



**EDUARDO ENRIQUE  
CERVANTES VAZQUEZ**

---

**REPORTE DE  
PROYECTO:  
LifeStore**

Lunes, 7 de febrero de 2022.

## Contenido

1.	Introducción .....	2
2.	Definición del código .....	2
2.1.	Bibliotecas .....	2
2.2	Definición de usuario:.....	3
2.3.	Colores de la terminal .....	3
2.4.	Inicio de sesión .....	3
2.4.1	Inicio de sesión denegado .....	4
2.4.2.	Inicio de sesión exitoso.....	5
2.5.	Opciones de visualización.....	5
2.6.	Productos por reseña en el servicio .....	6
2.7.	Funciones .....	7
2.7.1	Productos más vendidos.....	7
2.7.2.	Productos menos vendidos .....	8
2.7.3	Productos más buscados .....	9
2.7.4.	Productos menos buscados .....	10
2.7.5.	Productos con peores reseñas.....	12
2.7.6.	Productos con las peores reseñas .....	13
2.7.7.	Total, de ventas realizadas .....	14
2.7.8.	Ventas por Mes.....	14
2.7.9.	Entrada de teclado no valida .....	16
3.	GitHub .....	16
4.	Gráficos .....	16
4.1.	Artículos más vendidos.....	17
4.2.	Artículos menos vendidos .....	18
4.3.	Artículos más buscados .....	19
4.4.	Artículos menos buscados.....	20
4.5.	Ventas en artículos .....	21
4.6.	Ingresos .....	21
5.	Conclusión .....	22

## 1. Introducción

La ciencia de datos, es una tecnología emergente que requiere de habilidades en el manejo de las tecnologías de la información, antes podías realizar una estadística o interpretación de los datos obtenidos con ayuda de una hoja de cálculo y un operador, hoy en día los volúmenes de datos sobrepasan la capacidad que permitiría dar eficiencia al operador, por lo la ciencia de datos evoluciona, surgiendo y proponiendo una especialidad en su análisis, a este tipo de operador le llamaremos “Analista de datos”, cuyas habilidades no solo serán el manejo de una computadora, sino que deberá requerir de conocimientos en un lenguaje de programación, para este reporte, el lenguaje de programación utilizado por el estudiante de análisis de datos es Python, un lenguaje de programación interpretado cuyo origen hace hincapié en la facilidad de lectura e interpretación de su código. siendo un lenguaje de alto nivel, cuenta con una biblioteca muy extensa y con una compatibilidad entre todos los sistemas operativos existentes.

Tomando en cuenta las habilidades adquiridas durante la parte uno del curso, se propone dar solución a “LifeStore” una tienda virtual con amplia gama de artículos tecnológicos, cuya propuesta es desarrollar para ellos un sistema de análisis basados en los datos de su inventario. donde se incluya un sistema que permita el acceso a dicha información solo al ingresar con un usuario y contraseña.

## 2. Definición del código

### 2.1. Bibliotecas

```
from lifestore_file import lifestore_searches, lifestore_sales,
lifestore_products
import collections
from colorama import Style, Back, Fore
import getpass
```

`from lifestore_file import lifestore_searches, lifestore_sales, lifestore_products` proporciona al programa importar las listas y datos almacenados en el archivo “lifestore\_file.py”

El módulo `collections` nos ayuda a completar y manipular las estructuras de datos de una forma más eficiente.

`colorama` es un módulo que permite imprimir textos de colores en la salida de la terminal o consola, incluyendo el fondo o estilo del texto.

`getpass` Es un modulo que proporciona una forma segura de manejar las solicitudes de contraseña donde se interactúa a través de la terminal.

## 2.2 Definición de usuario:

```
5 usuario = 'jimmy'           #Especifique el usuario
6 contrasena = 'ymmij'        #Especifique la contraseña
7 intentos_permitidos= 3      #Especifique la cantidad de intentos que
                             permitirá
```

## 2.3. Colores de la terminal

```
colorin = Fore.GREEN          #Color para Login
negrita = Style.BRIGHT        #Letras en Negrita
clr_reset = Style.RESET_ALL   #Restablecer efectos de color o letra
clr_username = Fore.CYAN      #Color para Usuario
clr_tit = Fore.RED            #Color para titulo de opción seleccionada
clr_sub = Fore.MAGENTA        #Color para sub opción seleccionada
clr_mes = Fore.GREEN          #Color del mes de venta
clr_monto = Fore.RED          #Color de la cantidad vendida
clr_items = Fore.CYAN         #Color del # de items vendidos
backneg = Back.YELLOW         #Color de fondo de etiqueta
fneg = Fore.RED               #Color de letra de la etiqueta negada
```

Utilizando variables declaradas con lo que asignaría al texto mostrado en consola un color de fuente, un tipo de letra(negrita), un fondo al texto o restablecer los valores.

Se declara variable en esta sección del código para asignar un color, sin realizar muchos cambios.

## 2.4. Inicio de sesión

```
estado = False
intentos = 0
```

La variable definida como `estado` le indica al ciclo que no tiene autenticación para permitir el acceso al programa.

