

FUNDAMENTOS DE PROGRAMACIÓN CON PYTHON

PROYECTO FINAL

EMTECH INSTITUTE

TUTOR: JAIME SAÚL ALONSO SÁNCHEZ

ALUMNO: JORGE A. RODRÍGUEZ GONZÁLEZ



14/09/2021



Índice

1. Introducción	3
2. Definición del código.....	4
2.1. Función <i>segundo_elemento()</i>	4
2.2. Variables predefinidas.....	4
2.2.1. Variable <i>user</i>	4
2.2.2. Variable <i>password</i>	4
2.2.3. Variable <i>login</i>	4
2.2.4. Variable <i>option</i>	4
2.2.5. Variable <i>sub_option</i>	5
2.2.6. Listado de los nombres de los meses.....	5
2.2.7. Variable <i>n_top_ventas</i>	5
2.2.8. Variable <i>n_top_review_calificacion</i>	5
2.2.9. Variable <i>n_top_total_búsquedas</i>	5
2.2.10. Variable <i>n_top_meses_ventas</i>	5
2.2.11. Variable <i>n_bottom_review_calificacion</i>	5
2.2.12. Variable <i>n_bottom_ventas_categoria</i>	6
2.2.13. Variable <i>n_bottom_búsquedas_categoria</i>	6
2.3. Listas declaradas como vacías.....	6
2.3.1. Lista <i>total_ventas</i>	6
2.3.2. Lista <i>total_ventas_categoria_ascendente</i>	7
2.3.3. Lista <i>bottom_total_ventas_categoria</i>	7
2.3.4. Lista <i>total_búsquedas_categoria_ascendente</i>	7
2.3.5. Lista <i>bottom_total_búsquedas_categoria</i>	8
2.3.6. Lista <i>total_búsquedas</i>	8
2.3.7. Lista <i>total_reviews</i>	8
2.3.8. Lista <i>total_calificacion</i>	8
2.3.9. Lista <i>review_calificacion</i>	9
2.3.10. Lista <i>ingresos_mensuales</i>	9
2.3.11. Lista <i>ventas_mensuales</i>	9
2.3.12. Lista <i>top_meses_ventas</i>	10
2.4. Iniciación de listas	10

2.5. Conteo de ventas y reseñas	10
2.6 Promedio de las reseñas	13
2.7. Conteo de búsquedas.....	14
2.8. Listado de los productos más vendidos	15
2.9. Listado de los meses con más ventas del año.....	15
2.10. Listado de los productos con mejores reseñas	15
2.11. Listado de los productos con peores reseñas	16
2.12. Listado de los productos más buscados.....	16
2.13. Listado de los productos menos vendidos por categoría	17
2.14. Listado de los productos menos buscados por categoría	18
2.15. Variables impresión.....	20
2.16. Login	20
2.17. Menú principal	21
2.18. Submenús.....	21
3. Solución al problema.....	22
3.1. Productos más vendidos y productos rezagados.....	22
3.1.1. Listado de los 15 productos más vendidos	22
3.1.2. Listado de los 20 productos más buscados.....	22
3.1.3. Listado de los 5 productos con menos ventas por categoría.....	23
3.1.4. Listado de los 20 productos con menos búsquedas por categoría	24
3.2. Productos por reseña en el servicio	27
3.2.1. Listado de los 20 productos con mejores reseñas	27
3.2.2. Listado de los 20 productos con peores reseñas	27
3.3. Total de ingresos	28
3.3.1. Total de ingresos mensuales	28
3.3.2. Ventas promedio mensuales.....	29
3.3. Total Anual	29
3.4. Meses con más ventas al año.....	29
4. Conclusiones.....	30

1. Introducción

El presente documento reporta el proyecto de final del curso de “Fundamentos de Programación en Python”, el cual fue impartido a través de la plataforma de Emtech Institute.

En el proyecto se pusieron en práctica las bases de programación en Python para la clasificación y el análisis 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.

Todo lo anterior se aplicó en un caso práctico en donde se realizó un análisis de una tienda virtual que lleva por nombre LifeStore, esto con el fin de plantear un análisis de la rotación de productos identificando los siguientes elementos:

- Productos más vendidos y productos rezagados a partir del análisis de las categorías con menores ventas y categorías con menores búsquedas.
- Productos por reseña en el servicio a partir del análisis de categorías con mayores ventas y categorías con mayores búsquedas.
- Sugerir una estrategia de productos a retirar del mercado, así como sugerencia de cómo reducir la acumulación de inventario considerando los datos de ingresos y ventas mensuales.

Para la realización del caso práctico se proporcionó un dataset que contenía datos de los productos, de las ventas y de las búsquedas realizadas en la tienda.

A continuación, se describe como se realizó el análisis y la solución que se propone a la problemática.

2. Definición del código

En esta sección se describe el código en su totalidad de forma secuencial tal y como esta definido en el archivo de Python del proyecto.

2.1. Función *segundo_elemento()*

Esta función fue declarada con el fin de ser utilizada cuando se ordenan las listas a lo largo de todo el código. Esto porque en el parámetro **key** de la función **sorted()** se especifica el valor que tomará dicha función como referencia para realizar el ordenamiento, y dado a que en todos los casos se necesitaba ordenar listas de listas en donde el dato de referencia de ordenamiento era el segundo valor de la sub-lista, se requería una manera de extraer ese segundo dato.

Esa es la razón por lo cual fue declarada.

```
##### Función auxiliar #####  
  
def segundo_elemento(lista):  
    # Se retorna el segundo valor de la lista  
    return lista[1]
```

2.2. Variables predefinidas

Para el funcionamiento del programa se predefinieron algunas variables, las cuales son descritas a continuación.

2.2.1. Variable *user*

Esta variable fue definida con el fin de tener por defecto un nombre de usuario con el cual ingresar.

2.2.2. Variable *password*

Esta variable se definió con el fin de tener una contraseña por defecto para que el usuario tuviera acceso a través de esta.

2.2.3. Variable *login*

Esta variable se definió con el fin de poder censar si el usuario había ingresado o no a través del usuario y contraseña. Si la variable cuenta con el valor de 0, indica que no ha ingresado el usuario, en cambio, si el usuario ingreso, la variable cuenta con el valor de 1.

2.2.4. Variable *option*

Esta variable se definió para guardar el valor que ingresa el usuario en el menú principal. Los valores que puede tener esta variable varían del 1 al 4. Si se ingresa n valor diferente al propuesto se le informa al usuario.

2.2.5. Variable *sub_option*

Esta variable se definió con el fin de guardar el valor que ingresa el usuario en cada uno de los sub-menús del programa. Su valor esperado depende de cada uno de los sub-menús, pero si el usuario ingresa un valor diferente al esperado en cada sub-menú se informa.

```
# Usuario
user = "Jorge"
# Contraseña
password = "emtech"
# Variable que sensa el inicio de sesión,(1 Sesión activa, 0 Sesión inactiva)
login = 0
# Variable que almacena el valor seleccionado en el menú principal
option = 0
# Varianle que almacena el valor seleccionado en los sub menús
sub_option = 0
```

2.2.6. Listado de los nombres de los meses

Esta lista se definió con el fin de auxiliar en la impresión de listas que hacen referencia a los meses del año. Al hacer referencia al índice que corresponde al número del mes menos uno, se muestra el mes correspondiente.

```
# Listado de meses en el año
meses = ["Enero", "Febrero", "Marzo", "Abril", "Mayo", "Junio", "Julio", "Agosto",
"Septiembre", "Octubre", "Noviembre", "Diciembre"]
```

2.2.7. Variable *n_top_ventas*

Esta variable se declaró para conocer la longitud solicitada para la lista de productos con mayores ventas.

2.2.8. Variable *n_top_review_calificacion*

Esta variable se declaró para conocer la longitud solicitada para la lista de los productos con mejor calificación en sus reseñas.

2.2.9. Variable *n_top_total_búsquedas*

Esta variable se declaró para conocer la longitud solicitada para la lista de productos con mayores búsquedas.

2.2.10. Variable *n_top_meses_ventas*

Esta variable se declaró para conocer la longitud solicitada para la lista de meses con mayores ventas.

2.2.11. Variable *n_bottom_review_calificacion*

Esta variable se declaró para conocer la longitud solicitada para la lista de los productos con peor calificación en sus reseñas.

2.2.12. Variable *n_bottom_ventas_categoria*

Esta variable se declaró para conocer la longitud solicitada para la lista de los productos con mayores ventas clasificados por categoría.

2.2.13. Variable *n_bottom_búsquedas_categoria*

Esta variable se declaró para conocer la longitud solicitada para la lista de los productos con mayores búsquedas clasificados por categoría.

```
# Longitud del listado de productos con más ventas
n_top_ventas = 15
# Longitud del listado de los productos con mejores reseñas
n_top_review_calificacion = 20
# Longitud del listado de productos con más búsquedas
n_top_total_búsquedas = 20
# Longitud del listado de mese con mejores ventas
n_top_meses_ventas = 5
# Longitud del listado con los productos con peores reseñas
n_bottom_review_calificacion = 20
# Longitud del listado con peores vents por categoria
n_bottom_ventas_categoria = 5
# Listado del listado de lo productos con emnores búsquedas
n_bottom_búsquedas_categoria = 20
```

2.3. Listas declaradas como vacías

En esta sección se describen las listas que se declararon como vacías para posteriormente ser llenadas según se requería.

