EMTECH
Emerging Technologies Institute

# PROYECTO 1. INTRODUCCIÓN A PYTHON

## Grupo 4.- Data Science


## Elaborado por Luis David Ramírez de la Cruz


**“The Last Dance”**

**04/09/20**

# ÍNDICE



## Tabla de contenido

# OBJETIVO

Poner en práctica las bases de programación en Python para análisis y clasificación de datos mediante la creación de programas de entrada de usuario y validaciones, uso y definición de variables y listas, operadores lógicos y condicionales para la clasificación de información.

# DESCRIPCIÓN DEL CASO

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

# CONSIGNA

Derivado de la situación, la Gerencia de Ventas te solicita que realices un análisis de la rotación de productos identificando los siguientes elementos:

1. Productos más vendidos y productos rezagados a partir del análisis de las categorías con menores ventas y categorías con menores búsquedas.
2. Productos por reseña en el servicio a partir del análisis de categorías con mayores ventas y categorías con mayores búsquedas.
3. 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.

# INTRODUCCIÓN

En el documento se presenta la realización del algoritmo que resuelve lo solicitado por la Gerencia de Ventas, también se incluirán algunas recomendaciones respecto a los cambios que deben realizarse en respuesta a la información recaudada (pensando en la conveniencia del inventario, costo de almacenamiento, operación y de oportunidad).

El algoritmo está hecho para que al mostrar los datos, sea fácil de comprender incluso para quien no tiene conocimientos en informática o análisis de datos. Para ser mas comprensible y compacto se utilizaron identificadores para cada producto y un menú que permite navegar con facilidad. Se incluye también la obligación de ingresar un usuario y contraseña para ingresar a el menú antes mencionado.

# SOLUCIÓN AL PROBLEMA

Poniendo en práctica lo aprendido, planeé utilizar la lección sobre "Operaciones con Listas" y "Tuplas" para realizar un algoritmo que para cada producto asigne en una lista el precio, las compras, mes de compra, ingreso anual, ventas mensuales y su categoría, de esta manera se podrá mostrar de forma ordena la información según la requiera el usuario.
Se debe iniciar en primera instancia con la solicitud de credenciales para permitir acceso a la información, también se incluye un menú con el que el usuario tendrá interacción y decidirá la información que será mostrada.