La variable `intentos` puede ser modificada para aumentar o disminuir las oportunidades que tendrá el usuario para realizar un intento de sesión.

```
while (estado == False and intentos < intentos_permitidos): #Función de
bucle para inicio de sesión
    print('\n Número de intentos disponibles:', (intentos_permitidos-
intentos), '\n')
    username_input = input(colorin + negrita + 'Ingrese
usuario:'+clr_reset)
    password_input = input(colorin + negrita + 'Ingrese
contraseña:'+clr_reset)

    #Descomentar la linea inferior y comentar la linea anterior en caso
de que no desee que se muestre en consola cuando se teclee la contraseña
```

```

password_input = getpass.getpass(colorin + negrita + 'Ingrese
contraseña:' + clr_reset)

intentos = intentos + 1

if username_input == usuario and password_input == contrasena:
    estado = True
elif username_input != usuario:
    print(negrita + 'Usuario NO encontrado' + clr_reset)
    estado = False
elif username_input == usuario and password_input != contrasena:
    print(negrita + 'La contraseña es incorrecta' + clr_reset)
    estado = False

```

Presentado y desarrollado mediante un ciclo `while` en donde al usuario le muestra en consola:

```

Número de intentos disponibles: 3

Ingrese usuario:
Ingrese contraseña:

```

Fig. 1. Inicio de sesión.

#### 2.4.1 Inicio de sesión denegado

En donde en este caso dispondrá de 3 intentos para intentar iniciar sesión, en caso de introducir un usuario no registrado, en consola se imprimirá:

```

Ingrese usuario:2
Ingrese contraseña:2
Usuario NO encontrado

```

Fig. 2 Usuario NO encontrado.

Donde al notificarle que ese usuario no está registrado le pedirá reintentar iniciar sesión, además de ello, implementa un método de validación donde notifica en caso de tener un usuario correcto pero una contraseña incorrecta:

```

Ingrese usuario:lalo
Ingrese contraseña:a
La contraseña es incorrecta

Número de intentos disponibles: 2

```

Fig. 3 Contraseña incorrecta.

En caso de exceder el número de intentos permitidos, la condicional, se considera “Negada” e imprime una etiqueta para notificarle al usuario.

```

else:
    print(backneg + fneg + 'Intentos de acceso excedidos' + clr_reset)

```

```

Ingrese usuario:
Ingrese contraseña:
Usuario NO encontrado
Intentos de acceso excedidos

```

Fig. 4 intentos de inicio de sesión excedidos.

#### 2.4.2. Inicio de sesión exitoso

```

if estado == True:

    print('Bienvenido de nuevo'+ clr_username,  usuario + clr_reset +',
selecciona la opción que deseas visualizar:' +'''
1) Productos más vendidos y productos rezagados.
2) Productos por reseña en el servicio.
3) Total de ingresos
''')

    seleccion = input('Seleccione la opción que desea visualizar: ')

```

En caso de acreditar el inicio de sesión con las credenciales correctas, nuestra variable `estado` cambiara a un valor verdadero, lo que permitirá comenzar con un ciclo `if - elif - else`, y como primera instancia le mostrara a nuestro operador un mensaje de bienvenida, mostrando su nombre de usuario, así como una serie de opciones disponibles, permitiéndole introducir una entrada por teclado deseada mediante la función `input`

```

Ingrese usuario:lalo
Ingrese contraseña:1
Bienvenido de nuevo lalo, selecciona la opción que deseas visualizar:
1) Productos más vendidos y productos rezagados.
2) Productos por reseña en el servicio.
3) Total de ingresos

Seleccione la opción que desea visualizar: █

```

Fig. 5 Inicio de sesión acreditado.

#### 2.5. Opciones de visualización

Productos más vendidos y productos rezagados:

```

if seleccion == '1':      #Productos más vendidos y productos rezagados
    print(
'Opción seleccionada:' + clr_tit +" Productos más vendidos y productos
rezagados."+clr_reset+'''
1) Producto más vendido.
2) Top 5 de Productos más vendidos.

```

- 3) Top 5 de Productos menos vendidos.
- 4) Top 10 de Productos más buscados.
- 5) Top 10 de Productos menos buscados.

```
'''
seleccion = input('Seleccione la opción que desea visualizar:')
```

```
Ingrese usuario:lalo
Ingrese contraseña:1
Bienvenido de nuevo lalo, selecciona la opción que deseas visualizar:
  1) Productos más vendidos y productos rezagados.
  2) Productos por reseña en el servicio.
  3) Total de ingresos

Seleccione la opción que desea visualizar: 1
Opción seleccionada: Productos más vendidos y productos rezagados.
  1) Producto más vendido.
  2) Top 5 de Productos más vendidos.
  3) Top 5 de Productos menos vendidos.
  4) Top 10 de Productos más buscados.
  5) Top 10 de Productos menos buscados.
```

Fig. 6 Espera de selección, Opción 1.