2.3.1. Lista *total_ventas*

En esta lista se almacena el total de ventas por cada producto que se tenga en el dataset. Es una lista de listas en donde cada sub lista corresponde al id del producto junto con el total de sus ventas.

```
# Listado del total de ventas por producto
# Esta lista contendrá un listado de listas
# Cada sublista corresponderá a el id del producto con el respectivo número de ventas
# total_ventas = [[id_producto, n_ventas], [id_producto, n_ventas], ...]
total_ventas = []
```

2.3.2. Lista *total_ventas_categoria_ascendente*

En esta lista se almacenan todas las categorías que se encuentren dentro del dataset, por cada categoría se almacena un listado de listas que corresponden a los id de los productos pertenecientes a la categoría especificada junto con el total de ventas de cada producto. Esta lista estará ordenada de manera ascendente.

```
# Listado del total de ventas de cada producto por categoria
# Esta lista contendrá el nombre de cada categoria seguido de los productos pertenecientes a esa categoria
# Los productos serán almacenados en una lista de listas, en donde la sub lista contendrá el id del producto junto
# con el numero de ventas
# La lista de lista estará ordenada de forma ascendente según el número de ventas.
# total_ventas_categoria_ascendente = [[categoria, [[id_producto, n_ventas], [id_producto, n_ventas], ... ]], ...]
total_ventas_categoria_ascendente = []
```

2.3.3. Lista *bottom_total_ventas_categoria*

Esta lista tiene el mismo formato que la lista del punto **2.3.2.**, sin embargo, es más pequeña en longitud, ya que solo contará el número que se especifique para el listado de productos con menores ventas por categoría.

```
# Listado del total de ventas de cada producto por categoria
# Esta lista contendrá el nombre de cada categoria seguido de los productos pertenecientes a esa categoria
# Los productos serán almacenados en una lista de listas, en donde la sub lista contendrá el id del producto junto
# con el numero de ventas
# La lista de lista estará ordenada de forma ascendente según el número de ventas y contará con los productos con menos ventas
# bottom_total_ventas_categoria = [[categoria, [[id_producto, n_ventas], [id_producto, n_ventas], ... ]], ...]
bottom_total_ventas_categoria = []
```

2.3.4. Lista *total_busquedas_categoria_ascendente*

En esta lista se almacenan todas las categorías que se encuentren dentro del dataset, por cada categoría se almacena un listado de listas que corresponden a los id de los productos pertenecientes a la categoría especificada junto con el total de búsquedas de cada producto. Esta lista estará ordenada de manera ascendente.

```
# Listado del total de búsquedas de cada producto por categoria
# Esta lista contendrá el nombre de cada categoria seguido de los productos pertenecientes a esa categoria
# Los productos serán almacenados en una lista de listas, en donde la sub lista contendrá el id del producto junto
# con el numero de búsquedas
# La lista de lista estará ordenada de forma ascendente según el número de búsquedas.
# total_busquedas_categoria_ascendente = [[categoria, [[id_producto, n_búsquedas], [id_producto, n_búsquedas], ... ]], ...]
total_busquedas_categoria_ascendente = []
```


2.3.5. Lista *bottom_total_búsquedas_categoria*

Esta lista tiene el mismo formato que la lista del punto **2.3.4.**, sin embargo, es más pequeña en longitud, ya que solo contará el número que se especifique para el listado de productos con menores búsquedas por categoría.

```
# Listado del total de búsquedas de cada producto por categoría
# Esta lista contendrá el nombre de cada categoría seguido de los productos pertenecientes a esa categoría
# Los productos serán almacenados en una lista de listas, en donde la sub lista contendrá el id del producto junto
# con el número de búsquedas
# La lista de lista estará ordenada de forma ascendente según el número de búsquedas y contará con los productos menos buscados
# bottom_total_búsquedas_categoria = [[categoria, [[id_producto, n_búsquedas], [id_producto, n_búsquedas], ... ]], ...]
bottom_total_búsquedas_categoria = []
```

2.3.6. Lista *total_búsquedas*

En esta lista se almacena el total de búsquedas por cada producto que se tenga en el dataset. Es una lista de listas en donde cada sub lista corresponde al id del producto junto con el total de sus búsquedas.

```
# Listado del total de búsquedas por producto
# Esta lista contendrá un listado de listas
# Cada sublista corresponderá a el id del producto con el respectivo número de búsquedas
# total_búsquedas = [[id_producto, n_búsquedas], [id_producto, n_búsquedas], ...]
total_búsquedas = []
```

2.3.7. Lista *total_reviews*

En esta lista se almacena el total de las reseñas realizadas por cada producto que se tenga en el dataset. Es una lista de listas en donde cada sub lista corresponde al id del producto junto con suma de todas las reseñas.

```
# Listado del total de reviews por producto
# Esta lista contendrá un listado de listas
# Cada sublista corresponderá a el id del producto con el respectivo número de reviews
# total_reviews = [[id_producto, n_reviews], [id_producto, n_reviews], ...]
total_reviews = []
```

2.3.8 Lista *total_calificacion*

En esta lista se almacena la suma de todas las calificaciones de las reseñas por cada producto que se tenga en el dataset. Es una lista de listas en donde cada sub lista corresponde al id del producto junto con la suma de las calificaciones de las reseñas.

```
# Listado del total de la ponderación de reviews por producto
# Esta lista contendrá un listado de listas
# Cada sublista corresponderá a el id del producto con la respectiva ponderación de reviews
# total_calificacion = [[id_producto, ponderación], [id_producto, ponderación], ...]
total_calificacion = []
```

2.3.9. Lista *review_calificacion*

En esta lista se almacena el promedio de todas las calificaciones de las reseñas por cada producto que se tenga en el dataset. Es una lista de listas en donde cada sub lista corresponde al id del producto junto con el promedio de las calificaciones de las reseñas.

```
# Listado del total del promedio en de reviews por producto
# Esta lista contendrá un listado de listas
# Cada sublista corresponderá a el id del producto con el respectivo promedio de reviews
# review_calificacion = [[id_producto, ponderación], [id_producto, ponderación], ...]
review_calificacion = []
```

2.3.10. Lista *ingresos_mensuales*

En esta lista se almacenan todos los años que se encuentren dentro del dataset, por cada año se almacena un listado de listas que corresponden a los id de los meses del año junto con el total de ingresos de cada mes, y también se almacena la cantidad de ingresos correspondientes al año.

```
# Listado que se compone de tres elementos
# El primero es el año
# El segundo es una lista de listas que corresponde a los ingresos por meses del año especificado
# El tercero contendrá los ingresos totales en el año
# Para el listado de meses se tendrá el número del mes correspondiente y el total de ingresos
# ingresos_mensuales = [[año, [[mes, ingresos_mes], [mes, ingresos_mes, ...]], ingresos_año], ...]
ingresos_mensuales = []
```

2.3.11. Lista *ventas_mensuales*

En esta lista se almacenan todos los años que se encuentren dentro del dataset, por cada año se almacena un listado de listas que corresponden a los id de los meses del año junto con el total de ventas de cada mes, y también se almacena la cantidad de ventas correspondientes al año.

```
# Listado que se compone de tres elementos
# El primero es el año
# El segundo es una lista de listas que corresponde a las ventas por meses del año especificado
# El tercero contendrá las ventas totales en el año
# Para el listado de meses se tendrá el número del mes correspondiente y el total de ventas
# ventas_mensuales = [[año, [[mes, ventas_mes], [mes, ventas_mes, ...]], ventas_año], ...]
ventas_mensuales = []
```

2.3.12. Lista *top_meses_ventas*

Esta lista tiene el mismo formato que la lista del punto **2.3.11.**, sin embargo, es más pequeña en longitud, ya que solo contará el número que se especifique para el listado de mese con mayores ventas.

```
# Listado que se compone de tres elementos
# El primero es el año
# El segundo es una lista de listas que corresponde a las ventas por meses del año especificado
# El tercero contendrá las ventas totales en el año
# Para el listado de meses se tendrá el número del mes correspondiente y el total de ventas
# top_meses_ventas = [[año, [[mes, ventas_mes], [mes, ventas_mes, ...]], ventas_año], ...]
top_meses_ventas = []
```

2.4. Iniciación de listas

Debido a que algunas listas deben contar con la misma longitud del total de productos que se tiene en el dataset, se realizó un ciclo que recorriera toda la lista que contenía los productos de la tienda.

Por cada producto, se guardó el id junto con un número cero en cada una de las las siguientes listas:

- ***total_ventas***
- ***total_busquedas***
- ***total_reviews***
- ***total_calificacion***

```
##### Iniciación de listas #####

# Se recorre la lista lifestore_products
for producto in lifestore_products:
    # Para las listas especificadas se guardan el id de los productos existentes seguido de un 0
    # [id_producto, 0]
    id_producto = producto[0]
    total_ventas.append([id_producto, 0])
    total_busquedas.append([id_producto, 0])
    total_reviews.append([id_producto, 0])
    total_calificacion.append([id_producto, 0])
```

2.5. Conteo de ventas y reseñas

Recorriendo la lista que almacenaba las ventas se realizaron distintas operaciones, las cuales se irán describiendo a continuación.

