad de elementos $n \rightarrow O(n)$. En el caso del requerimiento 2 y 3 sucede algo similar, pues se realiza un filtro ya sea por categoría o país el cual depende de la cantidad de elementos n que contenga la lista inicial. Esto hace que se recorra la lista de manera lineal para generar un primer filtro, también es necesario generar una nueva lista la cual debe ser nuevamente recorrida para poder hacer el filtrado final.

j.montealegresQuniandes.edu.co
l.gomez1@uniandes.edu.co

Estudiante 1: Juan Montealegre Cod 202012723
Estudiante 2: Lina María Gómez Cod 201923531