Proyecto 1. Introducción a Python

Parte 1: Fundamentos de Programación con Python

Autor: Raúl Quintero Fernández
Instituto: Emtech Institute
Fecha: 12 de febrero del 2022

EMTECH
Emerging Technologies Institute

INDICE

Introducción

En este primer proyecto trabajaremos con los datos de **LifeStore** es una tienda virtual que maneja una amplia gama de artículos, recientemente, la Gerencia de ventas, se percató que la empresa tiene una importante acumulación de inventario. Asimismo, se ha identificado una reducción en las búsquedas de un grupo importante de productos, lo que ha redundado en una disminución sustancial de sus ventas del último trimestre.

Para nuestro trabajo estaremos manejando 3 listas principales, las cuales son:

- **Lista de productos**, la cual cuenta con la siguiente información:
  - *id de producto*: un identificador único para cada producto.
  - *nombre:* el cual es el identificador que se le presenta al consumidor.
  - *precio:* el valor monetario que le da la tienda a dicho producto.
  - *categoría:* hay productos que comparten ciertas características y cuando pasa eso las agrupamos en
    lo llamado: categoría.
  - *stock:* que es la cantidad de piezas que se tiene de dicho artículo.

- **Lista de productos**, la cual cuenta con la siguiente información:
  - *id de ventas:* un identificador único para cada venta.
  - *id de producto:* un identificador único para cada producto y la conexión con la lista de productos.
  - *calificación:* esta representa el valor que le dio el productor al producto después de comprarlo.
  - *fecha:* es el momento sonde se realizó la venta, presentada como día, mes y año.
  - *devolución:* indica si el producto fue regresado después de la venta, con devolución: 1 y sin devolución:0.

- **Lista de búsquedas**, la cual cuenta con la siguiente información:
  - *id de búsquedas:* un identificador único para cada búsqueda.
  - *id de producto:* un identificador único para cada producto y la conexión con la lista de productos.

#lifestore_products = [id_product, name, price, category, stock] (Lista de products)

#lifestore_sales = [id_sale, id_product, score (from 1 to 5), date, refund (1 for true or 0 to false)] (Lista de ventas)

#lifestore_searches = [id_search, id product] (Lista de busquedas)

Objetivo.

- Poner en práctica las bases de programación en Python para análisis y clasificación de datos mediante la creación de programas de entrada de usuario y validaciones, uso y definición de variables y listas, operadores lógicos y condicionales para la clasificación de información.
- LifeStore es una tienda virtual que maneja una amplia gama de artículos, recientemente, la Gerencia de ventas, se percató que la empresa tiene una importante acumulación de inventario. Asimismo, se ha identificado una reducción en las búsquedas de un grupo importante de productos, lo que ha redundado en una disminución sustancial de sus ventas del último trimestre.

Login

Recordemos que el login es un método de seguridad básica en cual consiste en una serie de sentencias donde si se realizan de forma correcta permite seguir con el acceso al código realizado. Para este caso usaremos un bucle *while,* la sentencia *if* y la función *input*, y nuestro login queda de la siguiente forma:

Usuario:
⟹ Emtech
Contraseña:
⟹ Emtech2022

```
# Bienvenida!
mensaje_bienvenida = 'Bienvenide al sistema!\n Accede con tus credenciales'
print(mensaje_bienvenida)

# Recibo constantemente sus intentos
while not usuarioAccedio:
    # Primero ingresa Credenciales
    usuario = input('Usuario: ')
    contras = input('Contraseña: ')
    intentos += 1
    # Reviso si el par coincide
    if usuario == 'Emtech' and contras == 'Emtech2022':
        usuarioAccedio = True
        print('Hola, ahora tienes acceso al proyecto 1 de RQ!')
    else:
        # print('Tienes', 3 - intentos, 'intentos restantes')
        print(f'Lo sentimos el usuario o contraseña esta mal, tienes {3 - intentos} intentos restantes.')

    if intentos == 3:
        exit()

print('Mala suerte, intenta contactarte con el autor si quieres acceder al proyecto.')
```

Resultado 1: respuesta correcta.

```
...

Usuario: Emtech

Contraseña: Emtech2022

Hola, ahora tienes acceso al proyecto 1 de RQ!

>>>
```

Resultado 2: respuestas incorrectas.

```
Usuario: Em
Contraseña: 2020
Lo sentimos el usuario o contraseña está mal, tienes 2 intentos restantes.

Usuario: Emt
Contraseña: E2020
Lo sentimos el usuario o contraseña está mal, tienes 1 intentos restantes.

Usuario: Emte
Contraseña: Emtch
Lo sentimos el usuario o contraseña esta mal, tienes 0 intentos restantes.
Mala suerte, intenta contactarte con el autor si quieres acceder al proyecto.
```

Sección 1: Productos más vendidos y buscados.

Sección 1.1 Lista de los 5 productos con mayores ventas

Como primera parte del análisis de la tienda crearemos una lista con los 5 productos más vendidos, con el fin de identificar los productos que mejores resultados nos están dando y tener en mente migrar a productos con características parecidas. La obtención de estos productos a través de Python se realiza de la siguiente manera:

```
vetas_repetidas=[elemento1[1] for elemento1 in lifestore_sales]

nombres=[nombre[1][0:33] for nombre in lifestore_products]
existencia=[parte[4] for parte in lifestore_products]
cantidad_vendida=[]
x=1
while x<(len(lifestore_products)+1):
    venta=vetas_repetidas.count(x)
    nombre=nombres[x-1]
    stock=existencia[x-1]
    cantidad_vendida.append([venta, nombre, stock])
    x+=1
```

Las `vetas_repetidas,` nos brindó una lista de las ventas pero con una peluciaridad cada valor de la lista es un id_producto causando que si se realizaron n ventas del mismo producto el id del mismo se repetirá el mismo número de veces, con lo que ahora solo necesitamos contar cuantas veces apareció tal id en la lista y para eso usamos *variable.count(x)* y *while,* el primero cuenta cuantas veces aparece el valor *x* en la *variable, que en este caso el* `vetas_repetida.` Ya que sabemos cómo saber cuántas veces apareció un valor ahora lo tenemos que hacer para todos los productos, en este momento usamos el bucle *while,* el cual hacemos correr por todos los id de los productos (`venta`) y guardando los resultados en una sola lista (`cantidad_vendida`), con el comando *.append()* .