Primero, se almacenaron algunos valores de la lista de ventas, tales como, el id del producto en la venta, la calificación de la reseña del producto, y el valor de devolución.

```
##### Recorrido de la lista de ventas #####

for venta in lifestore_sales:
    # Recorremos cada venta que se ha realizado
    # Se guarda el id del producto consultado
    id_producto = venta[1]
    # Se guarda la calificación (reseña) que se le dio al producto
    calificacion = venta[2]
    # Se guarda el valor de la devolución (refund)
    devolucion = venta[4]
```

Seguido de esto, se verifica si el producto fue devuelto o no, ya que si este fue devuelto no tiene caso contemplar los ingresos ni la venta realizada. Si el valor de devolución es 0, procedemos a contabilizar los ingresos y la venta.

Primero, se contabiliza la venta en la lista **total_ventas** en la posición correspondiente al id del producto. Después guardamos la fecha de la venta para posteriormente extraer el mes y el año, esto tomando las posiciones correspondientes de cada uva en el string.

También se consulta en la lista **lifestore_products** el costo del producto mediante su id y se verifica si el año de la venta que estamos consultando ya se encuentra contemplada en la lista de **ingresos_mensuales**. Esto último, debido a que el programa fue escrito con el fin de guardar todos los años con ventas que se tenían en el dataset, así que se debe ver la opción de agregar el año o bien guardar los valores en la posición que compete.

```
if devolucion == 0:
    # Si devolución es 0 (no se devilvió)
    # Se suma en uno las ventas de ese producto en total de ventas
    total_ventas[id_producto-1][1] += 1
    # Se guarda la fecha en que se realizo la venta
    fecha = venta[3]
    # Se guarda el mes dd/mm/aa
    mes = int(fecha[3:5])
    # Se guarda el año dd/mm/aa
    anio = fecha[6:]
    # Se consulta el precio del producto a través de lifestore_products
    # y se almacena en costo
    costo = lifestore_products[id_producto-1][2]
    # Se contruye una lista de los años que ya estan coontemplados
    # en la lista ingresos_mensuales, y se almacena en anio_contemplados
    anios_contemplados = [registro[0] for registro in ingresos_mensuales]
```

En el caso de que el año ya este contemplado, se busca el índice de este dentro de la lista ***anios_contemplados***, una vez que se tiene el índice se suma el costo del producto a los ingresos del mes que corresponde, al cual accedemos a través de su índice.

Se hace lo mismo para las ventas, se suma en uno la venta para ese mes y se suma el ingreso y las ventas anuales respectivamente.

```
if anio in anios_contemplados:
    # Si el año de la venta ya se encuentra contemplada en la lista
    # Se busca su posición en la lista y se guarda en indice_anio
    indice_anio = anios_contemplados.index(anio)
    # Se suma el costo de la venta
    # al ingreso mensual según el indice_anio y el mes
    ingresos_mensuales[indice_anio][1][mes-1][1] += costo
    # Se suma el costo de la venta
    # al ingreso anual según indice_anio
    ingresos_mensuales[indice_anio][2] += costo
    # Se suma en uno el total de ventas mensuales
    # según indice_anio y el mes
    ventas_mensuales[indice_anio][1][mes-1][1] += 1
    # Se suma en uno el total de ventas anuales
    ventas_mensuales[indice_anio][2] += 1
```

Para el caso en el que el año aún no este contemplado (este no se haya agregado a las listas), se agrega primero a la lista de ***ingresos_mensuales***. Posterior a esto, se crea un listado que representa a los meses, por lo que se genera una sublista para cada mes que contiene un número del 1 al 12 según el mes, seguido de un cero.

Y como se tiene el conocimiento de que el año en cuestión es el último de la lista, agregamos el listado de los meses en la última posición de la lista general. Una vez agregada la lista de los meses, usando el índice del mes, se suma el costo de la venta al mes que le corresponde y también al costo anual.

Todo lo anterior se repite para la lista ***ventas_mensuales***, con excepción de que en lugar de sumar el costo de la venta, se suma una unidad que representa la venta realizada.

```

else:
    # Si el año de la venta no esta contemplado
    # Se agrega a ingresos_mensuales el año, en la última posición
    ingresos_mensuales.append([anio])
    # Se genera una lista para los meses (1 -12)
    # meses_aux = [[1, 0], [2, 0], ... [12, 0]]
    meses_aux = [[m, 0] for m in range(1, 13)]
    # Se agrega la lista generada de los meses a ingresos_mensuales
    # en la última posición a continuación del año agregado
    ingresos_mensuales[-1].append(meses_aux)
    # Se suma el costo al respectivo mes de la venta
    ingresos_mensuales[-1][1][mes-1][1] += costo
    # Se agregan los ingresos anuales a la lista
    ingresos_mensuales[-1].append(costo)

    # Se agrega a ventas_mensuales el año, en la última posición
    ventas_mensuales.append([anio])
    # Se genera una lista para los meses (1 -12)
    # meses_aux = [[1, 0], [2, 0], ... [12, 0]]
    meses_aux = [[m, 0] for m in range(1, 13)]
    # Se agrega la lista generada de los meses a ingresos_mensuales
    # en la última posición a continuación del año agregado
    ventas_mensuales[-1].append(meses_aux)
    # Se suma en uno las ventas al respectivo mes
    ventas_mensuales[-1][1][mes-1][1] += 1
    # Se agregan las ventas anuales a la lista
    ventas_mensuales[-1].append(1)

```

Por último, sin importar que se haya devuelto o no el producto, se suma en uno la lista **total_reviews** en la posición del id del producto y se suma la calificación de la reseña para ese producto en la lista **total_calificacion**.

```

# Se hace el conteo del total de reseñas realizadas al producto
total_reviews[id_producto-1][1] += 1
# Se hace la ponderación de las calificaciones de las reseñas hechas para el producto
total_calificacion[id_producto-1][1] += calificacion

```

2.6 Promedio de las reseñas

Ya que se tiene el total de las reseñas hechas para cada producto y la suma total calificación que de cada una de estas se calcula el promedio.

Se recorre el total de reseñas, consultando primero la cantidad de reseñas realizadas, ya que no tiene caso realizar el calculo del promedio si no se tienen reseñas. Por lo que si la

cantidad de reseñas es diferente de 0, en la lista **review_calificacion** se almacena el id del producto en cuestión seguido de el calculo del promedio de la calificación de las reseñas. El promedio se calcula dividiendo la calificación de la suma de las reseñas, que se encuentra almacenada en **total_calificación**, entre la cantidad de reseñas.

```
##### Cálculo del promedio de calificación en la reseñas #####

for review in total_reviews:
    # Se hace el recorrido en el total de reseñas
    # Se guarda el total de las reseñas
    cantidad_reviews = review[1]
    if cantidad_reviews != 0:
        # Si la cantidad de ventas es diferente de 0
        # Se guarda el id del producto según la reseña
        id_producto = review[0]
        # Se hace el calculo del promedio de las calificaciones de las reseñas
        # y se almacena tanto el id como el promedio
        review_calificacion.append([id_producto, total_calificacion[id_producto-1][1]/total_reviews[id_producto-1][1]])
```

2.7. Conteo de búsquedas

Para el conteo de búsquedas se hace un recorrido a la lista **lifestore_searches** con la ayuda de un ciclo **for**.

Por cada búsqueda se almacena el id del producto y a través de este, se suma en una unidad la posición del id menos uno en la lista **total_busquedas**.

```
##### Conteo de búsquedas totales por producto #####

for busqueda in lifestore_searches:
    # Se hace un recorrido a través de la lista de búsquedas
    # Se almacena el id del producto
    id_producto = busqueda[1]
    # Se suma en uno el total de búsquedas para el producto según el id
    total_busquedas[id_producto-1][1] += 1
```

2.8. Listado de los productos más vendidos

Una vez que se hizo el calculo del total de ventas, con ayuda de la función **sorted** se ordena la lista **total_ventas** de forma descendente según la cantidad de ventas por producto y la lista ordenada se guarda en **total_ventas_descendente**.

Seguido se toman los primeros elementos de la lista que especifica la variable **n_top_ventas** y estos se guardan en la lista **top_total_ventas**.

```
##### Listado de los (n_top_ventas) productos más vendidos #####  
  
# Se ordena la lista del total_ventas de mayor a menor según las ventas y se almacena en total_ventas_descendente  
total_ventas_descendente = sorted(total_ventas, key = segundo_elemento, reverse=1)  
# Se genera una lista tomando los (n_top_ventas) productos más vendidos de total_ventas_descendente  
# y se almacena en top_total_ventas  
top_total_ventas = total_ventas_descendente[:n_top_ventas]
```

2.9. Listado de los meses con más ventas del año

Una vez que se tienen las ventas mensuales al año para todos los años dentro del dataset almacenados en la lista **ventas_mensuales**, se recorre dicha lista y por cada año, se ordenan los meses según el total de ventas de cada uno a través de la función **sorted**, y una vez ordenados se toman los primeros valores especificados por la variable **n_top_meses_ventas** y se almacenan en la lista **top_meses_ventas**.

```
##### Listado de los (n_top_meses_ventas) meses con más ventas #####  
  
for anio in ventas_mensuales:  
    # Se hace el recorrido de todos los años contemplados  
    # Se toma de la listas por año, para los meses se toman solo los primeros (n_top_meses_ventas) de la lista  
    # los cuales son ordenados al momento a través de la función sorted  
    top_meses_ventas.append([anio[0], sorted(anio[1][:n_top_meses_ventas], key=segundo_elemento, reverse=1), anio[2]])
```

2.10. Listado de los productos con mejores reseñas

Una vez que se hizo el cálculo el promedio de las calificaciones de las reseñas, con ayuda de la función **sorted** se ordena la lista **review_calificacion** de forma descendente según el promedio de la reseña por producto y la lista ordenada se guarda en **review_calificacion_descendente**.