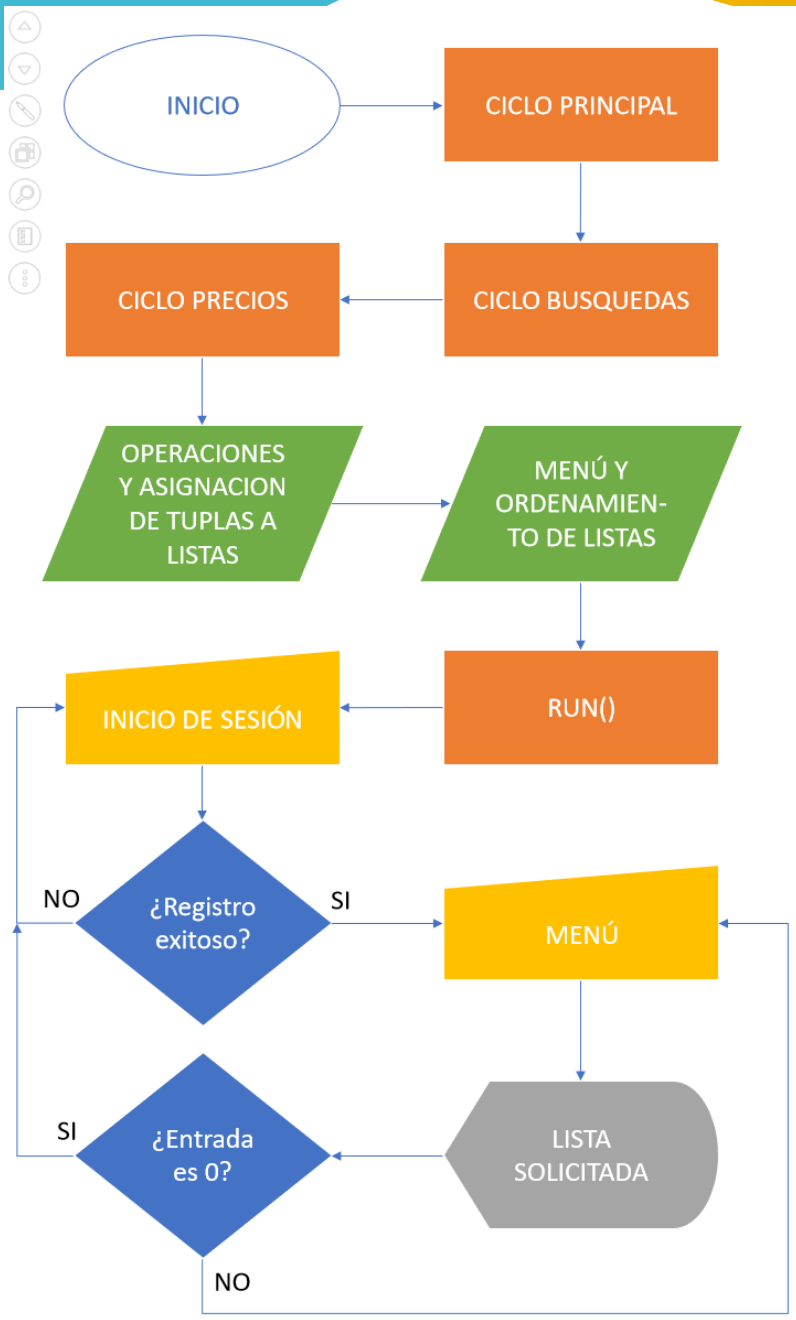


*Imagen 1.1 – Diagrama de flujo del algoritmo*

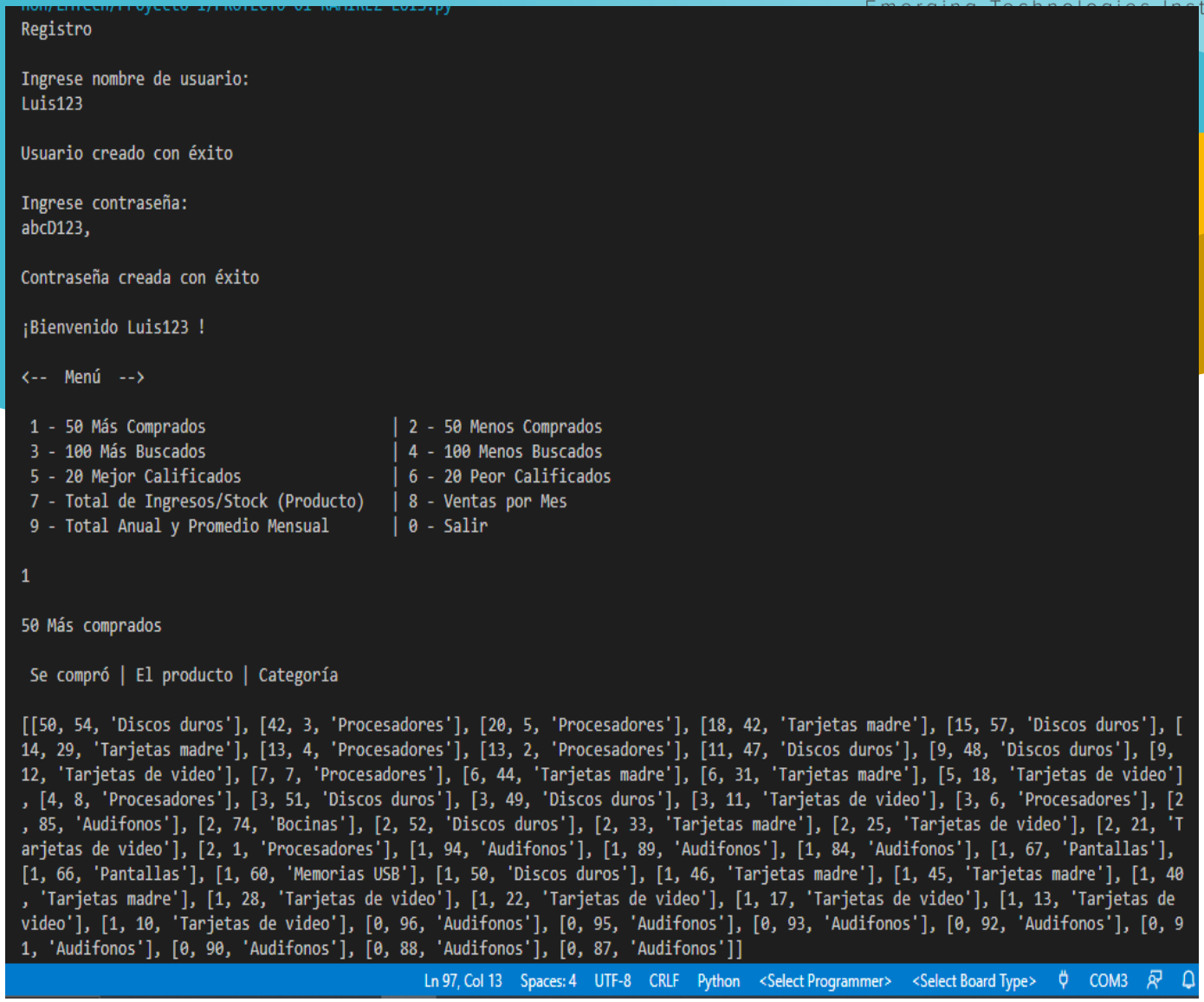


*Imagen 1.2 – Captura de pantalla del funcionamiento*

# DEFINICIÓN DEL CÓDIGO

En primera instancia se definen las listas y variables que serán utilizadas a lo largo del algoritmo, así mismo se importa la base de datos de LifeStore para poder trabajar con la información recabada.

```
import operator
from lifestore_file import lifestore_searches
from lifestore_file import lifestore_sales
from lifestore_file import lifestore_products
import user
import passw


#<!-- DECLARACIÓN DE VARIABLES  -->

#Listas
A=[]
B=[]
C=[]
D=[]
E=[]
F=[]
G=[]
H=[]

comprasTotales=[]
preciosTotales=[]
stock=[]

#Valores
Enero=0
Febrero=0
Marzo=0
Abril=0
Mayo=0
Junio=0
Julio=0
Agosto=0
Septiembre=0
Octubre=0
Noviembre=0
Diciembre=0
y=('')
z=1
```

## Login de Usuario

Se implementa mediante entrada de teclado un registro para el ingreso del usuario que desea acceso a los datos, para poder crear un "**Nickname**" se debe considerar:

1. El nombre de usuario debe contener un mínimo de 6 caracteres y un máximo de 12.
2. El nombre de usuario debe ser alfanumérico.
3. Nombre de usuario con menos de 6 caracteres, retorna el mensaje "El nombre de usuario debe contener al menos 6 caracteres".
4. Nombre de usuario con más de 12 caracteres, retorna el mensaje "El nombre de usuario no puede contener más de 12 caracteres".
5. Nombre de usuario con caracteres distintos a los alfanuméricos, retorna el mensaje "El nombre de usuario puede contener solo letras y números".
6. Nombre de usuario válido, retorna True y procede a pedir contraseña.

Para crear una **contraseña** se considera:

1. La contraseña debe contener un mínimo de 8 caracteres.
2. Una contraseña debe contener letras minúsculas, mayúsculas, números y al menos 1 carácter no alfanumérico.
3. La contraseña no puede contener espacios en blanco.
4. Contraseña válida, retorna True y procede al menú.
5. Contraseña no válida, retorna el mensaje "La contraseña elegida no es segura".