Ya tenemos la cantidad de veces que aparece un producto, ahora para identificarlo le agregaremos, lo que nos regresara una lista con de la siguiente forma: [(#ventas de id1, nombre de id1),..] Esto funciona correctamente dado que están en orden, en el primer ciclo cuenta los id de 1 que es el primer producto (`venta`) , pero también extrae el primer elemento de la lista `nombres,` que es el mismo que el id de 1, al proceso los llamamos `nombre,` para terminar todo se junta con *.append*

en la lista `cantidad_vendida`, y esto se repite por la cantidad de productos (`len(lifestore_products)` que hay gracias a *while*.

```
>>> cantidad_vendida
[[2, 'Procesador AMD Ryzen 3 3300X S-AM'], [13, 'Procesador AMD Ryzen
5 3600, S-AM'], [42, 'Procesador AMD Ryzen 5 2600, S-AM'], [13,
'Procesador AMD Ryzen 3 3200G con '], [20, 'Procesador Intel Core i3-
9100F, S'],...]
```

Ejemplo del resultado del código.

Ahora solo nos falta presentar en una lista los 5 productos con mayores ventas, eso lo realizaremos de la siguiente forma:

```
 cantidad_vendida.sort(reverse=True)
1
2 top5_mas_vendidos=cantidad_vendida[0:5]
3
4 for y in top5_mas_vendidos:
5     print(f'El producto {y[1]}..., esta entre los 5 más vendidos con
6 {y[0]} ventas ')
```

Con el comando `variable.sort(reverse=True)` ordeno los valores de mayor a menor, mero solo funciona con los primeros elementos. Ahora solo falta extraer los elementos que me interesan, para este caso con los primeros 5, que son los que tienen los productos más vendidos (`top5_mas_vendidos`), con esto consigo la lista necesaria, por ultimo para presentar los datos corremos un *for* y un pequeño texto, obteniendo la siguiente presentación.

```
El producto SSD Kingston A400, 120GB, SATA II..., esta entre los 5 más vendidos con 50 ventas
El producto Procesador AMD Ryzen 5 2600, S-AM..., esta entre los 5 más vendidos con 42 ventas
El producto Procesador Intel Core i3-9100F, S..., esta entre los 5 más vendidos con 20 ventas
El producto Tarjeta Madre ASRock Micro ATX B4..., esta entre los 5 más vendidos con 18 ventas
El producto SSD Adata Ultimate SU800, 256GB, ..., esta entre los 5 más vendidos con 15 ventas
```

Sección 1.2 Lista de los 10 productos con mayores búsquedas

Como un complemento al punto anterior buscaremos los productos con mayores búsquedas y eso se logra de la siguiente forma:

```
productos_buscados=[elemento1[1] for elemento1 in lifestore_searches]

cantidad_busquedas=[]
x1=1
while x1<(len(lifestore_products)+1):
    contar= productos_buscados.count(x1)
    nombre=nombres[x1-1]
    cantidad_busquedas.append([contar, nombre])
    x1+=1
```

Se realiza de forma similar al punto anterior, haremos una lista con las búsquedas según su id producto (`productos_buscados`) y realizamos el mismo proceso que en el anterior código: contamos cuantas veces aparece el id del producto en las búsquedas lo que nos dará la cantidad de búsquedas y lo acompáñanos con el nombre el producto para hacerlo con todos los id con ayuda de un ciclo *while,* a los resultados los guardamos en `cantidad_busquedas.`

Para la última parte realizamos lo siguiente:

```
cantidad_busquedas.sort(reverse=True)

top10_mas_buscados=cantidad_busquedas[0:10]

for x in top10_mas_buscados:
    print(f'El producto {x[1]}..., se encuentra entre los 10 más buscados con {x[0]} búsquedas')
```

En el sección 1.2 realizamos los mismos pasos: ordenamos de mayor a menor con `variable.sort(reverse=True),` extremos los elementos pedidos que en este caso son los 10 mas buscados igual a los primeros 10 elementos, con esto se obtendría lo requerido pero por estética los presentamos en un *for* con un texto, obteniendo el siguiente resultado:

Como en el anterior ejercicio realizamos los mismos pasos: ordenamos de mayor a menor con `variable.sort(reverse=True),` extremos los elementos pedidos que en este caso son los 10 mas buscados igual a los primeros 10 elementos, con esto se obtendría lo requerido pero por estética los presentamos en un *for* con un texto, obteniendo el siguiente resultado:

```
...
El producto SSD Kingston A400, 120GB, SATA II..., se encuentra entre los 10 más buscados con 263 búsquedas
El producto SSD Adata Ultimate SU800, 256GB, ..., se encuentra entre los 10 más buscados con 107 búsquedas
El producto Tarjeta Madre ASUS micro ATX TUF ..., se encuentra entre los 10 más buscados con 60 búsquedas
El producto Procesador AMD Ryzen 5 2600, S-AM..., se encuentra entre los 10 más buscados con 55 búsquedas
El producto Procesador AMD Ryzen 3 3200G con ..., se encuentra entre los 10 más buscados con 41 búsquedas
El producto Logitech Audífonos Gamer G635 7.1..., se encuentra entre los 10 más buscados con 35 búsquedas
El producto TV Monitor LED 24TL520S-PU 24, HD..., se encuentra entre los 10 más buscados con 32 búsquedas
El producto Procesador Intel Core i7-9700K, S..., se encuentra entre los 10 más buscados con 31 búsquedas
El producto SSD XPG SX8200 Pro, 256GB, PCI Ex..., se encuentra entre los 10 más buscados con 30 búsquedas
El producto Procesador Intel Core i3-9100F, S..., se encuentra entre los 10 más buscados con 30 búsquedas
>>>
```

Material extra de la sección

Ahora que conocemos los productos mas vendidos demos un breve vistazo como queda nuestro inventario con respeto a estos productos:

```
top10_mas_vendidos_stock=cantidad_vendida[0:10]

for q in top10_mas_vendidos_stock:
    print(f'Del producto {q[1]}..., el cual se entre los 5 más
vendidos la tienda cuanta con {q[2]-q[0]} piezas en existencia')
```

Para realizar esta sección extra usamos la lista antes vista que cuenta con la cantidad de ventas, nombre el producto y el elemento que nos interesa ahora: stock. Ahora creamos una nueva lista (top10_mas_vendidos_stock) no solo con el top 5 productos con mas ventas, sino con el top 10 para compararlo con las búsquedas, para calcular el stock actual obtenemos a diferencia entre el stock inicial con las ventas del producto, el resultado de esto se muestra con un bucle *for* y un texto que explicara mejor la situación.