Seguido se toman los primeros elementos de la lista que especifica la variable **n_top_review_calificacion** y estos se guardan en la lista **top_review_calificacion**.


```
##### Listado de los (n_top_review_calificacion) productos con mejores reseñas #####

# Se ordena de forma descendente la lista review_calificacion según el segundo valor de las sub-listas
# y se almacena en review_calificacion_descendente
review_calificacion_descendente = sorted(review_calificacion, key = segundo_elemento, reverse=1)

# Se toman los primeros (n_top_review_calificacion) valores de la lista y se almacenan
# en top_review_calificacion
top_review_calificacion = review_calificacion_descendente[:n_top_review_calificacion]
```

2.11. Listado de los productos con peores reseñas

Una vez que se hizo el cálculo el promedio de las calificaciones de las reseñas, con ayuda de la función **sorted** se ordena la lista **review_calificacion** de forma ascendente según el promedio de la reseña por producto y la lista ordenada se guarda en **review_calificacion_ascendente**.

Seguido se toman los primeros elementos de la lista que especifica la variable **n_bottom_review_calificacion** y estos se guardan en la lista **bottom_review_calificacion**.

```
##### Listado de los (n_bottom_review_calificacion) productos con peores reseñas #####

# Se ordena de forma ascendente la lista review_calificacion según el segundo valor de las sub-listas
# y se almacena en review_calificacion_ascendente
review_calificacion_ascendente = sorted(review_calificacion, key = segundo_elemento)

# Se toman los primeros (n_bottom_review_calificacion) valores de la lista y se almacenan
# en bottom_review_calificacion
bottom_review_calificacion = review_calificacion_ascendente[:n_bottom_review_calificacion]
```

2.12. Listado de los productos más buscados

Una vez que se hizo el cálculo del total de búsquedas, con ayuda de la función **sorted** se ordena la lista **total_búsquedas** de forma descendente según la cantidad de búsquedas por producto y la lista ordenada se guarda en **total_búsquedas_descendente**.

Seguido se toman los primeros elementos de la lista que especifica la variable **n_top_total_búsquedas** y estos se guardan en la lista **top_total_búsquedas**.

```
##### Listado de los (n_top_total_búsquedas) productos más buscados #####

# Se ordena de forma descendente la lista total_búsquedas según el segundo valor de las sub-listas
# y se almacena en total_búsquedas_descendente
total_búsquedas_descendente = sorted(total_búsquedas, key = segundo_elemento, reverse=1)

# Se toman los primeros (n_top_total_búsquedas) valores de la lista y se almacenan
# en top_total_búsquedas
top_total_búsquedas = total_búsquedas_descendente[:n_top_total_búsquedas]
```

2.13. Listado de los productos menos vendidos por categoría

Una vez que se hizo el cálculo del total de ventas, con ayuda de la función **sorted** se ordena la lista **total_ventas** de forma ascendente según la cantidad de ventas por producto y la lista ordenada se guarda en **total_ventas_ascendente**.

```
##### Listado de los (n_bottom_ventas_categoria) productos menos vendidos por categoria ##  
  
# Se ordena de forma ascendente la lista total_ventas según el segundo valor de las sub-listas  
# y se almacena en total_ventas_ascendente  
total_ventas_ascendente = sorted(total_ventas, key = segundo_elemento)
```

Se recorre la lista **total_ventas_ascendente**, por cada venta se guarda el id del producto y con este se consulta a través de la lista **lifestore_products** la categoría a la que pertenece el producto y se verifica si la categoría del producto que se está consultando ya se encuentra contemplada en la lista de **total_ventas_categoria**. Esto último, debido a que el programa fue escrito con el fin de guardar todas las categorías se tenían en el dataset, así que se debe ver la opción de agregar la categoría o bien guardar los valores en la posición que compete.

```
for venta in total_ventas_ascendente:  
    # Se recorren las ventas almacenadas en total_ventas_ascendente  
    # Se almacena el id del producto  
    id_producto = venta[0]  
    # Se consulta categoría según el id del producto y se almacena en categoria  
    categoria = lifestore_products[id_producto-1][3]  
    # Se crea un listado de las categorías que ya se encuentran contempladas en total_ventas_categoria_ascendente  
    categorias_listadas = [registro[0] for registro in total_ventas_categoria_ascendente]
```

En el caso de que la categoría ya este contemplada, se busca el índice de este dentro de la lista **categorías_listadas**, una vez que se tiene el índice, se agrega la venta en el índice que compete.

```
if categoria in categorias_listadas:  
    # Si la categoría ya se encuentra contemplada en el listado  
    # Se consulta su índice y se almacena  
    indice = categorias_listadas.index(categoria)  
    # Según el índice de la categoría, se guarda en el listado de ventas la venta consultada  
    total_ventas_categoria_ascendente[indice][1].append(venta)
```

En caso de no estar contemplada la categoría en el listado, se agrega y como sabemos la posición de la categoría que acabamos de agregar (que es la última), agregamos la venta.

```
else:
    # Si la categoría no se encuentra contemplada en el listado
    # Se agrega la nueva categoría al final de la lista total_ventas_categoria_ascendente
    total_ventas_categoria_ascendente.append([categoria])
    # Se guarda en el listado de ventas la venta consultada
    total_ventas_categoria_ascendente[-1].append([venta])
```

Por último, recorremos la lista **total_ventas_ascendente**, y por cada categoría extraemos los primeros elementos definidos por **n_bottom_ventas_categoria** y se almacenan en **bottom_total_ventas_categoria**.

```
for ventas_categoria in total_ventas_categoria_ascendente:
    # Se recorre la lista total_ventas_categoria_ascendente
    # Se toman los primeros (n_bottom_ventas_categoria) valores de las listas de ventas por cada categoría
    # y se almacenan en bottom_total_ventas_categoria
    bottom_total_ventas_categoria.append([ventas_categoria[0], ventas_categoria[1][:n_bottom_ventas_categoria]])
```

2.14. Listado de los productos menos buscados por categoría

Una vez que se hizo el cálculo del total de búsquedas, con ayuda de la función **sorted** se ordena la lista **total_búsquedas** de forma ascendente según la cantidad de ventas por producto y la lista ordenada se guarda en **total_búsquedas_ascendente**.

```
##### Listado de los (n_bottom_búsquedas_categoria) productos menos buscados por categoría #####
# Se ordena de forma ascendente la lista total_búsquedas según el segundo valor de las sub-listas
# y se almacena en total_búsquedas_ascendentes
total_búsquedas_ascendentes = sorted(total_búsquedas, key = segundo_elemento)
```

Se recorre la lista **total_búsquedas_ascendente**, por cada búsqueda se guarda el id del producto, con este se consulta a través de la lista **lifestore_products** la categoría a la que pertenece el producto y se verifica si la categoría del producto que se está consultando ya se encuentra contemplada en la lista de **total_búsquedas_categoria**. Esto último, debido a que el programa fue escrito con el fin de guardar todas las categorías se tenían en el dataset, así que se debe ver la opción de agregar la categoría o bien guardar los valores en la posición que compete.

```

for busqueda in total_busquedas_ascendentes:
    # Se recorren las búsquedas almacenadas en total_busquedas_ascendentes
    # Se almacena el id del producto
    id_producto = busqueda[0]
    # Se consulta categoría según el id del producto y se almacena en categoría
    categoria = lifestore_products[id_producto-1][3]
    # Se crea un listado de las categorías que ya se encuentran contempladas en total_busquedas_categoria_ascendente
    categorias_listadas = [registro[0] for registro in total_busquedas_categoria_ascendente]
    if categoria in categorias_listadas:

```

En el caso de que la categoría ya este contemplada, se busca el índice de este dentro de la lista ***categorías_listadas***, una vez que se tiene el índice, se agrega la búsqueda en el índice que compete.

```

    if categoria in categorias_listadas:
        # Si la categoría ya se encuentra contemplada en el listado
        # Se consulta su índice y se almacena
        indice = categorias_listadas.index(categoria)
        # Según el índice de la categoría, se guarda en el listado de búsquedas la búsqueda consultada
        total_busquedas_categoria_ascendente[indice][1].append(busqueda)

```

En caso de no estar contemplada la categoría en el listado, se agrega y como sabemos la posición de la categoría que acabamos de agregar (que es la última), agregamos la búsqueda.

```

else:
    # Si la categoría no se encuentra contemplada en el listado
    # Se agrega la nueva categoría al final de la lista total_busquedas_categoria_ascendente
    total_busquedas_categoria_ascendente.append([categoria])
    # Se guarda en el listado de búsquedas la búsqueda consultada
    total_busquedas_categoria_ascendente[-1].append([busqueda])

```

Por último, recorreremos la lista ***total_busquedas_ascendente***, y por cada categoría extraemos los primeros elementos definidos por ***n_bottom_busquedas_categoria*** y se almacenan en ***bottom_total_busquedas_categoria***.

```

for busquedas_categoria in total_busquedas_categoria_ascendente:
    # Se recorre la lista total_busquedas_categoria_ascendente
    # Se toman los primeros (n_bottom_busquedas_categoria) valores de las listas de búsquedas por cada categoría
    # y se almacenan en bottom_total_busquedas_categoria
    bottom_total_busquedas_categoria.append([busquedas_categoria[0], busquedas_categoria[1][:n_bottom_busquedas_categoria]])

```

2.15. Variables impresión

Se agregan dos variables para darle formato a la impresión en pantalla.

