NOMBRE DE LA ASIGNATURA Análisis de datos con Python (Nivel 1- Fundamentos) PROFESOR (A) Pedro Rotta	FECHA 23/01/2022	TRABAJO N°1 GRUPO 17

Alumnos:

Mondragón Payac Billy

Panta Panta Jimy Aldair

Alberca Elera Gustavo Alonso

Del Carmen Rangel Renzo Daniel

Vida Universitaria

UNIVERSIDAD DE PIURA

2022

1. Fundamentación

El presente trabajo busca poner en práctica los conocimientos aprendidos en el curso: Análisis de datos con Python (Nivel 1- Fundamentos) dentro de un entorno supuesto de una tienda que hemos denominado: “Tu Tienda Perú” sobre la cual se han plateado problemas reales a fin de darles solución y crear un entono efectivo y eficaz de cara a un uso tangible y operativo.

2. Abstract

El presente proyecto permite el funcionamiento administrativo y operativo de la tienda ficticia denominada “Tu Tienda Perú” la cual, para este caso, se dedica a la venta de prendas de vestir dirigidos a Adultos, Jóvenes y Niños. En este supuesto se sabe que la tienda trabaja con 4 hojas Excel las cuales debido a la gran cantidad de datos es casi imposible trabajarlas solamente con Excel y necesita de hojas y trabajo de campo para solventar la lectura de datos y el trabajo general de la empresa. Para resolver este problema, proponemos el uso del lenguaje de programación Python con el fin crear un programa dinámico e intuitivo que integre estas hojas y separe las funciones y datos en las 2 áreas más importantes de la empresa: Área administrativa y Área de almacén. Además, buscamos que este programa permita tomar decisiones de mercado al mostrar gráficas e histogramas de ventas y compras realizadas con el fin de ayudar al área de gerencia y efectivizar el trabajo y ser eficaces con la información interna que maneja la tienda en cuestión. Con este panorama, finalmente, buscamos solucionar un problema recurrente dentro de las pequeñas empresas el cual es la separación de áreas y la protección de datos internos que impiden la correcta contabilidad de material vendido y comprado. Tras la realización de este proyecto se ha podido utilizar librerías nuevas encontradas en línea, gracias a la comunidad, y uso de las librerías aprendidas en clase.

3. Objetivos

a. Objetivos especifico

Como objetivo principal buscamos integrar la cantidad de datos que utiliza la tienda Tu Tienda Perú mediante un software basado en Python de manera que permita llevar un orden contable. Esto se logra poniendo en práctica las clases recibidas dentro del curso de verano Python básico 2022.

b. Segundo objetivo

Separar las áreas de trabajo más demandantes como: el área administrativa y el área de almacén, y agilizar el trámite laboral de cada uno de los trabajadores.

c. Tercer objetivo

Crear una herramienta que ayude al área de gerencia a tomar mejores decisiones de mercado y le permita llevar un orden bajo una vista macro de todas y cada una de las acciones que se ejecuten en las áreas más importantes de la empresa.

d. Cuarto objetivo

Separar las áreas de trabajo y proteger los datos internos de la empresa a fin de independizar las áreas de trabajo y fortaleciendo responsabilidades internas como muestra de buena operatividad.

4. Metodología

Se han utilizado las diferentes librerías vistas en clase como: *numpy, pandas, myplotlib*, *getpass, myplotlib.pylab*; condicionales como *if, else, elif*; bucles: *for y while;* métodos: *plot, subplot, statter,array, plt.style.use, plt.hist, display , loc, apped, columns, read_excel, ax.set_xlabel, plt.sytle, plt.grid, plt.show, etc.*

Se utilizó la plataforma *colab.research* de Google para la programación del lenguaje Python del programa.

Se hicieron uso de 4 archivos Excel que tuvieron la función de base para la obtención y uso de datos para el proyecto: "Inventarios_Proyect2022"; "Salida_Articulos_Proyec2022"; "Stock_Actualizado_Proyec2022"; y "Trabajadores_Pagos_Mensualidad_Proyect2022".

5. Funcionamiento

5.1 Tipos de usuario

Debido a la constitución interna de la empresa que se ha considerado para la digitalización de sus funciones, se han tomado a 2 usuarios importantes que utilizarán el siguiente programa realizado en Python. Los 2 usuarios se dividen en: Administradores y Trabajadores de almacén, ambos tienen roles y funciones completamente distintas pero complementarias que sirven mucho para el funcionamiento operativo de la empresa.