El resultado es el siguiente

```
...
Del producto SSD Kingston A400, 120GB, SATA II..., el cual se entre los 5 más vendidos la tienda cuanta con 250 piezas en existencia

Del producto Procesador AMD Ryzen 5 2600, S-AM..., el cual se entre los 5 más vendidos la tienda cuanta con 945 piezas en existencia

Del producto Procesador Intel Core i3-9100F, S..., el cual se entre los 5 más vendidos la tienda cuanta con 110 piezas en existencia

Del producto Tarjeta Madre ASRock Micro ATX B4..., el cual se entre los 5 más vendidos la tienda cuanta con -18 piezas en existencia

Del producto SSD Adata Ultimate SU800, 256GB, ..., el cual se entre los 5 más vendidos la tienda cuanta con 0 piezas en existencia

Del producto Tarjeta Madre ASUS micro ATX TUF ..., el cual se entre los 5 más vendidos la tienda cuanta con -4 piezas en existencia

Del producto Procesador AMD Ryzen 5 3600, S-AM..., el cual se entre los 5 más vendidos la tienda cuanta con 169 piezas en existencia

Del producto Procesador AMD Ryzen 3 3200G con ..., el cual se entre los 5 más vendidos la tienda cuanta con 282 piezas en existencia

Del producto SSD XPG SX8200 Pro, 256GB, PCI Ex..., el cual se entre los 5 más vendidos la tienda cuanta con -3 piezas en existencia

Del producto Tarjeta de Video ASUS NVIDIA GeFo..., el cual se entre los 5 más vendidos la tienda cuanta con -9 piezas en existencia

>>>
```

Lógranos observar algunas irregularidades las cuales se discutirán en la sección de solución del problema.

Sección 2. Productos menos vendidos y búsquedas

Para el análisis de la tienda es importante saber por donde flaque y esto lo sabemos cuántos de nuestros productos no se venden, esto se logra de la siguiente forma:

```
>>> solo_ventas=[primero[0] for primero in cantidad_vendida]
>>> print(f'Contamos con {solo_ventas.count(0)} productos que no
vendieron en el periodo de estudio ')
Contamos con 54 productos que no vendieron en el periodo de estudio
```

Tenemos 54 productos sin una sola venta, de los 96 productos diferentes que tenemos en existencia.

Ahora obtenemos los productos que no se buscaron, esto de la siguiente forma:

```
>>> solo_buscados=[primero[0] for primero in cantidad_busquedas]
>>> print(f'Contamos con {solo_buscados.count(0)} productos que no se
buscaron en el periodo de estudio ')
Contamos con 40 productos que no se buscaron en el periodo de estudio
```

Tenemos 40 productos sin una sola búsqueda, de los 96 productos diferentes que tenemos en existencia.

Sección 2.1 Top 5 productos menos vendidos por categoría.

Nos damos cuenta de que son muchos los productos que ni son vendidos o buscados así se realizamos una búsqueda de manera un poco mas general buscando los 5 productos con menos ventas, pero ahora desde las categorías y estos se logra de la siguiente forma:

1. **Lista con los elementos que utilizaremos.**

```
extraer_categoria=[ categorias[3] for categorias in
lifestore_products]

cantidad_vendida_id=[]
x=1
while x<(len(lifestore_products)+1):
    venta=vetas_repetidas.count(x)
    nombre=nombres[x-1]
    cat=extraer_categoria[x-1]
    cantidad_vendida_id.append([venta, x, cat, nombre])
    x+=1
```

Explicación:

Como trabajaremos con las categorías extraeremos todas las categorías con *for* y las guardaremos en extraer_categoria.

Contando con las categorías ahora realizaremos un bucle *while* como en la sección 1, esto con el fin de crear una lista parecida a las de la anterior sección pero ahora añadiendo el id del producto y la categoría del mismo.

Nota: los id_producto solo es para revisar si todo esta bien de forma manual, no se usaran después.

2. Ordenamos y creamos diccionario de categorías.

```
cantidad_vendida_id.sort(reverse=False)

diccionario_nombres_por_ventas = {}
for par in cantidad_vendida_id:
    ventas = par[0]
    categoria = par[2]
    nombre1=par[3]
    if categoria not in diccionario_nombres_por_ventas.keys():
        diccionario_nombres_por_ventas[categoria] = []

diccionario_nombres_por_ventas[categoria].append([nombre1[0:24],ventas])
```

Explicación:

Estamos buscando los datos con menores ventas así que ahora ordenamos de menor a mayor; diferencia de la sección 1 ahora el *sort* contara con *False* en lugar de *True,* con este cambio logramos nuestro cometido.

Nosotros ya tenemos ordenados de menor a mayor las ventas de los productos ahora necesitamos subdividirlos por su categoria, esto lo logramos con ayuda de los diccionarios, para crear un diccionario usamos *{},* estos indican que trabajaremos con diccionarios. los diccionarios tienen dos partes: las *keys* son los valores que se repiten con regularidad y estos subdividirán nuestros valores según ellos, para nuestro caso serían las categorías ya que varios productos cuentas con la mismas categoría; la otra parte son los *vaules,* después de separar por *keys* los valores que acompañaban se guardan en una sub lista que tendrá como título su *key.*

En nuestro caso genera lo siguiente:

```
>>> diccionario_nombres_por_ventas

{'procesadores':

[['Procesador Intel Core i3', 0], ['Procesador AMD Ryzen 3 3', 2], ['Procesador Intel Core i9', 3],
['Procesador Intel Core i5', 4], ['Procesador Intel Core i7', 7], ['Procesador AMD Ryzen 5 3', 13],
['Procesador AMD Ryzen 3 3', 13], ['Procesador Intel Core i3', 20], ['Procesador AMD Ryzen 5 2',
42]],

'tarjetas de video':

[['Tarjeta de Video EVGA NV', 0], ['Tarjeta de Video EVGA NV', 0], ['Tarjeta de Video EVGA NV', 0],
['Tarjeta de Video Gigabyt', 0], ['Tarjeta de Video Gigabyt', 0], ['Tarjeta de Video MSI Rad', 0],
['Tarjeta de Video PNY NVI', 0], ['Tarjeta de Video VisionT', 0], ['Tarjeta de Video VisionT', 0], ['MSI
GeForce 210, 1GB GDD', 1], ['Tarjeta de Video Asus NV', 1], ['Tarjeta de Video Gigabyt', 1], ['Tarjeta
de Video MSI NVI', 1], ['Tarjeta de Video Zotac N', 1], ['Tarjeta de Video MSI AMD', 2], ['Tarjeta de
Video Sapphir', 2], ['Tarjeta de Video ASUS AM', 3], ['Tarjeta de Video Gigabyt', 5], ['Tarjeta de
Video ASUS NV', 9]], …
```

Nota: la anterior es parte de la repuesta original y esta retocada para una mejor explicación del funcionamiento.