La primera, **limpieza_pantalla** contiene 50 saltos de línea que, al ser impresos, dan la impresión de que la pantalla se limpió.

Por otro lado, se tiene a **separador**, que almacena 143 asteriscos y esta será utilizada para formato de las tablas.

```
# Variable que se imprime para limpieza de pantalla
limpieza_pantalla = "\n" * 50
# Variable que se imprime como separador
separador = "*" * 143
```

2.16. Login

A continuación, se muestra como está compuesta la interfaz que se le muestra al usuario.

Primero, tenemos el **while** global del programa, que mientras la variable **option**, que es la que almacena la opción seleccionada del menú principal, sea diferente de 4 el programa se repetirá.

Seguido, se encuentra el **while** de inicio de sesión, para el cual se usa la variable **login**, que mientras esta sea 0, continuará el ciclo.

Dentro del ciclo se le solicita el usuario ingresar el usuario y contraseña.

```
separador = "*" * 143
while option != 4:
    # Mientras option sea diferente a 4, se repetirá el programa
    ##### Inicio de sesión #####
    while login == 0:
        # Mientras la variable login se encuentre en 0 (Sesión inactiva)
        # Se va a repetir el código dentro del while
        print("*" * 26 + " Inicio de sesión " + "*" * 26)
        # Se solicita el usuario
        user_login = input("Ingresa el usuario: ")
        # Se solicita la contraseña
        password_login = input("Ingresa la contraseña: ")
        # Se comparan los credenciales
```

Una vez ingresado el usuario y contraseña se verifica si estos hacen match con los predefinidos, si es así, se cambia el valor de la variable **login** por lo que el ciclo de inicio de sesión se rompe. Sin embargo, si no hacen match, se le informa al usuario que algún valor está incorrecto y se le solicitan de nuevo los valores (se repite el ciclo).

```

if user == user_login and password == password_login:
    # login cambia a 1 (Sesión activa)
    login = 1
    print("\n\t Haz accedido \n")
else:
    print("\n\t Usuario o contraseña incorrecta\n")

```

2.17. Menú principal

Cuando se ingresa, se muestra el menú principal al usuario.

Y se le pide seleccionar una opción de las mostradas. Según la opción que elija se le dirige a un submenú.

```

# Se imprime el menú principal
print("***30 + " Lifestore " + "***30)
print("Elija una opción:")
print("\t 1. Productos más vendidos y productos rezagados")
print("\t 2. Productos por reseña en el servicio")
print("\t 3. Total de ingresos")
print("\t 4. Salir")
print("***71)
# Se lee la opción del usuario
option = input("Opción: ")
print(limpieza_pantalla)

```

Primero, se verifica que la opción que se haya elegido sea numérica, para continuar con los submenús (este fragmento se aplica a los submenús también).

```

# Se verifica que haya sido un dígito el ingresado
if option.isdigit():
    option = int(option)

```

En caso de no ser así se informa al usuario (este fragmento se aplica a los submenús también).

```

else:
    print("\n\t Ingresaste un valor incorrecto, la opción debe ser numérica\n")

```

En caso de seleccionar una opción numérica que no se encuentre en el menú mostrado, se le informa al usuario (este fragmento se aplica a los submenús también).

2.18. Submenús

Para cada uno de estos, se dirige al que corresponda según el valor que se elija en el menú principal.

También en estos se repiten fragmentos que se encuentran en el código para el menú principal.

Estos también cuentan con una opción que redirige al usuario al menú principal. Por cada opción de listado seleccionada, imprime el respectivo listado con un formato de tabla.

3. Solución al problema

Para la solución del problema, primero se van a consultar las tablas que el programa genera. **Se recomienda tener la consola en pantalla completa para una mejor visualización.**

3.1. Productos más vendidos y productos rezagados

3.1.1. Listado de los 15 productos más vendidos

15 productos más vendidos	
Producto	Ventas
SSD Kingston A400, 120GB, SATA III, 2.5'', 7mm	49
Procesador AMD Ryzen 5 2600, S-AM4, 3.40GHz, Six-Core, 16MB L3 Cache, con Disipador Wraith Stealth	42
Procesador Intel Core i3-9100F, S-1151, 3.60GHz, Quad-Core, 6MB Cache (9na. Generación - Coffee Lake)	20
Tarjeta Madre ASRock Micro ATX B450M Steel Legend, S-AM4, AMD B450, HDMI, 64GB DDR4 para AMD	18
SSD Adata Ultimate SU800, 256GB, SATA III, 2.5'', 7mm	15
Procesador AMD Ryzen 3 3200G con Gráficos Radeon Vega 8, S-AM4, 3.60GHz, Quad-Core, 4MB L3, con Disipador Wraith Spire	13
Tarjeta Madre ASUS micro ATX TUF B450M-PLUS GAMING, S-AM4, AMD B450, HDMI, 64GB DDR4 para AMD	13
Procesador AMD Ryzen 5 3600, S-AM4, 3.60GHz, 32MB L3 Cache, con Disipador Wraith Stealth	12
SSD XPG SX8200 Pro, 256GB, PCI Express, M.2	11
Tarjeta de Video ASUS NVIDIA GeForce GTX 1660 SUPER EVO OC, 6GB 192-bit GDDR6, PCI Express x16 3.0	9
SSD Kingston A2000 NVMe, 1TB, PCI Express 3.0, M2	9
Procesador Intel Core i7-9700K, S-1151, 3.60GHz, 8-Core, 12MB Smart Cache (9na. Generación Coffee Lake)	7
Tarjeta Madre MSI ATX B450 TOMAHAWK MAX, S-AM4, AMD B450, 64GB DDR4 para AMD	6
Tarjeta de Video Gigabyte NVIDIA GeForce GT 1030, 2GB 64-bit GDDR5, PCI Express x16 3.0	5
Procesador Intel Core i5-9600K, S-1151, 3.70GHz, Six-Core, 9MB Smart Cache (9na. Generación - Coffee Lake)	4

Se observa que los productos más vendidos en la tienda son procesadores a excepción de un par de discos duros y tarjetas madre, esto nos indica que tal vez, la tienda debería centrarse en los procesadores.

3.1.2. Listado de los 20 productos más buscados

20 productos más buscados	
Producto	Búsquedas
SSD Kingston A400, 120GB, SATA III, 2.5'', 7mm	263
SSD Adata Ultimate SU800, 256GB, SATA III, 2.5'', 7mm	107
Tarjeta Madre ASUS micro ATX TUF B450M-PLUS GAMING, S-AM4, AMD B450, HDMI, 64GB DDR4 para AMD	60
Procesador AMD Ryzen 5 2600, S-AM4, 3.40GHz, Six-Core, 16MB L3 Cache, con Disipador Wraith Stealth	55
Procesador AMD Ryzen 3 3200G con Gráficos Radeon Vega 8, S-AM4, 3.60GHz, Quad-Core, 4MB L3, con Disipador Wraith Spire	41
Logitech Audífonos Gamer G635 7.1, Alámbrico, 1.5 Metros, 3.5mm, Negro/Azul	35
TV Monitor LED 24TL520S-PU 24, HD, Widescreen, HDMI, Negro	32
Procesador Intel Core i7-9700K, S-1151, 3.60GHz, 8-Core, 12MB Smart Cache (9na. Generación Coffee Lake)	31
Procesador Intel Core i3-9100F, S-1151, 3.60GHz, Quad-Core, 6MB Cache (9na. Generación - Coffee Lake)	30
SSD XPG SX8200 Pro, 256GB, PCI Express, M.2	30
SSD Kingston A2000 NVMe, 1TB, PCI Express 3.0, M2	27
Tarjeta Madre MSI ATX B450 TOMAHAWK MAX, S-AM4, AMD B450, 64GB DDR4 para AMD	25
Procesador AMD Ryzen 5 3600, S-AM4, 3.60GHz, 32MB L3 Cache, con Disipador Wraith Stealth	24
Tarjeta Madre ASRock Micro ATX B450M Steel Legend, S-AM4, AMD B450, HDMI, 64GB DDR4 para AMD	23
Procesador Intel Core i5-9600K, S-1151, 3.70GHz, Six-Core, 9MB Smart Cache (9na. Generación - Coffee Lake)	20
Tarjeta de Video ASUS NVIDIA GeForce GTX 1660 SUPER EVO OC, 6GB 192-bit GDDR6, PCI Express x16 3.0	15
Tarjeta de Video MSI AMD Mech Radeon RX 5500 XT MECH Gaming OC, 8GB 128-bit GDDR6, PCI Express 4.0	15
TCL Smart TV LED 55S425 54.6, 4K Ultra HD, Widescreen, Negro	15
Tarjeta de Video Gigabyte NVIDIA GeForce GT 1030, 2GB 64-bit GDDR5, PCI Express x16 3.0	11
SSD Kingston UV500, 480GB, SATA III, mSATA	11

Una vez más, vemos que los que más esta presente en la lista son los procesadores, sin embargo, nos percatamos que el producto más buscado es el más vendido de la tienda. Esto nos dice que podemos colocarlo como producto estrella de la tienda.