## 2.6. Productos por reseña en el servicio

```
elif seleccion == '2': #Productos por reseña en el servicio
    print(
'Opción seleccionada:' + clr_tit + "Productos por reseña en el servicio." +
clr_reset + '''
  1) Top 5 con mejores reseñas
  2) Top 5 con las peores reseñas
'''
    seleccion = input('Seleccione la opción que desea visualizar: ')
```

```
Ingrese usuario:lalo
Ingrese contraseña:1
Bienvenido de nuevo lalo, selecciona la opción que deseas visualizar:
  1) Productos más vendidos y productos rezagados.
  2) Productos por reseña en el servicio.
  3) Total de ingresos

Seleccione la opción que desea visualizar: 2
Opción seleccionada:Productos por reseña en el servicio.
  1) Top 5 con mejores reseñas
  2) Top 5 con las peores reseñas

Seleccione la opción que desea visualizar: █
```

Fig. 7 Espera de selección, opción 2.

Ingresos:

```
elif seleccion == '3': #Ingresos
    print('Opción seleccionada:' +clr_tit+"Ingresos."+clr_reset,
        '''
        1) Total de ventas realizadas.
        2) Total de Ingresos.
        '''
    )
```

```
Ingrese usuario:1
Ingrese contraseña:1
Bienvenido de nuevo 1, selecciona la opción que deseas visualizar:
  1) Productos más vendidos y productos rezagados.
  2) Productos por reseña en el servicio.
  3) Total de ingresos

Seleccione la opción que desea visualizar: 3
Opción seleccionada:Ingresos.
  1) Total de ventas realizadas.
  2) Total de Ingresos.

Seleccione la opción que desea visualizar: []
```

Fig. 8 Espera de selección, opción 3.

## 2.7. Funciones

### 2.7.1 Productos más vendidos

```
print(clr_sub+' 5 Productos más vendidos: \n'+ clr_reset)
prod_compr =[]
for val in lifestore_sales: #Funcion para definir un listado
de las ventas
    prod_compr.append(val[1])
    repeticiones = {}

    for n in prod_compr:
        if n in repeticiones :
            repeticiones[n] += 1
        else:
            repeticiones[n] = 0
    primero =
(collections.Counter(repeticiones).most_common()[0][0])-1
    segundo =
(collections.Counter(repeticiones).most_common()[1][0])-1
    tercero =
(collections.Counter(repeticiones).most_common()[2][0])-1
    cuarto =
(collections.Counter(repeticiones).most_common()[3][0])-1
```



```

quinto =
(collections.Counter(repeticiones).most_common()[4][0])-1
print(
    lifestore_products[primero][1], '\n',
    lifestore_products[segundo][1], '\n',
    lifestore_products[tercero][1], '\n',
    lifestore_products[cuarto][1], '\n',
    lifestore_products[quinto][1], '\n'
)

```

La función se basa en tomar un ciclo for para separar determinada columna de la lista “lifestore\_sales” y añadir esos valores en una nueva lista de nombre “prod\_comp”.

Usando otro ciclo for se harán iteraciones de toda esa lista, la cual añadirá los valores y la cantidad de veces que se repita dentro de una lista de nombre “repeticiones”.

Tomando los datos de la lista repeticiones, el programa va a tomar los 5 valores más repetidos, tomando en cuenta que son los más vendidos, posterior a ello y tomando en cuenta que solo tiene el número de lista del artículo, realizara una búsqueda de ese artículo dentro de la lista “lifestore\_products” para posterior mente mostrarlo en pantalla

```

Seleccione la opción que desea visualizar:2
5 Productos más vendidos:

54 SSD Kingston A400, 120GB, SATA III, 2.5'', 7mm 259 discos duros 300
3 Procesador AMD Ryzen 5 2600, S-AM4, 3.40GHz, Six-Core, 16MB L3 Cache, con Disipador Wraith Stealth 3089 procesadores 987
5 Procesador Intel Core i3-9100F, S-1151, 3.60GHz, Quad-Core, 6MB Cache (9na. Generación - Coffee Lake) 1779 procesadores 130
42 Tarjeta Madre ASRock Micro ATX B450M Steel Legend, S-AM4, AMD B450, HDMI, 64GB DDR4 para AMD 1779 tarjetas madre 0
57 SSD Adata Ultimate SU800, 256GB, SATA III, 2.5'', 7mm 889 discos duros 15

```

Fig. 9 Productos más vendidos.