Primero denominamos a los usuarios con sus respectivas claves, esto permitirá el acceso a las diferentes funciones del programa. Para efectos prácticos se ha escogido un usuario para cada tipo de trabajador: un usuario para Administrador con la variable y nombre de logueo: "__usuario1":"admin" cuya contraseña en variable y nombre es: "__clave1": "admin123". Finalmente, para los trabajadores de almacén se ha denominado un usuario con la variable y nombre de logueo: "__usuario2":"1234" cuya contraseña en variable y nombre es: "__clave1": "1234".

```
#VARIABLES GLOBALES
#Clave y usuario_Area Logistica_Administrativa
__usuario1= "admin"
__clave1 = "admin123"
#Clave y usuario_Area Almacen
__usuario2="1234"
__clave2="1234"

#Variables globales
inventario_df=""
descarga_df=""
stockActualizado_df=""

print("BIENVENIDO AL SISTEMA VIRTUAL ADMINISTRATIVO DE TUTIENDAPERU")
print(10*"-",'SOLO ACCESO PARA PERSONAL AUTORIZADO','-'*10)
```

Se definió una función "main()" baja la cual se despliegan 3 opciones:

a.- Loguearse como Administrador

b.- Loguearse como Almacén

c.- Salir

```
#Menu Principal Inicial
def main():
    print ("\nOpciones de Exec de Usuario")
    print ("\t1 - Logearse Administrador")
    print ("\t2 - Logearse Almacen")
    print ("\t3 - Salir")
```

5.1.1 Loguearse como Administrador

Dentro de un bucle *while* se llama a la función *main()* que revisa, bajo la condición *if,* si se ha escogido Loguearse como Administrador. Se realiza la comparativa de si tanto __usuario1 y __clave1 coinciden ambas y al mismo tiempo para desplegar los diferentes menús que corresponden.

```
while True:
  main()
  eleccion=input("Inserta una Opción: ")
  if eleccion == "1":
    usuario=input("Ingrese su usuario: ")
    clave=getpass.getpass("Ingrese su contraseña: ")
    if usuario== __usuario1 and clave==__clave1 :
        menu_administracion()
    else:
        print("\nDATOS INGRESADOS INCORRECTOS (SI NO RECUERDA SUS DATOS COMUNIQUESE CON SU AREA RESPECTIVA)")
        input("\nPulsa 'START' para continuar...")
  elif eleccion=="2":
      usuario2=input("Favor Ingrese su usuario: ")
      clave2=getpass.getpass("Favor Ingrese su contraseña: ")
      if usuario2== __usuario2 and clave2==__clave2 :
        menu_almacen()
      else:
        print("\nDATOS INGRESADOS INCORRECTOS (SI NO RECUERDA SUS DATOS COMUNIQUESE CON SU ENCARGADO)")
  elif eleccion=="3":
        print("GRACIAS POR SU TIEMPO")
        break
  else:
        print("NO INGRESO UNA OPCION CORRECTA")
```

Para agregar seguridad al código y efectivizar su uso se encriptó la escritura de la clave utilizando la librería "*getpass*"

```
usuario2=input("Favor Ingrese su usuario: ")
clave2=getpass.getpass("Favor Ingrese su contraseña: ")
```

5.1.1.1 Privilegios

El administrador tiene privilegios de acceso a unas opciones de menú que le permite a este modificar de forma sustancial información que se ingrese en beneficio de la empresa. Siguiendo su rol dentro del esquema de trabajo de "Tu Tienda Perú", el administrador posee un privilegio de ingresar compras realizadas por la empresa (para su abastecimiento de mercadería); visualizar comparativas gráficas que permiten la toma de decisiones de la mano con gerencia, visualizar comparativas de ventas por fechas y ganancias por fechas (estas últimas con la opción de obtener histogramas independientes cuya finalidad es la de conocer tendencias en compras y ventas para poder tomar decisiones basados en el comportamiento del consumidor de la empresa misma en la adquisición de productos para su venta; y, finalmente visualizar pagos mensuales de los colaboradores de la empresa

5.1.1.2 Acciones

5.1.1.2.1 Login

El administrador necesita ingresar “admin” en la solicitud Usuario y “admin123” en la solicitud de contraseña.

5.1.1.2.2 Despliegue de menú

Si coinciden tanto el nombre de usuario como la contraseña, se despliega un menú que desarrolla los privilegios anteriormente mencionados para el administrador. Cada opción le brinda al administrador la opción de regresar a menús previos, edición de datos según donde se encuentre y funcionalidades interactivas que agilice su trabajo.