3.1.3. Listado de los 5 productos con menos ventas por categoría

5 productos con mas ventas por categoria	
Categoria: procesadores	
Producto	Ventas
Procesador Intel Core i3-8100, S-1151, 3.60GHz, Quad-Core, 6MB Smart Cache (8va. Generación - Coffee Lake)	0
Procesador AMD Ryzen 3 3300X S-AM4, 3.80GHz, Quad-Core, 16MB L2 Cache	2
Procesador Intel Core i9-9900K, S-1151, 3.60GHz, 8-Core, 16MB Smart Cache (9na. Generación Coffee Lake)	3
Procesador Intel Core i5-9600K, S-1151, 3.70GHz, Six-Core, 9MB Smart Cache (9na. Generación - Coffee Lake)	4
Procesador Intel Core i7-9700K, S-1151, 3.60GHz, 8-Core, 12MB Smart Cache (9na. Generación Coffee Lake)	7
Categoria: tarjetas de video	
Producto	Ventas
Tarjeta de Video EVGA NVIDIA GeForce GT 710, 2GB 64-bit GDDR3, PCI Express 2.0	0
Tarjeta de Video EVGA NVIDIA GeForce GTX 1660 Ti SC Ultra Gaming, 6GB 192-bit GDDR6, PCI 3.0	0
Tarjeta de Video EVGA NVIDIA GeForce RTX 2060 SC ULTRA Gaming, 6GB 192-bit GDDR6, PCI Express 3.0	0
Tarjeta de Video Gigabyte AMD Radeon R7 370 OC, 2GB 256-bit GDDR5, PCI Express 3.0	0
Tarjeta de Video Gigabyte NVIDIA GeForce GTX 1650 OC Low Profile, 4GB 128-bit GDDR5, PCI Express 3.0 x16	0
Categoria: tarjetas madre	
Producto	Ventas
Tarjeta Madre AORUS ATX Z390 ELITE, S-1151, Intel Z390, HDMI, 64GB DDR4 para Intel	0
Tarjeta Madre ASRock Z390 Phantom Gaming 4, S-1151, Intel Z390, HDMI, 64GB DDR4 para Intel	0
Tarjeta Madre ASUS ATX ROG STRIX B550-F GAMING WI-FI, S-AM4, AMD B550, HDMI, max. 128GB DDR4 para AMD	0
Tarjeta Madre Gigabyte micro ATX Z390 M GAMING, S-1151, Intel Z390, HDMI, 64GB DDR4 para Intel	0
Tarjeta Madre Gigabyte micro ATX Z490M GAMING X (rev. 1.0), Intel Z490, HDMI, 128GB DDR4 para Intel	0
Categoria: discos duros	
Producto	Ventas
SSD Addlink Technology S70, 512GB, PCI Express 3.0, M.2	0
SSD para Servidor Supermicro SSD-DM128-SMCMVN1, 128GB, SATA III, mSATA, 6Gbit/s	0
SSD para Servidor Lenovo Thinksystem S4500, 480GB, SATA III, 3.5'', 7mm	0
SSD para Servidor Lenovo Thinksystem S4510, 480GB, SATA III, 2.5'', 7mm	0
SSD Samsung 860 EVO, 1TB, SATA III, M.2	0

Vemos que se tienen bastantes productos con ventas en cero en todas las categorías, esto es un indicativo que deberíamos deshacernos de dichos productos de alguna forma, ya que solo nos están generando espacio en el stock y estos podrían ser reemplazados por los productos que más se venden en la tienda. Una forma de deshacernos de los productos sin perder dinero podría ser rematándolos ya que si pasamos más tiempos manteniéndolos en stock estos se depreciarán con el tiempo.

También nos podemos percatar que las memorias usb no generan muchas ventas, por lo que más conveniente sería retirar la categoría de la tienda.

Categoria: memorias usb	
Producto	Ventas
Kit Memoria RAM Corsair Vengeance LPX DDR4, 2400MHz, 32GB, Non-ECC, CL16	0
Kit Memoria RAM Corsair Dominator Platinum DDR4, 3200MHz, 16GB (2x 8GB), Non-ECC, CL16, XMP	1
Categoria: pantallas	
Producto	Ventas
Makena Smart TV LED 32S2 32'', HD, Widescreen, Gris	0
Seiki TV LED SC-39HS950N 38.5, HD, Widescreen, Negro	0
Samsung TV LED LH43QMREBGCXGO 43, 4K Ultra HD, Widescreen, Negro	0
Samsung Smart TV LED UN70RU7100FXZX 70, 4K Ultra HD, Widescreen, Negro	0
Makena Smart TV LED 40S2 40'', Full HD, Widescreen, Negro	0
Categoria: bocinas	
Producto	Ventas
Lenovo Barra de Sonido, Alámbrico, 2.5W, USB, Negro	0
Acetek Bocina con Subwoofer AXF-290, Bluetooth, Inalámbrico, 2.1, 18W RMS, 180W PMPO, USB, Negro	0
Verbatim Bocina Portátil Mini, Bluetooth, Inalámbrico, 3W RMS, USB, Blanco	0
Ghia Bocina Portátil BX300, Bluetooth, Inalámbrico, 40W RMS, USB, Rojo - Resistente al Agua	0
Naceb Bocina Portátil NA-0301, Bluetooth, Inalámbrico, USB 2.0, Rojo	0
Categoria: audifonos	
Producto	Ventas
ASUS Audifonos Gamer ROG Theta 7.1, Alámbrico, USB C, Negro	0
Acer Audifonos Gamer Galea 300, Alámbrico, 3.5mm, Negro	0
Audifonos Gamer Balam Rush Orphix RGB 7.1, Alámbrico, USB, Negro	0
Energy Sistem Audifonos con Micrófono Headphones 1, Bluetooth, Inalámbrico, Negro/Grafito	0
Genius GHP-400S Audifonos, Alámbrico, 1.5 Metros, Rosa	0

3.1.4. Listado de los 20 productos con menos búsquedas por categoría

20 productos con mas búsquedas por categoria	
Categoria: tarjetas de video	
Producto	Búsquedas
Tarjeta de Video EVGA NVIDIA GeForce GT 710, 2GB 64-bit GDDR3, PCI Express 2.0	0
Tarjeta de Video EVGA NVIDIA GeForce RTX 2060 SC ULTRA Gaming, 6GB 192-bit GDDR6, PCI Express 3.0	0
Tarjeta de Video Gigabyte NVIDIA GeForce GTX 1650 OC Low Profile, 4GB 128-bit GDDR5, PCI Express 3.0 x16	0
Tarjeta de Video Gigabyte NVIDIA GeForce RTX 2060 SUPER WINDFORCE OC, 8 GB 256 bit GDDR6, PCI Express x16 3.0	0
Tarjeta de Video MSI Radeon X1550, 128MB 64 bit GDDR2, PCI Express x16	0
Tarjeta de Video PNY NVIDIA GeForce RTX 2080, 8GB 256-bit GDDR6, PCI Express 3.0	0
MSI GeForce 210, 1GB GDDR3, DVI, VGA, HDCP, PCI Express 2.0	1
Tarjeta de Video VisionTek AMD Radeon HD5450, 2GB GDDR3, PCI Express x16	1
Tarjeta de Video Asus NVIDIA GeForce GTX 1050 Ti Phoenix, 4GB 128-bit GDDR5, PCI Express 3.0	2
Tarjeta de Video Gigabyte AMD Radeon R7 370 OC, 2GB 256-bit GDDR5, PCI Express 3.0	3
Tarjeta de Video EVGA NVIDIA GeForce GTX 1660 Ti SC Ultra Gaming, 6GB 192-bit GDDR6, PCI 3.0	4
Tarjeta de Video ASUS AMD Radeon RX 570, 4GB 256-bit GDDR5, PCI Express 3.0	5
Tarjeta de Video MSI NVIDIA GeForce GTX 1050 Ti OC, 4GB 128-bit GDDR5, PCI Express x16 3.0	5
Tarjeta de Video VisionTek AMD Radeon HD 5450, 1GB DDR3, PCI Express x16 2.1	5
Tarjeta de Video Zotac NVIDIA GeForce GTX 1660 Ti, 6GB 192-bit GDDR6, PCI Express x16 3.0	5
Tarjeta de Video Sapphire AMD Pulse Radeon RX 5500 XT Gaming, 8GB 128-bit GDDR6, PCI Express 4.0	10
Tarjeta de Video Gigabyte NVIDIA GeForce GT 1030, 2GB 64-bit GDDR5, PCI Express x16 3.0	11
Tarjeta de Video ASUS NVIDIA GeForce GTX 1660 SUPER EVO OC, 6GB 192-bit GDDR6, PCI Express x16 3.0	15
Tarjeta de Video MSI AMD Mech Radeon RX 5500 XT MECH Gaming OC, 8GB 128-bit GDDR6, PCI Express 4.0	15