Ambas condicionales se realizaron en módulos separados* incluidos en la carpeta del proyecto, a continuación, se muestra la petición de las credenciales en el módulo principal, donde se valida el usuario y la contraseña:

*Incluidos en el Anexo A y B

```
#<!--  CREDENCIALES  -->

def run():
    print("Registro\n")
    correcto=False
    while correcto==False:
            nombre=input("Ingrese nombre de usuario:\n")
            if user.nickname(nombre) == True:
                print("\nUsuario creado con éxito\n")
                correcto=True

    while correcto==True:
        contrasenia=input("Ingrese contraseña:\n")
```

```
        if passw.clave(contrasenia)==True:
            print("\nContraseña creada con éxito\n")
            correcto=False

    #Inicia el menú para mostrar la información que elegirá el usuario
    print("¡Bienvenido", nombre,"!")
```

## Ciclo principal

Algoritmo de ordenamiento, este es donde se recaba la información de cada producto, como su Categoría (y), ID_Producto (z), compras (i), calif (prom), devoluciones (d) y ventas (s).

Toda esta información recabada en cada lectura es almacenada en listas previamente definidas y asignadas a la posición del producto:*

*Para poder saber a detalle el funcionamiento por favor refiérase al Anexo C

```
#<!--  VENTAS  -->

for z in range(97):
    x=0
    i=0
    r=0
    d=0
    s=0
    prom=0


    #De la lista "lifestore_sales" se obtienen los valores con los que se trabajaran en est
e ciclo

    for row in lifestore_sales:

        valX = lifestore_sales[x][1]

        valW = lifestore_sales[x][2]

        valD = lifestore_sales[x][4]

        valF = lifestore_sales[x][3]
        valF = valF[3:5]
```

```
        x+=1

        if valX == z:
            if valD == 1:                                   #Revisa si hay alguna devolución
                d-=1
            #Categoría
            if z<10:…

            i+=1
            r=r+valW
            prom=r/i

        #Se filtra el mes de compra para saber en que mes se realizaron mas compras
            if valF=='01':…


        #Para evitar mostrar en los peores calificados a los productos que no han tenido co
mpras
        #se detecta si las compras (i) son iguales a Cero (0) para luego eliminarlas de la
lista F
        if i==0:
            prom=0

        #Toplas que asignan las veces que se compra (i) a cada producto (z),
        #categoría (y), calif (prom), devoluciones (d)
        sumaAscendente = [i,z,y]
        sumaDescendente = [y,i,z]
        sumaE = [y,prom,i,z,d]
        sumaF = [y,prom,i,z,d]
        sumaM = [[Enero,1],[Febrero,2],[Marzo,3],[Abril,4],[Mayo,5],[Junio,6],[Julio,7],[Ag
osto,8],[Septiembre,9],[Octubre,10],[Noviembre,11],[Diciembre,12]]



    #A = [Posición, [veces_comprado, producto, Categoría]]
    #B = [Posición, [Categoría, veces_comprado, producto]]
    #E = [Posición, [Categoría, Calificación, veces_comprado, producto, devoluciones]]
    #F = [Posición, [Categoría, Calificación, veces_comprado, producto, devoluciones]]
    #G = [Posición, [Compras_mes, Mes]]

    #Inserta Topla "SUMA" a las listas A, B, C, D, E, F, G con estas en posicion de cada pr
oducto (z)
    A.insert(z, sumaAscendente)
    E.insert(z, sumaE)
    B.insert(z, sumaDescendente)
    F.insert(z, sumaF)
    G.insert(z, sumaM)
```

```
#Compras hechas de cada producto en posicion del producto (z)
comprasTotales.insert(z, i)

if prom == 0:
        F.pop()
```

## Búsquedas

Dentro del mismo ciclo principal se encuentra una división que realiza la identificación del total de búsquedas por cada producto y se las asigna a su ID (z) en una Topla:

```
#<!--  BUSQUEDAS  -->

    x=0
    i=0

    #De la lista lifestore_searches obtiene las busquedas por producto

    for row1 in lifestore_searches:

        valY = lifestore_searches[x][1]
        x+=1

        #Asigna la categoría al producto
        if valY == z:…

            i+=1

        #Toplas que asignan busquedas (i) al producto (z) de categoría (y)
        sumaSearch = [i,z,y]
        sumaSearchCat = [y,i,z]

    #C = [Posición, [veces_buscado, producto, Categoría]]
    #D = [Posición, [Categoría, veces_buscado, producto]]

    C.insert(z, sumaSearch)

    D.insert(z, sumaSearchCat)
    z+=1
```

## Precios y Stock

Al salir del loop principal se lee el precio de cada producto y su stock para seguidamente, asignarlo en listas y poder ser leído por el usuario:

```
#<!--  PRECIOS  -->

x=0

#Ciclo para obtener el precio (valP) de cada producto (x)
for row in lifestore_products:

    valP = lifestore_products[x][2]
    valS = lifestore_products[x][4]
    x+=1
    preciosTotales.insert(x,valP)
    stock.insert(x, valS)
```

## Multiplicaciones finales

Como modo de preparación, se realizan multiplicaciones del precio de cada producto por el numero de ventas para obtener los ingresos:

```
#Ciclo para asignar a cada producto (z) el ingreso aportado en el año
H = [[(z+1), preciosTotales[z]*comprasTotales[z], stock[z]] for z in range(len(preciosTotal
es))]

#Ciclo para obtener los ingresos totales en el año
I = [preciosTotales[z]*comprasTotales[z] for z in range(0,len(preciosTotales))]
```