3. Manejo de diccionarios y resultados.

```
1  extrae_categorias=list(diccionario_nombres_por_ventas)
2  extrae_nombres=list(diccionario_nombres_por_ventas.values())
3
4  top5_productos_categorias=[]
5  x=0
6  while x < len(extrae_categorias):
7      top5_productos_categorias.append([extrae_categorias[x],
8  extrae_nombres[x][0:5]])
9      x+=1
10
11 for muestra_productos in top5_productos_categorias:
       print(muestra_productos)
```

Explicación:

Como vimos en el paso anterior los diccionarios son listas, eso significa que no podemos trabajar con ellos como normalmente lo hacemos, para trabajar con ellos los convertiremos en listas con el comando *list()*.

Sabemos que los diccionarios tienen dos partes así que los dividiremos por eso, en la primera: `extrae_categorias` obtendremos solo las categorías, en la segunda: `extrae_nombres`, tendrá los nombres y la cantidad de búsquedas, pero, con una cualidad importante, tendremos la

misma cantidad de elementos en las dos listas. Como habíamos dicho el diccionario subdividio según las categorías por eso pasa eso.

Ahora solo falta una ultima cosa: unirlas y presentarlas, para esto usaremos un bucle *while* y el comando *append(),* con esto uniremos nuevamente las categorías y los valores, pero aprovecharemos estos últimos para extraer solo los primeros 5 primeros valores de cada categoría así logramos nuestro cometido: en una sola lista (`top5_productos_categorias`) tener por categoría los 5 productos menos vendidos, con el extra de tener la cantidad de ventas.

Por estética presentamos los resultados en un *for* obteniendo lo siguiente:

```
>>> for muestra_productos in top5_productos_categorias:...

['procesadores', [['Procesador Intel Core i3', 0], ['Procesador AMD Ryzen 3 3', 2], ['Procesador Intel Core i9', 3], ['Procesador Intel Core i5', 4], ['Procesador Intel Core i7', 7]]]

['tarjetas de video', [['Tarjeta de Video EVGA NV', 0], ['Tarjeta de Video EVGA NV', 0], ['Tarjeta de Video EVGA NV', 0], ['Tarjeta de Video Gigabyt', 0], ['Tarjeta de Video Gigabyt', 0]]]['tarjetas madre', [['Tarjeta Madre AORUS ATX ', 0], ['Tarjeta Madre ASRock Z39', 0], ['Tarjeta Madre ASUS ATX R', 0], ['Tarjeta Madre Gigabyte m', 0], ['Tarjeta Madre Gigabyte m', 0]]]['discos duros', [['SSD Addlink Technology S', 0], ['SSD para Servidor Superm', 0], ['SSD para Servidor Lenovo', 0], ['SSD para Servidor Lenovo', 0], ['SSD Samsung 860 EVO, 1TB', 0]]]

['memorias usb', [['Kit Memoria RAM Corsair ', 0], ['Kit Memoria RAM Corsair ', 1]]]

['pantallas', [['Makena Smart TV LED 32S2', 0], ['Seiki TV LED SC-39HS950N', 0], ['Samsung TV LED LH43QMREB', 0], ['Samsung Smart TV LED UN7', 0], ['Makena Smart TV LED 40S2', 0]]]

['bocinas', [['Lenovo Barra de Sonido, ', 0], ['Acteck Bocina con Subwoo', 0], ['Verbatim Bocina Portátil', 0], ['Ghia Bocina Portátil BX3', 0], ['Naceb Bocina Portátil NA', 0]]]

['audifonos', [['ASUS Audífonos Gamer ROG', 0], ['Acer Audífonos Gamer Gal', 0], ['Audífonos Gamer Balam Ru', 0], ['Energy Sistem Audífonos ', 0], ['Genius GHP-400S Audífono', 0]]]

>>>
```

Sección 2.2 Top 10 productos menos búsquedas por categoría.

Esta sección es muy parecida a la anterior debido a que la forma de obtención es casi la misma.

1. **Lista con los elementos que utilizaremos.**

```
extraer_categoria

cantidad_buscada_categoria=[]
x=1
while x<(len(lifestore_products)+1):
    busaquedas_contadas=productos_buscados.count(x)
    nombre=nombres[x-1]
    cat=extraer_categoria[x-1]
    cantidad_buscada_categoria.append([busaquedas_contadas, cat,
nombre])
    x+=1
```

Explicación:

Como su similar en la anterior sección utilizaremos la lista que solo tiene las categorías (`extraer_categoria`) y generamos una lista con la cantidad de búsquedas, la categoría y el nombre, con ayuda del bucle *while* y nombran a esta nueva lista como: `cantidad_buscada_categoria=[]`

2. **Ordenamos y creamos diccionario de categorías.**

```
cantidad_buscada_categoria.sort(reverse=False)

diccionario_busquedas_categorias = {}
for par in cantidad_buscada_categoria:
    busqueas_contar = par[0]
    categoria=par[1]
    nombre1=par[2]
    if categoria not in diccionario_busquedas_categorias.keys():
        diccionario_busquedas_categorias[categoria] = []
    diccionario_busquedas_categorias[categoria].append([nombre1[0:24],
busqueas_contar])
```

Explicación:

Como en su contraparte de la anterior sección ordenamos los datos de menor a mayor.

Creamos un diccionario con las *keys* como categorías y los valores como el correspondiente a las categorías en un diccionario con un nombre excesivamente largo: `diccionario_nombres_por_busquedas_categorias,` pero en esta ocasión no mostramos los resultados debido a que ya se tiene la noción de que hace un diccionario.

3. Manejo de diccionarios y resultados.

```
extrae_categorias_buscados=list(diccionario_busquedas_categorias)
extrae_nombres_buscados=list(diccionario_busquedas_categorias.values())

top10_buscados_categorias=[]
x=0
while x < len(extrae_categorias_buscados):
    top10_buscados_categorias.append([extrae_categorias_buscados[x],
extrae_nombres_buscados[x][0:10]])
    x+=1

for muestra_buscados in top10_buscados_categorias:
    print(muestra_buscados)
```

Explicación:

Realizamos casi lo mismo que su contraparte: convertimos las categorías y valores de los diccionarios en listas, unimos las dos listas en un *while* y solo seleccionamos los primeros 10 valores de la segunda lista, con eso obtenemos lo requerido en la sección, con el extra de la cantidad de búsquedas que tiene.