Categoria: tarjetas madre	
Producto	Búsquedas
Tarjeta Madre AORUS ATX Z390 ELITE, S-1151, Intel Z390, HDMI, 64GB DDR4 para Intel	0
Tarjeta Madre ASRock Z390 Phantom Gaming 4, S-1151, Intel Z390, HDMI, 64GB DDR4 para Intel	0
Tarjeta Madre ASUS ATX PRIME Z390-A, S-1151, Intel Z390, HDMI, 64GB DDR4 para Intel	0
Tarjeta Madre ASUS ATX ROG STRIX B550-F GAMING WI-FI, S-AM4, AMD B550, HDMI, max. 128GB DDR4 para AMD	0
Tarjeta Madre Gigabyte micro ATX Z490M GAMING X (rev. 1.0), Intel Z490, HDMI, 128GB DDR4 para Intel	0
Tarjeta Madre ASRock ATX Z490 STEEL LEGEND, S-1200, Intel Z490, HDMI, 128GB DDR4 para Intel	0
Tarjeta Madre Gigabyte Micro ATX H310M DS2 2.0, S-1151, Intel H310, 32GB DDR4 para Intel	0
Tarjeta Madre ASUS micro ATX Prime H370M-Plus/CSM, S-1151, Intel H370, HDMI, 64GB DDR4 para Intel	0
Tarjeta Madre ASUS ATX ROG STRIX Z390-E GAMING, S-1151, Intel Z390, HDMI, 64GB DDR4 para Intel	0
Tarjeta Madre Gigabyte micro ATX Z390 M GAMING, S-1151, Intel Z390, HDMI, 64GB DDR4 para Intel	1
Tarjeta Madre ASRock ATX H110 Pro BTC+, S-1151, Intel H110, 32GB DDR4, para Intel	1
ASUS T. Madre uATX M4A88T-M, S-AM3, DDR3 para Phenom II/Athlon II/Sempron 100	3
Tarjeta Madre Gigabyte micro ATX GA-H110M-DS2, S-1151, Intel H110, 32GB DDR4 para Intel	4
Tarjeta Madre AORUS micro ATX B450 AORUS M (rev. 1.0), S-AM4, AMD B450, HDMI, 64GB DDR4 para AMD	10
Tarjeta Madre Gigabyte XL-ATX TRX40 Designare, S-sTRX4, AMD TRX40, 256GB DDR4 para AMD	10
Tarjeta Madre ASRock Micro ATX B450M Steel Legend, S-AM4, AMD B450, HDMI, 64GB DDR4 para AMD	23
Tarjeta Madre MSI ATX B450 TOMAHAWK MAX, S-AM4, AMD B450, 64GB DDR4 para AMD	25
Tarjeta Madre ASUS micro ATX TUF B450M-PLUS GAMING, S-AM4, AMD B450, HDMI, 64GB DDR4 para AMD	60

Categoria: discos duros	
Producto	Búsquedas
SSD Addlink Technology S70, 512GB, PCI Express 3.0, M.2	0
SSD para Servidor Supermicro SSD-DH128-SMCMVN1, 128GB, SATA III, mSATA, 6Gbit/s	0
SSD para Servidor Lenovo Thinksystem S4510, 480GB, SATA III, 2.5'', 7mm	0
SSD Samsung 860 EVO, 1TB, SATA III, M.2	1
SSD para Servidor Lenovo Thinksystem S4500, 480GB, SATA III, 3.5'', 7mm	2
SSD Western Digital WD Blue 3D NAND, 2TB, M.2	5
SSD Crucial MX500, 1TB, SATA III, M.2	7
Kit SSD Kingston KC600, 1TB, SATA III, 2.5, 7mm	10
SSD Kingston UV500, 480GB, SATA III, mSATA	11
SSD Kingston A2000 NVMe, 1TB, PCI Express 3.0, M2	27
SSD XPG SX8200 Pro, 256GB, PCI Express, M.2	30
SSD Adata Ultimate SU800, 256GB, SATA III, 2.5'', 7mm	107
SSD Kingston A400, 120GB, SATA III, 2.5'', 7mm	263

Categoria: memorias usb	
Producto	Búsquedas
Kit Memoria RAM Corsair Dominator Platinum DDR4, 3200MHz, 16GB (2x 8GB), Non-ECC, CL16, XMP	0
Kit Memoria RAM Corsair Vengeance LPX DDR4, 2400MHz, 32GB, Non-ECC, CL16	0

Categoria: pantallas	
Producto	Búsquedas
Makena Smart TV LED 32S2 32'', HD, Widescreen, Gris	0
Samsung TV LED LH43QMREBGCXGO 43, 4K Ultra HD, Widescreen, Negro	0
Samsung Smart TV LED UN70RU7100FXZX 70, 4K Ultra HD, Widescreen, Negro	0
Makena Smart TV LED 40S2 40'', Full HD, Widescreen, Negro	0
Hisense Smart TV LED 40H5500F 39.5, Full HD, Widescreen, Negro	0
Samsung Smart TV LED UN32J4290AF 32, HD, Widescreen, Negro	0
Hisense Smart TV LED 50H8F 49.5, 4K Ultra HD, Widescreen, Negro	0
Samsung Smart TV LED 43, Full HD, Widescreen, Negro	1
Seiki TV LED SC-39HS950N 38.5, HD, Widescreen, Negro	4
Samsung Smart TV LED UNS5TU7000FXZX 55, 4K Ultra HD, Widescreen, Negro/Gris	4
TCL Smart TV LED 55S425 54.6, 4K Ultra HD, Widescreen, Negro	15
TV Monitor LED 24TL520S-PU 24, HD, Widescreen, HDMI, Negro	32

Categoria: bocinas	
Producto	Búsquedas
Lenovo Barra de Sonido, Alámbrico, 2.5W, USB, Negro	0
Verbatim Bocina Portátil Mini, Bluetooth, Inalámbrico, 3W RMS, USB, Blanco	0
Ghia Bocina Portátil BX300, Bluetooth, Inalámbrico, 40W RMS, USB, Rojo - Resistente al Agua	0
Naceb Bocina Portátil NA-0301, Bluetooth, Inalámbrico, USB 2.0, Rojo	0
Ghia Bocina Portátil BX900, Bluetooth, Inalámbrico, 2.1 Canales, 34W, USB, Negro - Resistente al Agua	0
Ghia Bocina Portátil BX400, Bluetooth, Inalámbrico, 8W RMS, USB, Negro	0
Ghia Bocina Portátil BX500, Bluetooth, Inalámbrico, 10W RMS, USB, Gris	0
Ghia Bocina Portátil BX800, Bluetooth, Inalámbrico, 2.1 Canales, 31W, USB, Negro	1
Acteck Bocina con Subwoofer AXF-290, Bluetooth, Inalámbrico, 2.1, 18W RMS, 180W PMPO, USB, Negro	2
Logitech Bocinas para Computadora con Subwoofer G560, Bluetooth, Inalámbrico, 2.1, 120W RMS, USB, negro	6

Categoria: audifonos	
Producto	Búsquedas
ASUS Audifonos Gamer ROG Theta 7.1, Alámbrico, USB C, Negro	0
Acer Audifonos Gamer Galea 300, Alámbrico, 3.5mm, Negro	0
Audifonos Gamer Balam Rush Orphix RGB 7.1, Alámbrico, USB, Negro	0
Energy Sistem Audifonos con Micrófono Headphones 1, Bluetooth, Inalámbrico, Negro/Grafito	0
Getttech Audifonos con Micrófono Sonority, Alámbrico, 1.2 Metros, 3.5mm, Negro/Rosa	0
Klip Xtreme Audifonos Blast, Bluetooth, Inalámbrico, Negro/Verde	0
Ginga Audifonos con Micrófono GI18ADJ01BT-RO, Bluetooth, Alámbrico/Inalámbrico, 3.5mm, Rojo	1
Genius GHP-400S Audifonos, Alámbrico, 1.5 Metros, Rosa	2
Iogear Audifonos Gamer GHG601, Alámbrico, 1.2 Metros, 3.5mm, Negro	3
HyperX Audifonos Gamer Cloud Flight para PC/PS4/PS4 Pro, Inalámbrico, USB, 3.5mm, Negro	6
Cougar Audifonos Gamer Phontum Essential, Alámbrico, 1.9 Metros, 3.5mm, Negro.	7
Logitech Audifonos Gamer G332, Alámbrico, 2 Metros, 3.5mm, Negro/Rojo	10
Logitech Audifonos Gamer G635 7.1, Alámbrico, 1.5 Metros, 3.5mm, Negro/Azul	35

Categoria: procesadores	
Producto	Búsquedas
Procesador Intel Core i3-8100, S-1151, 3.60GHz, Quad-Core, 6MB Smart Cache (8va. Generación - Coffee Lake)	1
Procesador AMD Ryzen 3 3300X S-AM4, 3.80GHz, Quad-Core, 16MB L2 Cache	10
Procesador Intel Core i9-9900K, S-1151, 3.60GHz, 8-Core, 16MB Smart Cache (9na. Generación Coffee Lake)	10
Procesador Intel Core i5-9600K, S-1151, 3.70GHz, Six-Core, 9MB Smart Cache (9na. Generación - Coffee Lake)	20
Procesador AMD Ryzen 5 3600, S-AM4, 3.60GHz, 32MB L3 Cache, con Disipador Wraith Stealth	24
Procesador Intel Core i3-9100F, S-1151, 3.60GHz, Quad-Core, 6MB Cache (9na. Generación - Coffee Lake)	30
Procesador Intel Core i7-9700K, S-1151, 3.60GHz, 8-Core, 12MB Smart Cache (9na. Generación Coffee Lake)	31
Procesador AMD Ryzen 3 3200G con Gráficos Radeon Vega 8, S-AM4, 3.60GHz, Quad-Core, 4MB L3, con Disipador Wraith Spire	41
Procesador AMD Ryzen 5 2600, S-AM4, 3.40GHz, Six-Core, 16MB L3 Cache, con Disipador Wraith Stealth	55

Gracia al listado notamos que como en los productos con menores ventas, se encuentran productos que ni siquiera generan una búsqueda por lo que deberían ser retirados del stock.