## Menú y despliegue de listas

Por último, al darle acceso al usuario, podrá visualizar un menú en la terminal, donde podrá ingresar del 0 al 9 según la preferencia y se mostrará la información solicitada en forma de lista, si el numero ingresado es diferente de las opciones, mostrará un mensaje de error, y si es cero (0) cierra sesión:

```
def menu():

    while True:

        print("\n<--  Menú  -->\n ")

        #<!--  COMPRADOS  -->

        if opcionMenu=='1':
            A.sort(reverse = True)
            print("\n50 Más comprados\n \n Se compró | El producto | Categoría\n")
            print(A[:50])
            print()

        elif opcionMenu=='2':
            K= sorted(B, key=operator.itemgetter(0,1))
            print("\n50 Menos comprados (Por categoría)\n \n Categoría | Se compró | El pro
ducto\n")
            print(K[1:51])
            print()

        # <!--  BUSCADOS  -->

        elif opcionMenu=='3':
            C.sort(reverse= True)
            print("\n100 Mas Buscados\n \n Se Buscó | El producto | Categoría\n")
            print(C[:100])                                                  #Mostrar 100
            print()

        elif opcionMenu=='4':
            L= sorted(D, key=operator.itemgetter(0,1))
            print("\n100 Menos Buscados (Por categoría)\n \n Categoría | Se Buscó | El prod
ucto \n")
            print(L[1:101])                                                 #Mostrar 100
            print()

        # <!--  RESEÑAS  -->

        elif opcionMenu=='5':
            M= sorted(E, key=operator.itemgetter(4,1,0,2), reverse= True)
            print("\n20 Mejor Calificados (Por categoría)\n \n Categoría | Calificación | S
e compró | El producto | Devoluciones \n")
            print(M[:20])
            print()

        elif opcionMenu=='6':
```

```
            N= sorted(F, key=operator.itemgetter(4,1,0,2), reverse= False)
            print("\n20 Peor Calificados (Por categoría)\n \n Categoría | Calificación | Se
 compró | El producto | Devoluciones \n")
            print(N[1:21])
            print()

        # <!--  INGRESOS  -->

        elif opcionMenu=='7':
            J= sorted(H, key=operator.itemgetter(1,2,0), reverse= False)
            print("\nIngreso por producto (En el plazo anual)\n \n Producto | Ingreso | Sto
ck\n")
            print(J)
            print()

        elif opcionMenu=='8':
            print("\nVentas por mes (De Enero-Diciembre)\n \n Ventas | Mes \n")
            print(G[-1])
            print()

        elif opcionMenu=='9':
            print("\nTotal ingresos por venta (En el plazo anual)\n")
            print(sum(I))
            print()

            print("\nPromedio mensual (2020)\n")
            print(sum(I)/12)
            print()

        elif opcionMenu=='0':
            run()

        else:
            print("\nPor favor, ingresa una opción válida\n")
```

## Call

Para terminar, solo es llamada la actividad que mostrará la interfaz

```
run()
```

# CONCLUSIONES

1) Productos más vendidos y productos rezagados

A partir del análisis realizado en el algoritmo, el resultado sobre los productos con mayores ventas muestra que el producto 54 (SSD Kingston A400, 120GB), 3 (Procesador AMD Ryzen 5 2600), 5 (Procesador Intel Core i3-9100F), 42 (Tarjeta Madre ASRock Micro ATX B450M Steel Legend) y 57 (SSD Adata Ultimate SU800, 256GB) son los productos con mayores ventas, lo que por su precio: $259, $3089, $1779, $1779 y $3269 respectivamente, los clientes prefieren los producto por la relación calidad precio, si bien son los artículos más baratos de su categoría, siguen recibiendo un producto de calidad.

```
50 Más comprados

 Se compró | El producto | Categoría

[[50, 54, 'Discos duros'], [42, 3, 'Procesadores'], [20, 5, 'Procesadores'], [18, 42, 'Tarjetas madre'], [15, 57, 'Discos duros'], [
14, 29, 'Tarjetas madre'], [13, 4, 'Procesadores'], [13, 2, 'Procesadores'], [11, 47, 'Discos duros'], [9, 48, 'Discos duros'], [9,
12, 'Tarjetas de video'], [7, 7, 'Procesadores'], [6, 44, 'Tarjetas madre'], [6, 31, 'Tarjetas madre'], [5, 18, 'Tarjetas de video']
, [4, 8, 'Procesadores'], [3, 51, 'Discos duros'], [3, 49, 'Discos duros'], [3, 11, 'Tarjetas de video'], [3, 6, 'Procesadores'], [2
, 85, 'Audifonos'], [2, 74, 'Bocinas'], [2, 52, 'Discos duros'], [2, 33, 'Tarjetas madre'], [2, 25, 'Tarjetas de video'], [2, 21, 'T
arjetas de video'], [2, 1, 'Procesadores'], [1, 94, 'Audifonos'], [1, 89, 'Audifonos'], [1, 84, 'Audifonos'], [1, 67, 'Pantallas'],
[1, 66, 'Pantallas'], [1, 60, 'Memorias USB'], [1, 50, 'Discos duros'], [1, 46, 'Tarjetas madre'], [1, 45, 'Tarjetas madre'], [1, 40
, 'Tarjetas madre'], [1, 28, 'Tarjetas de video'], [1, 22, 'Tarjetas de video'], [1, 17, 'Tarjetas de video'], [1, 13, 'Tarjetas de
video'], [1, 10, 'Tarjetas de video'], [0, 96, 'Audifonos'], [0, 95, 'Audifonos'], [0, 93, 'Audifonos'], [0, 92, 'Audifonos'], [0, 9
1, 'Audifonos'], [0, 90, 'Audifonos'], [0, 88, 'Audifonos'], [0, 87, 'Audifonos']]
```

*Imagen 2.1 – 50 más comprados*

En el caso de los productos rezagados, se recomienda realizar un cambio en el alcance del mercado debido a lo observado. Las categorías de "Audífonos", "Bocinas" y "Pantallas" son las que mas aparecen en ambas listas (mostradas a continuación), sin embargo, la categoría de "Memorias USB" es la menos buscada y esto ocasiona bajas ventas y costo por perdida de oportunidad.

```
50 Menos comprados (Por categoría)

 Categoría | Se compró | El producto

[['Audifonos', 0, 86], ['Audifonos', 0, 87], ['Audifonos', 0, 88], ['Audifonos', 0, 90], ['Audifonos', 0, 91], ['Audifonos', 0, 92],
 ['Audifonos', 0, 93], ['Audifonos', 0, 95], ['Audifonos', 0, 96], ['Audifonos', 1, 84], ['Audifonos', 1, 89], ['Audifonos', 1, 94],
 ['Audifonos', 2, 85], ['Bocinas', 0, 75], ['Bocinas', 0, 76], ['Bocinas', 0, 77], ['Bocinas', 0, 78], ['Bocinas', 0, 79], ['Bocinas
', 0, 80], ['Bocinas', 0, 81], ['Bocinas', 0, 82], ['Bocinas', 0, 83], ['Bocinas', 2, 74], ['Discos duros', 0, 53], ['Discos duros',
 0, 55], ['Discos duros', 0, 56], ['Discos duros', 0, 58], ['Discos duros', 0, 59], ['Discos duros', 1, 50], ['Discos duros', 2, 52]
, ['Discos duros', 3, 49], ['Discos duros', 3, 51], ['Discos duros', 9, 48], ['Discos duros', 11, 47], ['Discos duros', 15, 57], ['D
iscos duros', 50, 54], ['Memorias USB', 0, 61], ['Memorias USB', 0, 62], ['Memorias USB', 0, 63], ['Memorias USB', 1, 60], ['Pantall
as', 0, 64], ['Pantallas', 0, 65], ['Pantallas', 0, 68], ['Pantallas', 0, 69], ['Pantallas', 0, 70], ['Pantallas', 0, 71], ['Pantall
as', 0, 72], ['Pantallas', 0, 73], ['Pantallas', 1, 66], ['Pantallas', 1, 67]]
```