Presentamos los datos con *for* y obtenemos lo siguiente:

...

['audifonos', [['ASUS Audífonos Gamer ROG', 0], ['Acer Audífonos Gamer Gal', 0], ['Audífonos Gamer Balam Ru', 0], ['Energy Sistem Audífonos ', 0], ['Getttech Audífonos con M', 0], ['Klip Xtreme Audífonos Bl', 0], ['Ginga Audífonos con Micr', 1], ['Genius GHP-400S Audífono', 2], ['Iogear Audífonos Gamer G', 3], ['HyperX Audífonos Gamer C', 6]]]

['bocinas', [['Ghia Bocina Portátil BX3', 0], ['Ghia Bocina Portátil BX4', 0], ['Ghia Bocina Portátil BX5', 0], ['Ghia Bocina Portátil BX9', 0], ['Lenovo Barra de Sonido, ', 0],...]

['discos duros', [['SSD Addlink Technology S', 0], ['SSD para Servidor Lenovo', 0], ['SSD para Servidor Superm', 0], ['SSD Samsung 860 EVO, 1TB', 1], ['SSD para Servidor Lenovo', 2], ['SSD Western Digital WD B', 5], ...]

['pantallas', [['Hisense Smart TV LED 40H', 0], ['Hisense Smart TV LED 50H', 0], ['Makena Smart TV LED 32S2', 0], ['Makena Smart TV LED 40S2', 0], ['Samsung Smart TV LED UN3', 0], ['Samsung Smart TV LED UN7', 0], ['Samsung TV LED LH43QMREB', 0], ['Samsung Smart TV LED 43,', 1], ['Samsung Smart TV LED UN5', 4], ['Seiki TV LED SC-39HS950N', 4]]]

['tarjetas de video', [['Tarjeta de Video EVGA NV', 0], ['Tarjeta de Video EVGA NV', 0], ['Tarjeta de Video Gigabyt', 0], ['Tarjeta de Video Gigabyt', 0], ['Tarjeta de Video MSI Rad', 0], ['Tarjeta de Video PNY NVI', 0], ['MSI GeForce 210, 1GB GDD', 1], ['Tarjeta de Video VisionT', 1], ['Tarjeta de Video Asus NV', 2], ['Tarjeta de Video Gigabyt', 3]]]

['tarjetas madre', [['Tarjeta Madre AORUS ATX ', 0], ['Tarjeta Madre ASRock ATX', 0], ['Tarjeta Madre ASRock Z39', 0], ['Tarjeta Madre ASUS ATX P', 0], ['Tarjeta Madre ASUS ATX R', 0], ['Tarjeta Madre ASUS ATX R', 0],

Nota: los anteriores datos fueron manipulados, por el gran volumen de datos algunos fueron recortados y otros omitidos, para conocer los productos completos correr el programa

Material extra de la sección

Para solucionar el problema de inventario realizar búsquedas por categoría es una gran idea por ello calcularemos el top3 productos mas vendidos por categorías, añadiendo la cantidad de piezas vendidas. Para ello realizamos los siguiente

```
top3_mas_vendidos_categorias=[]
q=0
while q < len(extrae_categorias):
    top3_mas_vendidos_categorias.append([extrae_categorias[q],
extrae_nombres[q][-3:]])
    q+=1

for muestra_productos_top in top3_mas_vendidos_categorias:
    print(muestra_productos_top)
```

Explicación:

Para lograr el cometido utilizamos la base de los productos con menos ventas por categoría en la sección 2.1 punto 3, pero esta vez no extraemos los primeros 5 elementos, sino los últimos 3, que como se ordenaron de menor a mayor estos serían los más vendidos.

Para presentar el resultado realizamos un bucle *for,* logrando el siguiente resultado:

```
['procesadores', [['Procesador AMD Ryzen 3 3', 13], ['Procesador Intel Core i3', 20], ['Procesador AMD Ryzen 5 2', 42]]]

['tarjetas de video', [['Tarjeta de Video ASUS AM', 3], ['Tarjeta de Video Gigabyt', 5], ['Tarjeta de Video ASUS NV', 9]]]

['tarjetas madre', [['Tarjeta Madre MSI ATX B4', 6], ['Tarjeta Madre ASUS micro', 14], ['Tarjeta Madre ASRock Mic', 18]]]

['discos duros', [['SSD XPG SX8200 Pro, 256G', 11], ['SSD Adata Ultimate SU800', 15], ['SSD Kingston A400, 120GB', 50]]]

['memorias usb', [['Kit Memoria RAM Corsair ', 0], ['Kit Memoria RAM Corsair ', 1]]]

['pantallas', [['Samsung Smart TV LED UN5', 0], ['TCL Smart TV LED 55S425 ', 1], ['TV Monitor LED 24TL520S-', 1]]]

['bocinas', [['Ghia Bocina Portátil BX4', 0], ['Ghia Bocina Portátil BX5', 0], ['Logitech Bocinas para Co', 2]]]
```

Como podemos observar tenemos unos resultados interesantes que se comentaran en la sección 5.

Sección 3: Productos por reseña en el servicio.

Para valorar la calidad de los productos lo que se realiza al momento de la compra es generar una encuesta donde el consumidor califica el producto adquirido y este se registra en la página principal. Al momento de valorar la calidad de los productos según los consumidores y exponerla puede causar dos situaciones: que el mismo consumidor deje de comprar con nosotros y que otros consumidores interesados en el producto no lo compre.

Al conocer lo anterior podemos cambiar de proveedor o seguir con los mismo, por lo tanto realizamos un estudio con los productos con mejores calificación y los de peor calificación.

Sección 3.1: Top 5 productos con las mejores reseñas.

1. **Sumas de señas**

```
suma_score=[]
sum1=0
for produc in lifestore_products:
    for review in lifestore_sales:
        if produc[0]==review[1]:
            sum1+=review[2]
    suma_score.append(sum1)

scores_totales=[suma_score[0],]
n=1
while n<len(lifestore_products):
    nuevo=suma_score[n]-suma_score[n-1]
    scores_totales.append(nuevo)
    n+=1
```

Explicación:

Para lograr el objetivo lo primero que hacemos es obtener la suma de las reseñas de los artículos con el mismo id que esto se logra con el anterior mostrado para eso se realizo lo siguiente: se creo una lista llamada `suma_score` la cual genera que las reseñas se sumen de forma consecutida, esto es útil ya que solo necesitaríamos restarle el valor anterior para conseguir el valor real y esto se logra con la segunda parte del código. Obtenemos `scores_totales` con ayuda de un bucle *while,* el cual toma el elemento actual de la lista `suma_score` y le resta el anterior elemento, creando una cadena de restas, los resultados de esta resta se formara el nuevo elemento actual y se almacenará con ayuda de *append()* a la lista `scores_totales` como es de suponer para realizar esta resta es necesario restar el valor anterior con lo que no podemos empezar con el primer elemento de lista ya que daría un error, además de que es un valor correcto por su situación no es necesario restarle nada así que empezamos por el segundo elemento. Para obtener

la lista de forma correcta necesitamos contar con el primer elemento, para esto insertamos este elemento de forma directa a la nueva lista.