El menú mostrado es el siguiente:

1.- Visualizar Stock existente

2.-Ingresar nuevas Compras (inventario)

3.-Información de pago mensual a los colaboradores

4.-Cuadros gráficos

5.-Salir

```
def menu_administracion() :
  global stockActualizado_df
  global inventario_df
  print("")
  print(10*'-',"BIENVENIDO A LA LISTA DE OPCIONES DE ADMINISTRADOR",'-'*10)
  eleccion_administracion=input("""
  \t1 - VISUALIZAR STOCK EXISTENTE
  \t2 - INGRESAR NUEVAS COMPRAS(INVENTARIADO)
  \t3 - INFORMACION PAGO MENSUAL A LOS TRABAJADORES
  \t4 - CUADROS GRAFICOS
  \t5 - SALIR
  \nINSERTE UNA OPCIÓN: """)
  if eleccion_administracion == "1":
          stock_actualizado()
          input("\nPRESIONE 'START' PARA CONTINUAR...")
          menu_administracion()
  elif eleccion_administracion=="2":
          inventariado_articulos()
          inventario_articulos_submenu()
          input("\nPRESIONE 'START' PARA CONTINUAR: ")
  elif eleccion_administracion=="3":
          Informacion_Mensualidad()
          input("\nPRESIONE 'START' PARA CONTINUAR: ")
          menu_administracion()
  elif eleccion_administracion=="4":
          graficos()
          input("\nPRESIONE 'START' PARA CONTINUAR: ")
          menu_administracion()
  elif eleccion_administracion=="5":
          print("GRACIAS")
  else:
        input("NO INGRESASTE UNA OPCION CORRECTA...\nPulsar una tecla para continuar: ")
        menu_administracion()
```

5.1.1.2.3 Funcionamiento del menú

5.1.1.2.3.1 Visualizar Stock existente

Este apartado permite visualizar a modo de tabla una lista detallada de artículos existentes hasta la fecha actual en la tienda Tu Tienda Perú. En la tabla se aprecia la descripción del artículo, el tipo de artículo, el stock (cantidad que se tiene de cada uno de los diferentes productos) y su ubicación dentro del almacén de la empresa. Se utilizó una variable global llamada *stockactualizado_df* y el comando *read_excel* para enlazar el archivo Excel original a utilizar con el programa en Python.

```
def stock_actualizado():
  global stockActualizado_df
  ruta_stock= '/content/Stock_Actualizado_Proyec2022.xlsx'
  stockActualizado_df=pd.read_excel(ruta_stock)
  display(stockActualizado_df)
  filtro_stock()
```

Una vez presentada la tabla se despliegan 2 opciones bajo la función *"filtro_stock()"* que le permiten al administrador filtrar los resultados por categoría de producto. Tenemos 3 grandes categorías que utiliza la empresa Tu Tienda Perú: Productos para Adultos, Productos para Jóvenes y Productos para Niños.

5.1.1.2.3.1.1 Filtrar por categoría Adulto, Joven, Niño

Esta opción permite filtrar por categoría las diferentes prendas según correspondan al público objetivo que van dirigidos: Adultos, Jóvenes y Adultos. Se toma en consideración el stock actualizado.

Se hizo uso del método *"df.loc"* de la mano de la variable global *stockActualizado_df* para filtrar según sea la consulta que se realice. Es necesario que el administrador tipee en mayúsculas ya sea "ADULTO", "JOVEN", "NIÑO" y el programa mostrará una tabla detallada con cada uno de los artículos disponibles según sea la categoría a la que pertenezcan

```
#FILTRO DE STOCK PARA AMBOS CASOS ADMINITRADOR Y ALMACEN , CORRESPONDIENTE A LA OPCION 1 DE SUS MENU(s)
def filtro_stock():
  global stockActualizado_df
  print("")
  filtro_stock=input("""QUE ACCIÓN DESEA REALIZAR:
  \t1 - FILTRAR
  \t2 - SALIR
  \nSELECCIONE UNA OPCION: """)
  if filtro_stock == "1":
    print("POR EL MOMENTO SOLO SE PUEDE FILTRAR LOS ARTÍCULOS POR 'CATEGORIA'")
    stockActualizado_df.describe()
    respuesta_filtro =stockActualizado_df.loc[stockActualizado_df['TIPO'] == input("""\n-JOVEN\n-ADULTO\n-NIÑO
    \nINGRESE PALABRA A FILTRAR: """)]
    display(respuesta_filtro)
  elif filtro_stock == "2":
    menu_administracion()
```

5.1.1.2.3.1.2 Salir

Esta opción nos regresa al menú principal de administrador, como se observa *menú_administracion()* está declarado al final de la función *filtro_stock()*