*Imagen 2.2 – 50 menos comprados*

```
100 Menos Buscados (Por categoría)

 Categoría | Se Buscó | El producto

[['Audifonos', 0, 86], ['Audifonos', 0, 87], ['Audifonos', 0, 88], ['Audifonos', 0, 90], ['Audifonos', 0, 92], ['Audifonos', 0, 96],
 ['Audifonos', 1, 93], ['Audifonos', 2, 91], ['Audifonos', 3, 95], ['Audifonos', 6, 94], ['Audifonos', 7, 89], ['Audifonos', 10, 84]
, ['Audifonos', 35, 85], ['Bocinas', 0, 75], ['Bocinas', 0, 77], ['Bocinas', 0, 78], ['Bocinas', 0, 79], ['Bocinas', 0, 81], ['Bocin
as', 0, 82], ['Bocinas', 0, 83], ['Bocinas', 1, 80], ['Bocinas', 2, 76], ['Bocinas', 6, 74], ['Discos duros', 0, 53], ['Discos duros
', 0, 55], ['Discos duros', 0, 58], ['Discos duros', 1, 59], ['Discos duros', 2, 56], ['Discos duros', 5, 52], ['Discos duros', 7, 5
0], ['Discos duros', 10, 49], ['Discos duros', 11, 51], ['Discos duros', 27, 48], ['Discos duros', 30, 47], ['Discos duros', 107, 57
], ['Discos duros', 263, 54], ['Memorias USB', 0, 60], ['Memorias USB', 0, 61], ['Memorias USB', 0, 62], ['Pantallas', 0, 64], ['Pan
tallas', 0, 65], ['Pantallas', 0, 68], ['Pantallas', 0, 69], ['Pantallas', 0, 71], ['Pantallas', 0, 72], ['Pantallas', 1, 70], ['Pan
tallas', 4, 63], ['Pantallas', 4, 73], ['Pantallas', 15, 66], ['Pantallas', 32, 67], ['Procesadores', 1, 9], ['Procesadores', 10, 1]
```

*Imagen 2.3 – 100 menos buscados*

2) Productos por reseña en el servicio

Debido a las pocas ventas de las tarjetas de video las calificaciones pueden ser engañosas ya que se muestran como las mejor calificados, sin embargo, realizando un análisis más detallado vemos que la categoría con mas recurrencia es la de "Procesadores", tienen un gran ingreso por venta y muy buenas calificaciones por parte de los compradores (gran oportunidad para enfocar campañas a esta categoría).

```
20 Mejor Calificados (Por categoría)

 Categoría | Calificación | Se compró | El producto | Devoluciones

[['Tarjetas madre', 5.0, 1, 40, 0], ['Tarjetas de video', 5.0, 3, 11, 0], ['Tarjetas de video', 5.0, 2, 21, 0], ['Tarjetas de video'
, 5.0, 2, 25, 0], ['Tarjetas de video', 5.0, 1, 22, 0], ['Tarjetas de video', 5.0, 1, 28, 0], ['Procesadores', 5.0, 7, 7, 0], ['Proc
esadores', 5.0, 4, 8, 0], ['Procesadores', 5.0, 3, 6, 0], ['Procesadores', 5.0, 2, 1, 0], ['Pantallas', 5.0, 1, 66, 0], ['Pantallas'
, 5.0, 1, 67, 0], ['Memorias USB', 5.0, 1, 60, 0], ['Discos duros', 5.0, 3, 49, 0], ['Discos duros', 5.0, 2, 52, 0], ['Discos duros'
, 5.0, 1, 50, 0], ['Audifonos', 5.0, 2, 85, 0], ['Audifonos', 5.0, 1, 84, 0], ['Discos duros', 4.866666666666666, 15, 57, 0], ['Proc
esadores', 4.809523809523809, 42, 3, 0]]
```