2. **Cantidad de calificaciones y pre-resultado.**

```
cantidad_score=[]
for produc in lifestore_products:
    a = produc[0]
    b=vetas_repetidas.count(a)
    cantidad_score.append(b)

score_promedio=[]
x=0
while x<96:
    if scores_totales[x]!=0:
        total=scores_totales[x]/cantidad_score[x]
        score_promedio.append(round(total, 3))
    else:
        score_promedio.append(0.00)
    x+=1
```

Explicación:

Ya obtenida la suma de las calificaciones de las reseñas por producto ahora necesitamos saber cuántas veces se calificó el producto, esta respuesta la obtenemos con la primera parte del código en la cual contamos cuantas veces aparece un producto en las ventas y como anteriormente se hizo se utiliza *count()* para logarlo y un bucle *for* para correr por cada elemento de los id de producto, con un resultado almacenado en la lista `cantidad_score`.

Como ya tenemos la suma de las reseñas y las veces que califico podemos calcular el promedio, esto se logra dividiendo la suma de las reseñas, sobre la cantidad de veces que se calificó. El problema está con los productos que no se calificaron, esto produce que su suma sea 0 causando que al momento de realizar la división nos regrese un error en estos casos ya que no podemos dividir sobre 0. Para solucionar el problema realizamos un *if* el cual si la suma de las reseñas es diferente de 0 se realice la división y en caso contrario volver a regresar 0, el resultado se agrega a una lista llamada: `score_promedio.`

Nota: hacemos que regrese 0 para no perder el orden que tenemos, esto se explica con mas detalle en el siguiente punto.

3. Resultado

```
   los_mejores_calificados=[]
1  x=0
 2 while x<96:
 3     los_mejores_calificados.append([score_promedio[x], nombres[x]])
 4     x+=1
 5
 6 los_mejores_calificados.sort(reverse=True)
 7
 8 top5_mejores_calificados=los_mejores_calificados[:5]
 9
10 for mejores_calificados in top5_mejores_calificados:
11     print(f'El producto: {mejores_calificados[1]}..., tiene una
12 calificaión de {mejores_calificados[0]}, esto la hace estar en el
13 top 5 de mejores calificaciones.')
```

Explicación:

Ya logramos el promedio de las calificaiónes ahora les daremos un retoque, uniremos en una lista las califición promedio y el nombre del producto que la obstenta, esto por medio de un bicle *while,* y como en el anterior paso logramos conservar la misma cantidad de productos ahora podemos unirlos 1 a 1 sin problema en la lista `los_mejores_calificados`.

Como esta sección nos pide los 5 mejores calificados entonces ordenamos los valores de mayor a menor con el comando *sort(reverse=True)* y extremos los primeros 5 elementos (`top5_mejores_calificados`), con esto obtenemos la lista de los objetos mejor calificados.

Por estética los presentamos con un bucle *for* y un texto de acompañamiento, logrando el siguiente resultado:

```
El producto: Tarjeta de Video Zotac NVIDIA GeF..., tiene una calificación de 5.0, esto la hace estar en el top 5 mejores calificaciones.

El producto: Tarjeta de Video Sapphire AMD Pul..., tiene una calificación de 5.0, esto la hace estar en el top 5 mejores calificaciones.

El producto: Tarjeta de Video MSI NVIDIA GeFor..., tiene una calificación de 5.0, esto la hace estar en el top 5 mejores calificaciones.

El producto: Tarjeta de Video MSI AMD Mech Rad..., tiene una calificación de 5.0, esto la hace estar en el top 5 mejores calificaciones.

El producto: Tarjeta de Video ASUS AMD Radeon ..., tiene una calificación de 5.0, esto la hace estar en el top 5 mejores calificaciones.
```

Sección 3.2: Top 10 productos con peores reseñas.

1. Resultado

```
   los_mejores_calificados.sort(reverse=False)
1
 2 los_peores_calificados=[]
 3 for x in los_mejores_calificados:
 4     x1=x[0]
 5     x2=x[1]
 6     if x1!=0:
 7         los_peores_calificados.append([x1,x2])
 8
 9 top5_peores_calificados=los_peores_calificados[:5]
10
11 for peores_calificados in top5_peores_calificados:
12     print(f'El producto: {peores_calificados[1]}..., tiene una
13 calificaión de {peores_calificados[0]}, esto la hace estar en el
   top 5 peores calificaciones.')
```

Explicación:

Como ya tenemos gran parte de lo necesario gracias a la anterior sección solo realizamos los cambios correspondientes:

- Ahora ordenamos los valores de menor a mayor con el comando *sort(reverse=True)*.
- Los productos si calificación son omitidos gracias a el comando *if* que guarda todos los valores menos los anteriormente dichos en una nueva lista: `los_peores_calificados`.
- Extraemos los primeros 5 valores que corresponden a los peores calificados, con lo que logramos el objetivo de la sección.

Por estética los presentamos con un bucle *for* y un texto de acompañamiento, logrando el siguiente resultado:

El producto: Tarjeta Madre ASRock ATX H110 Pro..., tiene una calificación de 1.0, esto la hace estar en el top 5 peores calificaciones.

El producto: Tarjeta de Video Gigabyte AMD Rad..., tiene una calificación de 1.0, esto la hace estar en el top 5 peores calificaciones.

El producto: Tarjeta Madre AORUS micro ATX B45..., tiene una calificación de 1.833, esto la hace estar en el top 5 peores calificaciones.

El producto: Tarjeta Madre Gigabyte micro ATX ..., tiene una calificación de 2.0, esto la hace estar en el top 5 peores calificaciones.

El producto: Cougar Audífonos Gamer Phontum Es..., tiene una calificación de 3.0, esto la hace estar en el top 5 peores calificaciones.

Sección 4: Ingresos y ventas.