### 2.7.2. Productos menos vendidos

```

print(clr_sub+'5 Productos menos vendidos.\n'+clr_reset)
prod_compr = []
for val in lifestore_sales: #Funcion para definir un listado
de las ventas
    prod_compr.append(val[1])
    repeticiones = {}

    for n in prod_compr:
        if n in repeticiones :
            repeticiones[n] += 1
        else:
            repeticiones[n] = 0
    primero1 = collections.Counter(repeticiones).most_common()[1][0]-1
    segundo1 = collections.Counter(repeticiones).most_common()[2][0]-1
    tercero1 = collections.Counter(repeticiones).most_common()[3][0]-1
    cuarto1 = collections.Counter(repeticiones).most_common()[4][0]-1
    quinto1 = collections.Counter(repeticiones).most_common()[5][0]-1

```

```
print(
    lifestore_products[primerol] [1], '\n',
    lifestore_products[segundol] [1], '\n',
    lifestore_products[tercerol] [1], '\n',
    lifestore_products[cuartol] [1], '\n',
    lifestore_products[quintol] [1], '\n'
)
```

Tomando en cuenta la resolución para los productos más buscados, el programa va a tomar los 5 valores menos repetidos, tomando en cuenta que se usara valores negativos para ubicar su posición, posterior a ello y tomando en cuenta que solo tiene el número de lista del artículo, realizara una búsqueda de ese artículo dentro de la lista "lifestore\_products" para posterior mente mostrarlo en pantalla

```
Seleccione la opción que desea visualizar:3
5 Productos menos vendidos.

94 HyperX Audífonos Gamer Cloud Flight para PC/PS4/PS4 Pro, Inalámbrico, USB, 3.5mm, Negro 2869 audifonos 12
89 Cougar Audífonos Gamer Phontum Essential, Alámbrico, 1.9 Metros, 3.5mm, Negro. 859 audifonos 4
84 Logitech Audífonos Gamer G332, Alámbrico, 2 Metros, 3.5mm, Negro/Rojo 1089 audifonos 83
67 TV Monitor LED 24TL520S-PU 24, HD, Widescreen, HDMI, Negro 3229 pantallas 411
66 TCL Smart TV LED 55S425 54.6, 4K Ultra HD, Widescreen, Negro 8049 pantallas 188
```

Fig. 10 Productos menos vendidos.

### 2.7.3 Productos más buscados

```
print(clr_sub+'10 Productos más buscados \n'+clr_reset)
busquedas = []
for val in lifestore_searches: #Funcion para definir un
listado de las ventas
    busquedas.append(val[1])
    repeticiones = {}

for n in busquedas:
    if n in repeticiones :
        repeticiones[n] += 1
    else:
        repeticiones[n] = 0
    primero =
(collections.Counter(repeticiones).most_common()[0][0])-1
    segundo =
(collections.Counter(repeticiones).most_common()[1][0])-1
    tercero =
(collections.Counter(repeticiones).most_common()[2][0])-1
    cuarto =
(collections.Counter(repeticiones).most_common()[3][0])-1
    quinto =
(collections.Counter(repeticiones).most_common()[4][0])-1
    sexto =
(collections.Counter(repeticiones).most_common()[5][0])-1
    septimo =
(collections.Counter(repeticiones).most_common()[6][0])-1
```

```

        octavo =
(collections.Counter(repeticiones).most_common()[7][0])-1
        noveno =
(collections.Counter(repeticiones).most_common()[8][0])-1
        decimo =
(collections.Counter(repeticiones).most_common()[9][0])-1
    print(
lifestore_products[primero][1], '\n',
lifestore_products[segundo][1], '\n',
lifestore_products[tercero][1], '\n',
lifestore_products[cuarto][1], '\n',
lifestore_products[quinto][1], '\n',
lifestore_products[sexto][1], '\n',
lifestore_products[septimo][1], '\n',
lifestore_products[octavo][1], '\n',
lifestore_products[noveno][1], '\n',
lifestore_products[decimo][1], '\n'
    )

```

La función se basa en tomar un ciclo for para separar determinada columna de la lista “lifestore\_search” y añadir esos valores en una nueva lista de nombre “busquedas”.

Usando otro ciclo for se harán iteraciones de toda esa lista, la cual añadirá los valores y la cantidad de veces que se repita dentro de una lista de nombre “repeticiones”.

Tomando los datos de la lista repeticiones, el programa va a tomar los 5 valores más repetidos, tomando en cuenta que son los más buscados, posterior a ello y tomando en cuenta que solo tiene el número de lista del artículo, realizara una búsqueda de ese artículo dentro de la lista “lifestore\_products” para posterior mente mostrarlo en pantalla.

```

Seleccione la opción que desea visualizar:4
10 Productos más buscados
54 SSD Kingston A400, 120GB, SATA III, 2.5'', 7mm 259 discos duros 300
57 SSD Adata Ultimate SU800, 256GB, SATA III, 2.5'', 7mm 889 discos duros 15
29 Tarjeta Madre ASUS micro ATX TUF B450M-PLUS GAMING, S-AM4, AMD B450, HDMI, 64GB DDR4 para AMD 2499 tarjetas madre 10
3 Procesador AMD Ryzen 5 2600, S-AM4, 3.40GHz, Six-Core, 16MB L3 Cache, con Disipador Wraith Stealth 3089 procesadores 987
4 Procesador AMD Ryzen 3 3200G con Gráficos Radeon Vega 8, S-AM4, 3.60GHz, Quad-Core, 4MB L3, con Disipador Wraith Spire 2209 procesadores 295
85 Logitech Audifonos Gamer G635 7.1, Alámbrico, 1.5 Metros, 3.5mm, Negro/Azul 2159 audifonos 39
67 TV Monitor LED 24TL520S-PU 24, HD, Widescreen, HDMI, Negro 3229 pantallas 411
7 Procesador Intel Core i7-9700K, S-1151, 3.60GHz, 8-Core, 12MB Smart Cache (9na. Generación Coffee Lake) 8559 procesadores 114
5 Procesador Intel Core i3-9100F, S-1151, 3.60GHz, Quad-Core, 6MB Cache (9na. Generación - Coffee Lake) 1779 procesadores 130
47 SSD XPG SX8200 Pro, 256GB, PCI Express, M.2 1209 discos duros 8

```

Fig. 11 Productos más buscados.