*Imagen 2.4 – 20 mejor calificados*

De igual manera, los productos con mayores búsquedas son los de la categoría "Procesadores" y "Discos duros", una muy buena señal que nos muestra la preferencia de nuestros compradores y los productos con mayor alcance.

```
100 Mas Buscados

 Se Buscó | El producto | Categoría

[[263, 54, 'Discos duros'], [107, 57, 'Discos duros'], [60, 29, 'Tarjetas madre'], [55, 3, 'Procesadores'], [41, 4, 'Procesadores'],
 [35, 85, 'Audifonos'], [32, 67, 'Pantallas'], [31, 7, 'Procesadores'], [30, 47, 'Discos duros'], [30, 5, 'Procesadores'], [27, 48,
'Discos duros'], [25, 44, 'Tarjetas madre'], [24, 2, 'Procesadores'], [23, 42, 'Tarjetas madre'], [20, 8, 'Procesadores'], [15, 66,
'Pantallas'], [15, 21, 'Tarjetas de video'], [15, 12, 'Tarjetas de video'], [11, 51, 'Discos duros'], [11, 18, 'Tarjetas de video'],
 [10, 84, 'Audifonos'], [10, 49, 'Discos duros'], [10, 40, 'Tarjetas madre'], [10, 31, 'Tarjetas madre'], [10, 25, 'Tarjetas de vide
o'], [10, 6, 'Procesadores'], [10, 1, 'Procesadores'], [7, 89, 'Audifonos'], [7, 50, 'Discos duros'], [6, 94, 'Audifonos'], [6, 74,
'Bocinas'], [5, 52, 'Discos duros'], [5, 28, 'Tarjetas de video'], [5, 26, 'Tarjetas de video'], [5, 22, 'Tarjetas de video'], [5, 1
1, 'Tarjetas de video'], [4, 73, 'Pantallas'], [4, 63, 'Pantallas'], [4, 46, 'Tarjetas madre'], [4, 15, 'Tarjetas de video'], [3, 95
, 'Audifonos'], [3, 39, 'Tarjetas madre'], [3, 17, 'Tarjetas de video'], [2, 91, 'Audifonos'], [2, 76, 'Bocinas'], [2, 56, 'Discos d
uros'], [2, 13, 'Tarjetas de video'], [1, 93, 'Audifonos'], [1, 80, 'Bocinas'], [1, 70, 'Pantallas'], [1, 59, 'Discos duros'], [1, 4
5, 'Tarjetas madre'], [1, 35, 'Tarjetas madre'], [1, 27, 'Tarjetas de video'], [1, 10, 'Tarjetas de video'], [1, 9, 'Procesadores'],
```

*Imagen 2.5 – 100 más buscados*

3) Sugerir una estrategia

Después de realizar un análisis de cada categoría y búsquedas de los productos se llegó a la conclusión de que la mayor oportunidad se tiene en "Procesadores", claramente los compradores tienen una buena impresión de nuestra parte para comprar sus procesadores.

Por otro lado, tenemos productos que superan 200 unidades en stock y generaron cero (0) ingresos en el trabajo ejercido en el año, en su mayoría "Pantallas", "Audífonos" y "Tarjetas Madre", lo cual produce un mayor costo de operación que ingreso, lo que es evidentemente no conveniente.

Se propone realizar una campaña que impulse aún mas a los compradores a recurrir a LifeStore para consumir sus componentes de procesamiento (Procesadores) y hacer un mejor análisis del mercado para comprender las tendencias de los usuarios (ya que hay actividad que no se incluyen en la base de datos como las características cualitativas) y esto podría inclinar las preferencias hacia componentes o periféricos con diseños y características diferentes de las ofrecidas actualmente, lo que produce perdida por oportunidad.

En el año los meses con mas recurrencia en compras fueron los primeros 5 (de Enero a Mayo), considerando esto se podrían realizar una menor inversión en el segundo semestre para evitar perdidas, esto sugiere tanto menor abastecimiento de inventario como recortes. A pesar de esto, es bien conocido que en los últimos meses del año el público busca abastecerse con la mejor tecnología actual, tomándolo en cuenta, se pueden realizar campañas con temáticas de fechas feriadas para atraer un mayor tráfico.

```
Ventas por mes (De Enero-Diciembre)

 Ventas | Mes

[[53, 1], [41, 2], [51, 3], [75, 4], [36, 5], [11, 6], [11, 7], [3, 8], [1, 9], [0, 10], [1, 11], [0, 12]]
```

*Imagen 2.6 – Ventas por mes*

```
Total ingresos por venta (En el plazo anual)

760177


Promedio mensual (2020)

63348.083333333336
```

*Imagen 2.7 – Total de ingresos*

# ANEXOS

## Anexo A – User.py

```
def nickname(nombre_usuario):           #Modulo que contiene las condiciones para la creaci
ón de usuarios

        long=len(nombre_usuario)        #Calcular la longitud del nomre de usuario
        y=nombre_usuario.isalnum()      #Calcula que la cadena contenga valores alfanumérico
s

        if y== False:                   # La cadena contiene valores no alfanuméricos
            print("El nombre de usuario puede contener solo letras y números")

        if long < 6:
            print("El nombre de usuario debe contener al menos 6 caracteres")

        if long > 12:
            print("El nombre de usuario no puede contener más de 12 caracteres")

        if long >5 and long <13 and y ==True:
            return True                 #Verdadero si el tamaño es mayor a 5 y menor a 13
```