Como sabemos lo más importante para una tienda a conocer somo están sus finanzas y para ello nosotros calcularemos algunos indicadores como: ingreso total, ingreso por año, meses con mas ventas y promedio de ventas mensualmente. Para esta sección no solo presentaremos las ventas, también los ingresos por esas ventas.

Para facilitar los cálculos se obtendrán algunas listas que se usaran en la mayoría de subsecciones:

```
   #Nos regresara el id del producto vendido y la fecha de su venta
1  por_fecha=[[primer[1], primer[3]] for primer in lifestore_sales]
 2
 3 #obtendremos el id del producto y su precio
 4 precio_de_productos_con_id=[[precio[0], precio[2]] for precio in
 5 lifestore_products]
 6
 7 #Con lo siguiente obtenemos una lista con el precio del producto
 8 vendido por fecha
 9 ingresos_por_fecha=[]
10 for calcular in por_fecha:
11     id=calcular[0]
12     fecha=calcular[1]
13     for extraer in precio_de_productos_con_id:
14         id_precio=extraer[0]
15         precio=extraer[1]
16         if id==id_precio:
               ingresos_por_fecha.append([precio, fecha])
```

Sección 4.1: Ingresos totales.

Tal vez el punto más fácil ya que solo la cantidad de ventas ya las conocemos, es el ultimo id de la lista de ventas, por lo que se omitirá y solo se obtendrán los ingresos totales.

1. **Resultado**

```
>>> ingresos_por_producto=[precio[0] for precio in
ingresos_por_fecha]
>>> print(f'En el periodo estudiado se lograron obtener unos
ingresos de ${sum(ingresos_por_producto)}')

En el periodo estudiado se lograron obtener unos ingresos de
$760177
```

Explicación:

Como al inicio de esta sección se presento una lista con los ingresos según su fecha (ingresos_por_fecha), extraemos la columna de ingresos y guardamos el resultado en una nueva lista llamada: ingresos_por_producto, con esta lista de valores solo necesito sumar de los mismos y esto se logra con el comando *sum(),* con esto logramos el objetivo de la subsección y lo presentamos con texto de acompañamiento y se presenta en la última línea del código de esta sub sección.

Sección 4.2: Ingresos por año.

1. **Nuevo formato de fecha.**

```
fecha_extendida=[]
for fecha in ingresos_por_fecha:
    venta=fecha[0]
    dia=int(fecha[1][0:2])
    mes=int(fecha[1][3:5])
    year=int(fecha[1][6:])
    fecha_extendida.append([venta, dia, mes, year])


fecha_year_2019=[]
fecha_year_2020=[]
for years in fecha_extendida:
    ventas= years[0]
    dias= years[1]
    meses= years[2]
    year= years[3]
    if year==2020:
        fecha_year_2020.append([ventas, dias, meses, year])
    else:
        fecha_year_2019.append([ventas, dias, meses, year])
```

Explicación:

Em la primera parte del código lo que hacemos es separar por partes la fecha y cambiarle a formato numérico, con esto podemos trabajar de mejor forma.

La segunda parte del código es algo opcional, como en este trabajo no tenemos muchos datos podemos observar que solo hay dos fechas si que podemos separarla entre 2019 y 2020, pero a continuación creamos un código que se puede usar en más situaciones, suponiendo que no sabemos cuántos años diferentes hay en el periodo de estudio.

2. Diccionario de fechas

```
diccionario_fecha_extendida = {}
for par in fecha_extendida:
    ventas = par[0]
    categoria = par[3]
    if categoria not in diccionario_fecha_extendida.keys():
        diccionario_fecha_extendida[categoria] = []
    diccionario_fecha_extendida[categoria].append(ventas)

diccionario_fecha_extendida.keys()

extrae_categorias_year=list(diccionario_fecha_extendida)
extrae_ventas_year=list(diccionario_fecha_extendida.values())
```

Explicación:

Como hablamos en el punto anterior la forma común para conocer cuantos años hay en el periodo de estudio es por medio de diccionarios, donde ahora las *keys* serán los años, que es precisamente los que hace la primera parte del código almacenando los datos en: `diccionario_fecha_extendida`,

Como paso rápido para conocer cunatos años hay en nuestro estudio corremos este código `diccionario_fecha_extendida.keys()`, como hemos realizado en secciones anteriores creamos listas con las partes del diccionario las cuales llamamos `extrae_categorias_year` y `extrae_ventas_year`.

3. Resultado.

```
total_ventas_year=[]
x=0
while x<len(extrae_categorias_year):
    total_ventas_year.append([extrae_categorias_year[x],
sum(extrae_ventas_year[x])])
    x+=1

total_ventas_year.sort()

print(f'''\n En el {total_ventas_year[0][0]} se obtuvieron unos
ingresos de ${total_ventas_year[0][1]}, con la venta de
{len(fecha_year_2019)} producto(s) \n
 En el {total_ventas_year[1][0]} se obtuvieron unos ingresos de
${total_ventas_year[1][1]}, con la venta de {len(fecha_year_2020)}
producto(s) '''  )
```

Explicación:

En `total_ventas_year` unimos nuevamente las listas de los diccionarios.

Con `total_ventas_year.sort()` ordenamos de menor a mayor los años, ahora solo falta presentar los datos y esto o hacemos en un *print* el cual tiene un texto especial para que e comprenda mejor el resultado, obteniendo lo siguiente. Como extra se calcularon la cantidad de ventas por año con el comando *len(),*

```
>>> print(f'''\n En el {total_ventas_year[0][0]} se obtuvieron unos ingresos de ${total_ventas_year[0][1]}, con la venta de {len(fecha_year_2019)} producto(s) \n   En el {total_ventas_year[1][0]} se obtuvieron unos ingresos de ${total_ventas_year[1][1]}, con la venta de {len(fecha_year_2020)} producto(s) '''  )

 En el 2019 se obtuvieron unos ingresos de $4209, con la venta de 1 producto(s)

 En el 2020 se obtuvieron unos ingresos de $755968, con la venta de 282 producto(s)

>>>
```

Sección 4.3: Meses con más ventas.