Para esto podríamos utilizar la misma estrategia que se planteó en el punto anterior. También notamos que los procesadores, son los más buscados en la tienda por lo que nos deberíamos centrar en esa categoría.

3.2. Productos por reseña en el servicio

3.2.1. Listado de los 20 productos con mejores reseñas

20 productos con mejores reseñas	
Producto	Score
Procesador AMD Ryzen 3 3300X S-AM4, 3.80GHz, Quad-Core, 16MB L2 Cache	5.00
Procesador Intel Core i9-9900K, S-1151, 3.60GHz, 8-Core, 16MB Smart Cache (9na. Generación Coffee Lake)	5.00
Procesador Intel Core i7-9700K, S-1151, 3.60GHz, 8-Core, 12MB Smart Cache (9na. Generación Coffee Lake)	5.00
Procesador Intel Core i5-9600K, S-1151, 3.70GHz, Six-Core, 9MB Smart Cache (9na. Generación - Coffee Lake)	5.00
Tarjeta de Video ASUS AMD Radeon RX 570, 4GB 256-bit GDDR5, PCI Express 3.0	5.00
Tarjeta de Video MSI AMD Mech Radeon RX 5500 XT MECH Gaming OC, 8GB 128-bit GDDR6, PCI Express 4.0	5.00
Tarjeta de Video MSI NVIDIA GeForce GTX 1050 Ti OC, 4GB 128-bit GDDR5, PCI Express x16 3.0	5.00
Tarjeta de Video Sapphire AMD Pulse Radeon RX 5500 XT Gaming, 8GB 128-bit GDDR6, PCI Express 4.0	5.00
Tarjeta de Video Zotac NVIDIA GeForce GTX 1660 Ti, 6GB 192-bit GDDR6, PCI Express x16 3.0	5.00
Tarjeta Madre Gigabyte XL-ATX TRX40 Designare, S-sTRX4, AMD TRX40, 256GB DDR4 para AMD	5.00
Kit SSD Kingston KC600, 1TB, SATA III, 2.5, 7mm	5.00
SSD Crucial MX500, 1TB, SATA III, M.2	5.00
SSD Western Digital WD Blue 3D NAND, 2TB, M.2	5.00
Kit Memoria RAM Corsair Dominator Platinum DDR4, 3200MHz, 16GB (2x 8GB), Non-ECC, CL16, XMP	5.00
TCL Smart TV LED 55S425 54.6, 4K Ultra HD, Widescreen, Negro	5.00
TV Monitor LED 24TL5205-PU 24, HD, Widescreen, HDMI, Negro	5.00
Logitech Audifonos Gamer G332, Alámbrico, 2 Metros, 3.5mm, Negro/Rojo	5.00
Logitech Audifonos Gamer G635 7.1, Alámbrico, 1.5 Metros, 3.5mm, Negro/Azul	5.00
SSD Adata Ultimate SU800, 256GB, SATA III, 2.5'', 7mm	4.87
Procesador AMD Ryzen 5 2600, S-AM4, 3.40GHz, Six-Core, 16MB L3 Cache, con Disipador Wraith Stealth	4.81

Una vez más los procesadores son los que destacan entre nuestros clientes, por lo que se suma aun punto más para centrarse en esa categoría. Sin embargo, También hay varias tarjetas madre y discos duros que se encuentran entre los más vendidos que les están dejando un buen sabor de boca a los clientes, por lo que también deberíamos de centrarnos en esos productos.

3.2.2. Listado de los 20 productos con peores reseñas

20 productos con peores reseñas	
Producto	Score
Tarjeta de Video Gigabyte AMD Radeon R7 370 OC, 2GB 256-bit GDDR5, PCI Express 3.0	1.00
Tarjeta Madre ASRock ATX H110 Pro BTC+, S-1151, Intel H110, 32GB DDR4, para Intel	1.00
Tarjeta Madre AORUS micro ATX B450 AORUS M (rev. 1.0), S-AM4, AMD B450, HDMI, 64GB DDR4 para AMD	1.83
Tarjeta Madre Gigabyte micro ATX GA-H110M-DS2, S-1151, Intel H110, 32GB DDR4 para Intel	2.00
Cougar Audifonos Gamer Phontum Essential, Alámbrico, 1.9 Metros, 3.5mm, Negro.	3.00
MSI GeForce 210, 1GB GDDR3, DVI, VGA, HDCP, PCI Express 2.0	4.00
Tarjeta de Video Asus NVIDIA GeForce GTX 1050 Ti Phoenix, 4GB 128-bit GDDR5, PCI Express 3.0	4.00
HyperX Audifonos Gamer Cloud Flight para PC/PS4/PS4 Pro, Inalámbrico, USB, 3.5mm, Negro	4.00
Tarjeta Madre ASUS micro ATX TUF B450M-PLUS GAMING, S-AM4, AMD B450, HDMI, 64GB DDR4 para AMD	4.14
Procesador AMD Ryzen 5 3600, S-AM4, 3.60GHz, 32MB L3 Cache, con Disipador Wraith Stealth	4.23
Tarjeta de Video Gigabyte NVIDIA GeForce GT 1030, 2GB 64-bit GDDR5, PCI Express x16 3.0	4.40
Procesador AMD Ryzen 3 3200G con Gráficos Radeon Vega 8, S-AM4, 3.60GHz, Quad-Core, 4MB L3, con Disipador Wraith Spire	4.46
Tarjeta Madre ASUS ATX PRIME Z390-A, S-1151, Intel Z390, HDMI, 64GB DDR4 para Intel	4.50
Logitech Bocinas para Computadora con Subwoofer G560, Bluetooth, Inalámbrico, 2.1, 120W RMS, USB, negro	4.50
SSD XPG SX8200 Pro, 256GB, PCI Express, M.2	4.55
Tarjeta Madre ASRock Micro ATX B450M Steel Legend, S-AM4, AMD B450, HDMI, 64GB DDR4 para AMD	4.56
Tarjeta Madre MSI ATX B450 TOMAHAWK MAX, S-AM4, AMD B450, 64GB DDR4 para AMD	4.67
SSD Kingston A2000 NVMe, 1TB, PCI Express 3.0, M2	4.67
SSD Kingston UV500, 480GB, SATA III, mSATA	4.67
Procesador Intel Core i3-9100F, S-1151, 3.60GHz, Quad-Core, 6MB Cache (9na. Generación - Coffee Lake)	4.70

Hay tarjetas madre que han resultados pésimas para los clientes, por lo que habría que considerar quitarlas del stock. Por lo demás, los productos de arriba de 4 en el score, podrían continuar manteniéndose en el stock.

3.3. Total de ingresos

3.3.1. Total de ingresos mensuales

Total de ingresos mensuales		
Año: 2020		
Mes	Ingresos	Porcentaje anual (%)
Enero	117738	15.96
Febrero	107270	14.54
Marzo	162931	22.08
Abril	191066	25.89
Mayo	91936	12.46
Junio	36949	5.01
Julio	26949	3.65
Agosto	3077	0.42
Septiembre	0	0.00
Octubre	0	0.00
Noviembre	0	0.00
Diciembre	0	0.00

A partir del mes de agosto los ingresos son mínimos, por lo que, se debe considerar no reabastecerse de grandes cantidades de stock para esos meses o bien plantear algunos descuentos para aumentar el flujo de ingresos durante esos meses. También. Se ve que, entre marzo y abril, se concentra más de la mitad de los ingresos, por lo que el abastecimiento de stock durante esos meses se debe mantener constante o incluso, aumentar.

3.3.2. Ventas promedio mensuales

Como era de esperarse, se replica el comportamiento de los ingresos en las ventas a diferencia de que en enero se concentra un volumen considerable de ventas, por lo que junto con abril y marzo, debe mantener el abastecimiento de stock.

Ventas promedio mensuales		
Año: 2020		
Mes	Ventas	Porcentaje anual (%)
Enero	52	18.98
Febrero	40	14.60
Marzo	49	17.88
Abril	74	27.01
Mayo	34	12.41
Junio	11	4.01
Julio	11	4.01
Agosto	3	1.09
Septiembre	0	0.00
Octubre	0	0.00
Noviembre	0	0.00
Diciembre	0	0.00

3.3. Total Anual

Total anual	
Año: 2020	
Total de ingresos anual	Total de ventas anuales
737916	274

3.4. Meses con más ventas al año

Meses con más ventas		
Año: 2020		
Mes	Ingresos	Porcentaje anual (%)
Abril	74	27.01
Enero	52	18.98
Marzo	49	17.88
Febrero	40	14.60
Mayo	34	12.41

Como se había observado en puntos anteriores, abril, enero y marzo son los meses con mayores ventas.

4. Conclusiones

Vemos que Python es una herramienta muy poderosa de análisis de datos, no importa de donde provengan estos ya que podemos desde darle el formato que sea necesario a datos en crudo hasta tomar datos que están en una base de datos, esto con el fin de darle una interpretación que nos puede llegar a la resolución de problemas o bien, al descubrimiento de hechos que no eran visibles a simple vista.

También cabe mencionar que no se hizo uso de ninguna librería extra para la realización del proyecto, así que el análisis que se hizo aquí se puede mejorar de forma exponencial si se llega a replicar apoyándose de alguna librería para mostrar de una forma más agradable los datos o bien para el propio análisis.

En lo personal, me dejo muy satisfecho el trabajo que realice y espero poder seguir mejorando para llevar mis conocimientos aún más lejos.