## Anexo B – passw.py

```
def clave(password):                        #Modulo que contiene las condiciones para la cr
eación de contraseñas

        validar=False                       #que se vayan cumpliendo los requisitos uno a u
no.
        long=len(password)                  #Calcula la longitud de la contraseña
        espacio=False                       #variable para identificar espacios
        mayuscula=False                     #variable para identificar letras mayúsculas
        minuscula=False                     #variable para contar/identificar letras minúsc
ulas
        numeros=False                       #variable para identificar números
        y=password.isalnum()                #si es alfanumérica retona True
        correcto=True                       #verifica que hayan mayuscula, minuscula, numer
os y no alfanuméricos
```

```
        for carac in password:              #ciclo for que recorre caracter por caracter en
 la contraseña

            if carac.isspace()==True:       #Saber si el caracter es un espacio
                espacio=True                #si encuentra un espacio se cambia el valor use
r


            if carac.isupper()== True:      #saber si hay mayuscula
                mayuscula=True              #acumulador o contador de mayusculas

            if carac.islower()== True:      #saber si hay minúsculas
                minuscula=True              #acumulador o contador de minúsculas

            if carac.isdigit()== True:      #saber si hay números
                numeros=True                #acumulador o contador de numeros

        if espacio==True:                   #hay espacios en blanco
                print("La contraseña no puede contener espacios")
        else:
            validar=True                    #se cumple el primer requisito que no hayan esp
acios

        if long <8 and validar==True:
            print("Mínimo 8 caracteres")
            validar=False                   #cambia a Flase si no se cumple el requisito mó
inimo de caracteres

        if mayuscula == True and minuscula ==True and numeros == True and y== False and val
idar ==True:
           validar = True                   #Cumple el requisito de tener mayuscula, minusc
ula, numeros y no alfanuméricos
        else:
           correcto=False                   #uno o mas requisitos de mayuscula, minuscula,
numeros y no alfanuméricos no se cumple

        if validar == True and correcto==False:
           print("La contraseña elegida no es segura: debe contener letras minúsculas, mayú
sculas, números y al menos 1 carácter no alfanumérico")

        if validar == True and correcto ==True:
           return True                      #Permite seguir el programa
```

```
#Se importan en el Módulo la base de datos correspondiente a LifeStore
#así como las listas a utilizar, Usuario y contraseña

import operator
from lifestore_file import lifestore_searches
from lifestore_file import lifestore_sales
from lifestore_file import lifestore_products
import user
import passw


#<!-- DECLARACIÓN DE VARIABLES  -->

#Listas
A=[]
B=[]
C=[]
D=[]
E=[]
F=[]
G=[]
H=[]

comprasTotales=[]
preciosTotales=[]
stock=[]

#Valores
Enero=0
Febrero=0
Marzo=0
Abril=0
Mayo=0
Junio=0
Julio=0
Agosto=0
Septiembre=0
Octubre=0
Noviembre=0
Diciembre=0
y=('')
z=1
```

```
#El codigo empezará con una validación por parte del usuario creando credenciales
#para permitirle el acceso al menú.
#En el menú se le permitirá seleccionar la información que desea ver en la pantalla
#para evitar mostar todas las consignas al mismo tiempo

#<!--  CREDENCIALES  -->

def run():
    print("Registro\n")
    correcto=False
    while correcto==False:                                      #Primero en un ciclo qu
e pedirá
            nombre=input("Ingrese nombre de usuario:\n")        #el nickname hasta ser
validado
            if user.nickname(nombre) == True:
                print("\nUsuario creado con éxito\n")
                correcto=True

    while correcto==True:                                       #Segundo ciclo, esta ve
z solicitando
        contrasenia=input("Ingrese contraseña:\n")              #la contraseña hasta se
r validada
        if passw.clave(contrasenia)==True:
            print("\nContraseña creada con éxito\n")
            correcto=False

    #Inicia el menú para mostrar la información que elegirá el usuario
    print("¡Bienvenido", nombre,"!")
    menu()

#<!--  ALGORITMOS DE ORDENAMIENTO  -->

#<!--  VENTAS  -->

for z in range(97):                                             #CICLO PRINCIPAL PARA RECABAR LA
 INFORMARCION
    x=0                                                         #DE CADA PRODUCTO (96 productos)
. Categoría (y),
    i=0                                                         #ID_Producto (z), compras (i), c
alif (prom)
    r=0                                                         #devoluciones (d), ventas (s) y
mes (m)
    d=0
    s=0
    prom=0
```

```
    #De la lista "lifestore_sales" se obtienen los valores con los que se trabajaran en est
e ciclo

    for row in lifestore_sales:

        valX = lifestore_sales[x][1]                        #Selecciona el Spot 1 de cada li
nea asignado a ID
                                                            #para contar cuantas veces se co
mpró un objeto
        valW = lifestore_sales[x][2]                        #De igual manera el Spot 2 para
las reseñas

        valD = lifestore_sales[x][4]                        #Spot 4 para revisar las devoluc
iones

        valF = lifestore_sales[x][3]                        #Spot 3 para revisar la fecha de
 venta
        valF = valF[3:5]                                    #Solo nos interesa el mes de la
compra

        x+=1

        #Filtro de categorías (Decidí hacerlo con if para no hacer un hardcoding con las li
stas)
        #Se detecta el producto analizado en el loop actual (z), si corresponde, se suma un
a compra,
        #se le asigna su categoría y el mes de compra


        if valX == z:
            if valD == 1:                                   #Revisa si hay alguna devolución
                d-=1
            #Categoría
            if z<10:
                y = 'Procesadores'
            elif z>9 and z<29:
                y = 'Tarjetas de video'
            elif z>28 and z<47:
                y = 'Tarjetas madre'
            elif z>46 and z<60:
                y = 'Discos duros'
            elif z==60 or z==61:
                y = 'Memorias USB'
            elif z>61 and z<74:
                y = 'Pantallas'
```

```
            elif z>73 and z<84:
                y = 'Bocinas'
            elif z>83:
                y = 'Audifonos'

            i+=1                                        #Suma las compras (i), las cal
ificaciones (r)
            r=r+valW                                    #y finalmente, el promedio de
calif (prom)
            prom=r/i

        #Se filtra el mes de compra para saber en que mes se realizaron mas compras
            if valF=='01':
                Enero+=1
            elif valF=='02':
                Febrero+=1
            elif valF=='03':
                Marzo+=1
            elif valF=='04':
                Abril+=1
            elif valF=='05':