1. **Diccionario meses.**

```
diccionario_meses = {}
for par in fecha_year_2020:
    ventas = par[0]
    categoria = par[2]
    if categoria not in diccionario_meses.keys():
        diccionario_meses[categoria] = []
    diccionario_meses[categoria].append(ventas)

extrae_categorias_meses=list(diccionario_meses)
extrae_ventas_meses=list(diccionario_meses.values())

total_ventas_meses=[]
x=0
while x<len(extrae_categorias_meses):
    total_ventas_meses.append([sum(extrae_ventas_meses[x]),
extrae_categorias_meses[x], len(extrae_ventas_meses[x]) ])
    x+=1
```

Explicación:

Para hacerlo de forma fácil solo usaremos las fechas del 2020, ya que como el 2019 solo tiene un elemento entonces esa seria su mes con mayores ingresos. Continuando con el trabajo creamos un diccionario (`diccionario_meses`) donde en este caso nuestras *keys* serán los meses, no importa

la cantidad de meses del estudio. Como hacemos con los diccionarios, convertimos las partes de los diccionarios en listas. Para luego unirlas en una sola lista que llamamos `total_ventas_meses`.

2. **Resultado.**

```
 total_ventas_meses.sort(reverse=True)
1
2 for ingresos_meses in total_ventas_meses:
3     print(f'En el mes {ingresos_meses[1]} del 2020 se obtuvieron
4 unos ingresos de ${ingresos_meses[0]}, con la venta de
  {ingresos_meses[2]} producto(s)')
```

Explicación:

Ordenamos por cantidad de ventas con ayuda de *sort(reverse=True)* lo que nos genera el resultado deseado, ahora solo lo presentaremos en un *for* acompañado de un texto con el extra de que también tendrá los ingresos además de las ventas, obteniendo como resultado:

```
...
En el mes 4 del 2020 se obtuvieron unos ingresos de $193295, con la venta de 75 producto(s)
En el mes 3 del 2020 se obtuvieron unos ingresos de $164729, con la venta de 51 producto(s)
En el mes 1 del 2020 se obtuvieron unos ingresos de $120237, con la venta de 53 producto(s)
En el mes 2 del 2020 se obtuvieron unos ingresos de $110139, con la venta de 41 producto(s)
En el mes 5 del 2020 se obtuvieron unos ingresos de $96394, con la venta de 36 producto(s)
En el mes 6 del 2020 se obtuvieron unos ingresos de $36949, con la venta de 11 producto(s)
En el mes 7 del 2020 se obtuvieron unos ingresos de $26949, con la venta de 11 producto(s)
En el mes 9 del 2020 se obtuvieron unos ingresos de $4199, con la venta de 1 producto(s)
En el mes 8 del 2020 se obtuvieron unos ingresos de $3077, con la venta de 3 producto(s)
>>>
```

Sección 4.4: Ventas promedio mensual.

1. **Resultado**

```
ingresos_promedio_mensial= round(sum(ingresos_por_producto)/
len(extrae_categorias_meses), 3)
ventas_promedio_mensial=len(fecha_year_2020)//len(extrae_categorias_meses)

print(f'Los ingresos promedio por mes son de ${ingresos_promedio_mensial},
con un promedio de {ventas_promedio_mensial} productos vendidos.')
```

Explicación:

Como en la subsección anterior solo trabajaremos con el 2020 debido que el 2019 solo tiene un valor causando que el mismo sea su promedio de ventas.

Las ventas promedio las obtenemos reusando el resultado de las ventas en 2020 (`sum(ingresos_por_producto))` y la cantidad de meses en el estudio (`len(extrae_categorias_meses))` con esto logramos nuestro objetivo ahora presentamos los datos en un *print,* con el extra de ingresos promedio obteniendo lo siguiente;

```
Los ingresos promedio por mes son de $84464.111, con un promedio de 31 productos vendidos.

>>>
```

Sección 5: Solución al problema.

Al inicio de la sección 2 nos dimos cuenta que en un periodo de un poco mas de 8 meses solo se habían vendido cerca

1. **Sobre inventario**, generado por la poca movilidad de productos debido a que el 56.25% de los productos no han tenido una sola venta en el periodo de estudio.
    a. Se recomienda sacar el producto, ya sea dando descuentos o vendiendo a otras tiendas por convenio.
2. **Estrategia de ventas:** las ventas en el 3 trimestre en adelante tuvieron demasiadas bajas, revisar si se cambio de proveedor, la pagina tuvo problemas o
3. **Calidad de los productos:** realizar pláticas con el proveedor de **tarjetas madre** o cambiar de proveedor. Dentro de los artículos con peor calificación se encuentras 3 tarjetas madre, lo que nos habla de dos cosas.
    a. El producto se está vendiendo, pero con mala recepción.

b. Como el producto se vende, pero tiene mala calificación causa que otros consumidores elijan otra opción.

4. **Venta y búsquedas:** ejemplos como el producto' SSD Adata Ultimate SU800…' se encuentran en el puesto número 2 de búsquedas, pero en el 5 de ventas. Esto ocurrió que se vendieron todas las unidades que se tenían., esto también ocurre con otros productos
5. **Problemas en el inventario:** un ejemplo de los mismo es el producto 'Tarjeta Madre ASUS micro ATX TUF…' del cual se tenían en existencia 10 piezas y se vendieron 13 (cotando devoluciones), aunque también entra en el ejemplo anterior y otros productos.
6. **Ventas por categorías:** en la sección 2 en material extra podemos ver los productos con más ventas por categoría, lo interesante es que tenemos 2 categorías en las que su producto más vendido tiene solo 1 unidad y otras dos categorías donde las ventas de unidades de su producto estrella es de 2. Esto es un complemento al punto 1 el cual sacar estos productos lograría tener un mayor inventario y así poder invertir en nuevos productos.
7. **Donde reinvertir:** al contrario del punto anterior tenemos 3 categorías donde las ventas son mayores, las cuales son: discos duros, procesadores y tarjetas madre.

Sección 6: Conclusión.

Realizar este proyecto fue todo un reto para mí, por la parte de código aunque trabaje más de 9 días en el los primeros pasos fueron complicados, no podía avanzar y me frustraba, no sabia como empezar a contestar las preguntas sin usar funciones o tal vez objetos.

Con el paso el tiempo fui investigando más y más, para el día 7 el código iba por buen camino así que con los nuevos conocimientos que había adquirido como el uso correcto el bucle *while* y el uso de las librerías, decidí hacer otra vez el código ahora cambiando las grandes listas de *if, else* y creando tantas listas como meses del año y como categorías puede cambiar eso.

Estoy muy contento de mi código sé que puedo mejorarlo, pero por ahora aprendí muchísimo, desde lógica matemática a conocer bien los bucles… Por último, la verdad no resolví a que se debe la baja de las búsquedas y ventas. Tendría que ver que pasaba en esos meses, si fue por causa del covid que aumento en esos tiempos o grandes promociones de otras tiendas, espero que al ver gráficas esto pueda ser mas claro para mi.