#### 2.7.4. Productos menos buscados

```

print(clr_sub+'10 Productos menos buscados. \n '+clr_reset)
busquedas = []
for val in lifestore_searches: #Funcion para definir un
listado de las ventas
    busquedas.append(val[1])
    repeticiones = {}

for n in busquedas:

```

```

        if n in repeticiones :
            repeticiones[n] += 1
        else:
            repeticiones[n] = 0
    primero = (collections.Counter(repeticiones).most_common() [-
1][0]) - 1
    segundo = (collections.Counter(repeticiones).most_common() [-
2][0]) - 1
    tercero = (collections.Counter(repeticiones).most_common() [-
3][0]) - 1
    cuarto = (collections.Counter(repeticiones).most_common() [-
4][0]) - 1
    quinto = (collections.Counter(repeticiones).most_common() [-
5][0]) - 1
    sexto = (collections.Counter(repeticiones).most_common() [-
6][0]) - 1
    septimo = (collections.Counter(repeticiones).most_common() [-
7][0]) - 1
    octavo = (collections.Counter(repeticiones).most_common() [-
8][0]) - 1
    noveno = (collections.Counter(repeticiones).most_common() [-
9][0]) - 1
    decimo = (collections.Counter(repeticiones).most_common() [-
10][0]) - 1

    print(
        lifestore_products[primero] [1], '\n',
        lifestore_products[segundo] [1], '\n',
        lifestore_products[tercero] [1], '\n',
        lifestore_products[cuarto] [1], '\n',
        lifestore_products[quinto] [1], '\n',
        lifestore_products[sexto] [1], '\n',
        lifestore_products[septimo] [1], '\n',
        lifestore_products[octavo] [1], '\n',
        lifestore_products[noveno] [1], '\n',
        lifestore_products[decimo] [1], '\n'
    )

```

Tomando en cuenta la resolución para los productos más buscados, el programa va a tomar los 5 valores menos repetidos, tomando en cuenta que se usara valores negativos para ubicar su posición, posterior a ello y tomando en cuenta que solo tiene el número de lista del artículo, realizara una búsqueda de ese artículo dentro de la lista “lifestore\_products” para posterior mente mostrarlo en pantalla.

```

Seleccione la opción que desea visualizar:5
10 Productos menos buscados.

93 Ginja Audífonos con Micrófono GI18ADJ01BT-RO, Bluetooth, Alámbrico/Inalámbrico, 3.5mm, Rojo 160 audifonos 139
80 Ghia Bocina Portátil BX800, Bluetooth, Inalámbrico, 2.1 Canales, 31W, USB, Negro 1359 bocinas 15
70 Samsung Smart TV LED 43, Full HD, Widescreen, Negro 7679 pantallas 10
59 SSD Samsung 860 EVO, 1TB, SATA III, M.2 5539 discos duros 10
45 Tarjeta Madre ASRock ATX H110 Pro BTC+, S-1151, Intel H110, 32GB DDR4, para Intel 2869 tarjetas madre 25
35 Tarjeta Madre Gigabyte micro ATX Z390 M GAMING, S-1151, Intel Z390, HDMI, 64GB DDR4 para Intel 3419 tarjetas madre 30
27 Tarjeta de Video VisionTek AMD Radeon HD5450, 2GB GDDR3, PCI Express x16 2109 tarjetas de video 43
10 MSI GeForce 210, 1GB GDDR3, DVI, VGA, HDCP, PCI Express 2.0 889 tarjetas de video 13
9 Procesador Intel Core i3-8100, S-1151, 3.60GHz, Quad-Core, 6MB Smart Cache (8va. Generación - Coffee Lake) 2549 procesadores 35
91 Genius GHP-400S Audífonos, Alámbrico, 1.5 Metros, Rosa 137 audifonos 16

```

Fig. 12 Productos menos buscados.