                Mayo+=1
            elif valF=='06':
                Junio+=1
            elif valF=='07':
                Julio+=1
            elif valF=='08':
                Agosto+=1
            elif valF=='09':
                Septiembre+=1
            elif valF=='10':
                Octubre+=1
            elif valF=='11':
                Noviembre+=1
            elif valF=='12':
                Diciembre+=1

        #Para evitar mostrar en los peores calificados a los productos que no han tenido co
mpras
        #se detecta si las compras (i) son iguales a Cero (0) para luego eliminarlas de la
lista F
        if i==0:
            prom=0

        #Toplas que asignan las veces que se compra (i) a cada producto (z),
        #categoría (y), calif (prom), devoluciones (d)
        sumaAscendente = [i,z,y]
```

```
        sumaDescendente = [y,i,z]
        sumaE = [y,prom,i,z,d]
        sumaF = [y,prom,i,z,d]
        sumaM = [[Enero,1],[Febrero,2],[Marzo,3],[Abril,4],[Mayo,5],[Junio,6],[Julio,7],[Ag
osto,8],[Septiembre,9],[Octubre,10],[Noviembre,11],[Diciembre,12]]



    #A = [Posición, [veces_comprado, producto, Categoría]]
    #B = [Posición, [Categoría, veces_comprado, producto]]
    #E = [Posición, [Categoría, Calificación, veces_comprado, producto, devoluciones]]
    #F = [Posición, [Categoría, Calificación, veces_comprado, producto, devoluciones]]
    #G = [Posición, [Compras_mes, Mes]]

    #Inserta Topla "SUMA" a las listas A, B, C, D, E, F, G con estas en posicion de cada pr
oducto (z)
    A.insert(z, sumaAscendente)
    E.insert(z, sumaE)
    B.insert(z, sumaDescendente)
    F.insert(z, sumaF)
    G.insert(z, sumaM)

    #Compras hechas de cada producto en posicion del producto (z)
    comprasTotales.insert(z, i)

    if prom == 0:                                           #Si no hay compras del produc
to, no se mostrará
            F.pop()                                         #en los peores calificados (e
vita mostrar 0)


#<!--  BUSQUEDAS  -->

    x=0
    i=0

    #De la lista lifestore_searches obtiene las busquedas por producto

    for row1 in lifestore_searches:

        valY = lifestore_searches[x][1]                     #Revisar el numero de busquedas
 de cada producto
        x+=1

        #Asigna la categoría al producto
        if valY == z:
            if z<10:
```

```
            y = 'Procesadores'

        elif z>9 and z<29:
            y = 'Tarjetas de video'

        elif z>28 and z<47:
            y = 'Tarjetas madre'

        elif z>46 and z<60:
            y = 'Discos duros'

        elif z==60 or z==61:
            y = 'Memorias USB'

        elif z>61 and z<74:
            y = 'Pantallas'

        elif z>73 and z<84:
            y = 'Bocinas'

        elif z>83:
            y = 'Audifonos'

        i+=1

    #Toplas que asignan busquedas (i) al producto (z) de categoría (y)
    sumaSearch = [i,z,y]
    sumaSearchCat = [y,i,z]

#C = [Posición, [veces_buscado, producto, Categoría]]
#D = [Posición, [Categoría, veces_buscado, producto]]

C.insert(z, sumaSearch)

D.insert(z, sumaSearchCat)
z+=1


#<!--  PRECIOS  -->

x=0

#Ciclo para obtener el precio (valP) y stock (valS) de cada producto (x)
for row in lifestore_products:

    valP = lifestore_products[x][2]
    valS = lifestore_products[x][4]
```

```
    x+=1
    preciosTotales.insert(x,valP)
    stock.insert(x, valS)

#Multiplicar compras por precio para saber cual es el aporte de cada producto

comprasTotales.pop(0)                                   #Porque el primero valor es d
el producto 0 (no existe)

#Ciclo para asignar a cada producto (z) el ingreso aportado en el año
H = [[(z+1), preciosTotales[z]*comprasTotales[z], stock[z]] for z in range(len(preciosTotal
es))]

#Ciclo para obtener los ingresos totales en el año
I = [preciosTotales[z]*comprasTotales[z] for z in range(0,len(preciosTotales))]

#Muestra el menú hasta que el usuario seleccione la opción salir (0)

def menu():

    while True:

        print("\n<--  Menú  -->\n ")
        opcionMenu = input(" 1 - 50 Más Comprados                    | 2 - 50 Menos Compra
dos\n 3 - 100 Más Buscados                    | 4 - 100 Menos Buscados\n 5 - 20 Mejor Cali
ficados                | 6 - 20 Peor Calificados\n 7 - Total de Ingresos/Stock (Producto)
  | 8 - Ventas por Mes\n 9 - Total Anual y Promedio Mensual       | 0 - Salir\n\n")

        #<!--  COMPRADOS  -->

        if opcionMenu=='1':
            A.sort(reverse = True)
            print("\n50 Más comprados\n \n Se compró | El producto | Categoría\n")
            print(A[:50])                                                #Mostrar solo los
 primeros 50
            print()

        elif opcionMenu=='2':
            K= sorted(B, key=operator.itemgetter(0,1))
            print("\n50 Menos comprados (Por categoría)\n \n Categoría | Se compró | El pro
ducto\n")
            print(K[1:51])                                               #Mostrar solo los
 primeros 50
            print()

        # <!--  BUSCADOS  -->
```

```
        elif opcionMenu=='3':
            C.sort(reverse= True)
            print("\n100 Mas Buscados\n \n Se Buscó | El producto | Categoría\n")
            print(C[:100])                                          #Mostrar 100
            print()

        elif opcionMenu=='4':
            L= sorted(D, key=operator.itemgetter(0,1))
            print("\n100 Menos Buscados (Por categoría)\n \n Categoría | Se Buscó | El prod
ucto \n")
            print(L[1:101])                                         #Mostrar 100
            print()

        # <!--  RESEÑAS  -->

        elif opcionMenu=='5':
            M= sorted(E, key=operator.itemgetter(4,1,0,2), reverse= True)
            print("\n20 Mejor Calificados (Por categoría)\n \n Categoría | Calificación | S
e compró | El producto | Devoluciones \n")
            print(M[:20])                                           #Mostrar solo los
 primeros 20
            print()

        elif opcionMenu=='6':
            N= sorted(F, key=operator.itemgetter(4,1,0,2), reverse= False)
            print("\n20 Peor Calificados (Por categoría)\n \n Categoría | Calificación | Se
 compró | El producto | Devoluciones \n")
            print(N[1:21])                                          #Mostrar solo los
 primeros 20
            print()

        # <!--  INGRESOS  -->

        elif opcionMenu=='7':
            J= sorted(H, key=operator.itemgetter(1,2,0), reverse= False)
            print("\nIngreso por producto (En el plazo anual)\n \n Producto | Ingreso | Sto
ck\n")
            print(J)
            print()

        elif opcionMenu=='8':
            print("\nVentas por mes (De Enero-Diciembre)\n \n Ventas | Mes \n")
            print(G[-1])
            print()

        elif opcionMenu=='9':
            print("\nTotal ingresos por venta (En el plazo anual)\n")
```

```
            print(sum(I))
            print()

            print("\nPromedio mensual (2020)\n")
            print(sum(I)/12)                                    #Total Anual/12
            print()

        elif opcionMenu=='0':
            run()

        else:
            print("\nPor favor, ingresa una opción válida\n")

#Inicio del programa y procede al menú
run()
```