## 2.7.5. Productos con peores reseñas

```

print(clr_sub+'Productos con mejores reseñas'+ clr_reset)
def merge(resenas, id_producto):
    merged_list = tuple(zip(resenas, id_producto))
    return merged_list

resenas = []
id_producto = []

for val in lifestore_sales:
    resenas.append(val[2])

for id in lifestore_sales:
    id_producto.append(id[1])

lista_fusion = merge(id_producto, resenas)

repeticiones = {}

for n in lista_fusion:
    if n in repeticiones :
        repeticiones[n] += 1
    else:
        repeticiones[n] = 0

primero =
(collections.Counter(repeticiones).most_common()[0][0])
segundo =
(collections.Counter(repeticiones).most_common()[1][0])
tercero =
(collections.Counter(repeticiones).most_common()[2][0])
cuarto =
(collections.Counter(repeticiones).most_common()[3][0])
quinto =
(collections.Counter(repeticiones).most_common()[4][0])

print(
    lifestore_products[(primero[0])-1][1], '\n',
    lifestore_products[(segundo[0])-1][1], '\n',
    lifestore_products[(tercero[0])-1][1], '\n',
    lifestore_products[(cuarto[0])-1][1], '\n',
    lifestore_products[(quinto[0])-1][1], '\n',
)

```

Mediante dos ciclos for, se analizarán dos determinadas columnas de la lista “lifestore\_sales” para posteriormente unir las mediante una definición de nombre “merge”, con ayuda de la función “zip” con la cual podremos unir ambas listas para posteriormente realizar un ciclo “for” para contabilizar

el número de veces que se repite. Posteriormente utilizando la función “counter” y “most\_common” podemos determinar cuáles fueron las opciones más populares de la lista, de lo cual tomaremos las primeras 5, con solo los primeros números para obtener cual es la posición del producto, para posteriormente buscarlas en la lista de “lifestore\_products” e imprimir el valor.

```

Seleccione la opción que desea visualizar: 1
Productos con mejores reseñas
54 SSD Kingston A400, 120GB, SATA III, 2.5'', 7mm 259 discos duros 300
3 Procesador AMD Ryzen 5 2600, S-AM4, 3.40GHz, Six-Core, 16MB L3 Cache, con Disipador Wraith Stealth 3089 procesadores 987
5 Procesador Intel Core i3-9100F, S-1151, 3.60GHz, Quad-Core, 6MB Cache (9na. Generación - Coffee Lake) 1779 procesadores 130
57 SSD Adata Ultimate SU800, 256GB, SATA III, 2.5'', 7mm 889 discos duros 15
54 SSD Kingston A400, 120GB, SATA III, 2.5'', 7mm 259 discos duros 300

```

Fig. 13 Productos con mejores reseñas.

### 2.7.6. Productos con las peores reseñas

```

print(clr_sub+'Productos con peores reseñas'+clr_reset)
def merge(resenas, id_producto):
    merged_list = tuple(zip(resenas, id_producto))
    return merged_list

resenas = []
id_producto = []

for val in lifestore_sales:
    resenas.append(val[2])

for id in lifestore_sales:
    id_producto.append(id[1])

lista_fusion = merge(id_producto, resenas)

repeticiones = {}

for n in lista_fusion:
    if n in repeticiones :
        repeticiones[n] += 1
    else:
        repeticiones[n] = 0

primero = (collections.Counter(repeticiones).most_common() [-
1] [0])
segundo = (collections.Counter(repeticiones).most_common() [-
2] [0])
tercero = (collections.Counter(repeticiones).most_common() [-
3] [0])
cuarto = (collections.Counter(repeticiones).most_common() [-
4] [0])
quinto = (collections.Counter(repeticiones).most_common() [-
5] [0])

print(lifestore_products[(primero[0])-1] [1], '\n',

```

```
lifestore_products[(segundo[0])-1][1], '\n',
lifestore_products[(tercero[0])-1][1], '\n',
lifestore_products[(cuarto[0])-1][1], '\n',
lifestore_products[(quinto[0])-1][1], '\n',
)
```

Se realiza el mismo procedimiento mostrado con anterioridad, solo que, a diferenciar del anterior, en este caso se ubicaran las posiciones con valores negativos para posteriormente realizar la búsqueda del artículo.

```
Seleccione la opción que desea visualizar: 2
Productos con peores reseñas
94 HyperX Audifonos Gamer Cloud Flight para PC/PS4/PS4 Pro, Inalámbrico, USB, 3.5mm, Negro 2869 audifonos 12
89 Cougar Audifonos Gamer Phontum Essential, Alámbrico, 1.9 Metros, 3.5mm, Negro. 859 audifonos 4
84 Logitech Audifonos Gamer G332, Alámbrico, 2 Metros, 3.5mm, Negro/Rojo 1089 audifonos 83
74 Logitech Bocinas para Computadora con Subwoofer G560, Bluetooth, Inalámbrico, 2.1, 120W RMS, USB, negro 4239 bocinas 1
74 Logitech Bocinas para Computadora con Subwoofer G560, Bluetooth, Inalámbrico, 2.1, 120W RMS, USB, negro 4239 bocinas 1
```

Fig. 14 Productos con las peores reseñas.

### 2.7.7. Total, de ventas realizadas

```
print(clr_sub+'Total de ventas realizadas'+ clr_reset + '\n')
ventas_totales =[]

for val in lifestore_sales:
    ventas_totales.append(val[1])

print('Se realizaron un total de:',len(ventas_totales),
'ventas realizadas. \n')
```

Estableciendo como primera instancia una nueva lista de nombre “ventas\_totales” en donde serán almacenados los valores del ciclo. El ciclo `for` será el encargado de leer todos los valores de la segunda columna de la lista “lifestore\_sales” en donde se almacenan los productos vendidos una vez tomo todos los valores, los almacenara en la lista que hemos creado.

Posterior a ello se mostrara en consola un mensaje con uso de la función `len()` que será la encargada de contar la cantidad de elementos en la lista.

```
Seleccione la opción que desea visualizar: 1
Total de ventas realizadas

Se realizaron un total de: 283 ventas realizadas.
```

Fig. 15 Total de productos vendidos.

### 2.7.8. Ventas por Mes

```
id_fecha =[[sale[0], sale[3]] for sale in lifestore_sales if
sale[4]==0]
```

```

meses = {}

for par in id_fecha:
    id = par[0]
    _, mes, _ = par[1].split('/')
    if mes not in meses.keys():
        meses[mes] = []
    meses[mes].append(id)

for key in meses.keys():
    list_mes = meses[key]
    venta = 0
    for id_venta in list_mes:
        indice = id_venta - 1
        info_venta = lifestore_sales[indice]
        id_prod = info_venta[1]
        precio = lifestore_products[id_prod - 1][2]
        venta += precio
        if key == '01':
            key = 'Enero'
        elif key == '02':
            key = 'Febrero'
        elif key == '03':
            key = 'Marzo'
        elif key == '04':
            key = 'Abril'
        elif key == '05':
            key = 'Mayo'
        elif key == '06':
            key = 'Junio'
        elif key == '07':
            key = 'Julio'
        elif key == '08':
            key = 'Agosto'
        elif key == '09':
            key = 'Septiembre'
        elif key == '10':
            key = 'Octubre'
        elif key == '11':
            key = 'Noviembre'
        elif key == '12':
            key = 'Diciembre'

    print('Durante el mes de:', clr_mes, negrita, key, clr_reset, '\n'
          'Se realizó una venta por:', Fore.RED, negrita, venta, clr_reset, 'MXN\n', 'Con'
          'un total de:', clr_items, negrita, len(list_mes), clr_reset, 'articulos'
          'vendidos')

```

Mediante el primer ciclo for tendremos creada un diccionario de manera en que allí va a almacenar específicamente las fechas tomaras de la lista de ventas y posteriormente con otro ciclo for realizara

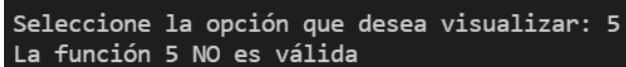


la suma de ventas con respecto al precio de cada producto vendido en determinado mes. Para realizar una impresión en pantalla mucho mas amigable con el usuario, mediante condicionales, evaluará el numero del mes e imprimirá el nombre del mes, junto con el precio y la cantidad de artículos que se vendieron en aquel mes.

#### 2.7.9. Entrada de teclado no valida

```
else:  
    print('La función' + seleccion + 'NO es válida')
```

Si el operador llegase a insertar una opción que no se muestre en consola, el comparador detectara que esa entrada es ajena a las declaraciones y valores establecidos, por lo cual mostrara en consola un mensaje donde mencione que la entrada que inserto no es legítima.



```
Seleccione la opción que desea visualizar: 5  
La función 5 NO es válida
```

Fig. 16 Entrada de teclado no valida.

### 3. GitHub

Siendo un portal para alojar el código de aplicaciones, la plataforma esta creada para que los desarrolladores suban el código de sus aplicaciones y herramientas, donde como usuario no solo puedas bajar la aplicación, sino colaborar con el desarrollo de esta o documentarte sobre el funcionamiento de esta.

<https://github.com/lal0cervantes/JaibaTech>

## 4. Gráficos

### 4.1 Artículo más vendido



Fig. 17 Producto más vendido.

### 4.2. Artículos más vendidos



Fig. 18 Top 5 más vendido

## 4.3. Artículos menos vendidos



Fig. 19 Top 5 menos vendido

## 4.4. Artículos más buscados



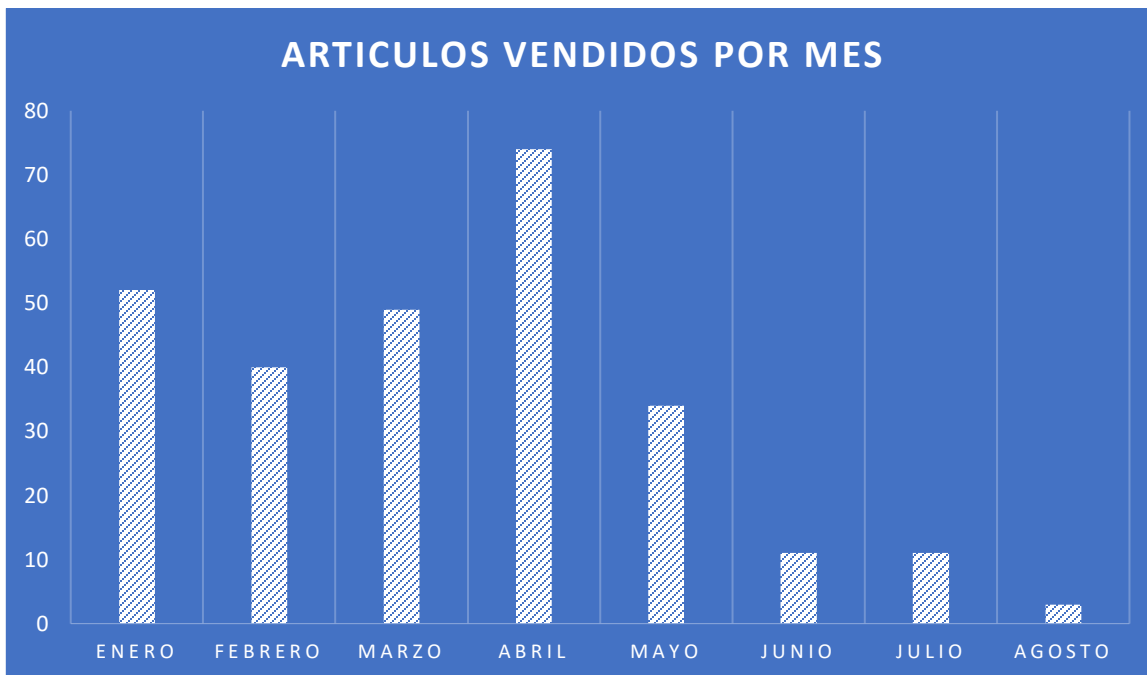
Fig. 20 Top 10 de productos más buscados

## 4.5. Artículos menos buscados

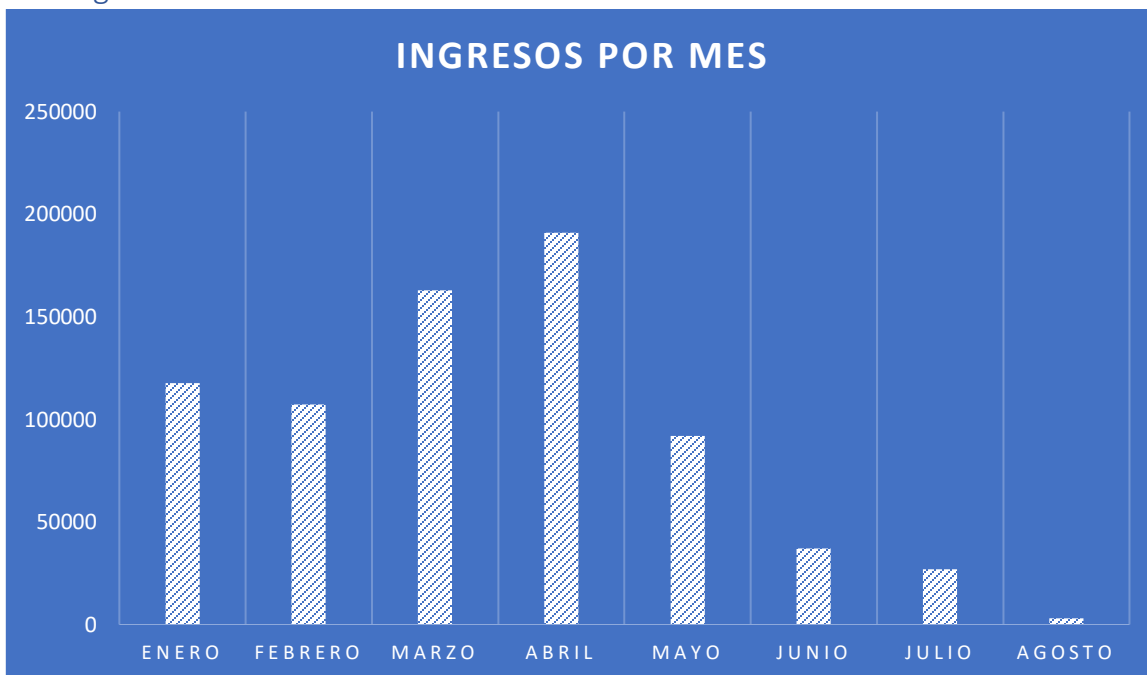


Fig. 21 Top 10 de productos menos buscados

## 4.6. Ventas en artículos



## 4.7. Ingresos



## 5. Conclusión

Durante todo el proceso de desarrollo tuve algunos problemas, debido a no saber cómo interpretar los datos, pero una vez me centre en desarrollar las funciones, paso a paso y pensar en cómo lo haría de manera manual todo, flujo, aunque también fue de aprendizaje ya que en diversos momentos o determinados métodos no lo conocía, aunque ya hubiera trabajado con ellos con anterioridad, podría ser un gran programa, así como añadir otras mejoras o funciones como:

- 1.- Entrada de teclado para cerrar sesión.
- 2.- Entrada de teclado para regresar al menú anterior.
- 3.- Entrada de teclado para regresar al menú principal.
- 4.- Entrada de teclado para salir del sistema de análisis de datos.
- 5.- Entrada de teclado para guardar los datos seleccionados en un archivo de texto.