5.1.1.2.3.2 Ingresar nuevas compras

5.1.1.2.3.2.1 Agregar artículos

```
#SUB MENU PERTENECIENTE A LA VISUALIZACION DEL ADMINISTRADOR (OPCION 2 , DEL MENU_ADMINISTRACION)
def inventario_articulos_submenu():
  global inventario_df
  eleccion_inventario=input("""
  \t1 - AGREGAR ARTICULOS
  \t2 - MOSTRAR LISTA DE ARTICULOS AGREGADOS
  \t3 - FILTRAR ARTICULOS AGREGADOS POR FECHA
  \t4 - SALIR
  \n INSERTE UNA OPCION: """)
  if eleccion_inventario == "1":
    agregar_fila_inventario()
    inventario_articulos_submenu()
  elif eleccion_inventario == "2":
    display(inventario_df)
    inventario_articulos_submenu()
  elif eleccion_inventario == "3":
    filtro_ingresos()
  elif eleccion_inventario == "4":
    menu_administracion()
  else:
     print("OPCION INCORRECTA")
```

El administrador podrá ingresar los artículos que la empresa compró para abastecerse en favor de sus ventas, es decir, para el llenado de su stock.

El administrador al ser un usuario con un nivel superior de privilegios, puede agregar ´pr columnas información necesaria como: Artículos, Precio, Fecha de adquisición, Cantidad y Gasto de la compra.

Las librería utilizada ha sido *pandas*, con ella se le ha llamado *as pd* para agregar bajo el comando *append* los diferentes datos por artículo. Para ello se ha creado un diccionario *fila* y dentro de esta la función *columnas*.

Se creó la función *agregar_fila_inventario()* invocando la variable global *inventario_df().* Creando un diccionario vacío con el nombre *"dicc_fila"*. Dentro de este diccionario almacenamos los datos una variable llamada *"columnas_df*", y en esta última relacionamos los datos por columnas creados en el Archivo Excel, los cuales son: *Artículos, precio, fecha, cantidad y gasto compra*.

Creando un controlador de flujo *for columnas in columnas_df:* que permite almacenar por columnas cada uno de los datos ingresados, estos se irán almacenando los datos dentro de la variable global *"inventario_df"*. Finalmente, el administrador podrá visualizar todos los datos ingresados a manera de tabla y de forma detallada.

```
#AGREGAR FILAS AL EXCEL INVENTARIADO_AREA ADMINISTRACION, PERTENECIENTE AL SUB MENU "INVENTARIADO_DE_ARTICULOS" DE MENU ADMINISTRACION
def agregar_fila_inventario():
  global inventario_df
  dicc_fila={}
  columnas_df=Columns(inventario_df)
  display(inventario_df)
  for columna in columnas_df:
    dicc_fila[columna] = input('Ingrese valor de la columna {} :'.format(columna))
  inventario_df = inventario_df.append(dicc_fila, ignore_index = True)
  display(inventario_df)
  #inventario_df.to_excel('/content/Inventarios_Proyect2022.xlsx')
```

Finalizada la visualización se muestran nuevamente, al pie de la tabla, el menú correspondiente a la función *inventario_artículos_submenu()* y una opción para volver al menú principal del administrador.

5.1.1.2.3.2.2 Filtrar por fecha

El programa utiliza la variable global *"inventario_df"* para filtrar solo por fecha con el objetivo de tener a la mano la visualización de un histórico de ingresos. Se creó una función llamada *filtro_ingresos()* de la cual se despliega un sub menú con 2 opciones : filtrar y salir (regresar al menú

anterior) en la opción 1 solo permite realizar el filtro por fecha. Utilizando el comando *"display"* se mostrará el encabezado haciendo uso de la variable global y el comando: "*inventario_df.head(1)"*. El programa, dentro de una variable local llamada *"respuesta_filtro_inventario"* se usa el método *".loc."* para que el administrador filtrar valores de un *data frame*, en este caso será la información de la columna *"Fecha*" bajo un input *"ingrese la fecha a filtrar*". El resultado es una tabla ordenada de manera ordenada por indexación todos los artículos ingresados en la fecha filtrada.

```
#FILTRO INGRESOS_ PERTENECIENTE AL ADMINISTRADOR
def filtro_ingresos():
  global inventario_df
  filtro_inventario=input("""\nSECCION FILTROS POR FECHA, QUE ACCIÓN DESEA REALIZAR:
  \t1 - FILTRAR
  \t2 - SALIR
  \nSELECCIONE UNA OPCION: """)
  if filtro_inventario == "1":
    print("POR EL MOMENTO SOLO SE PUEDE FILTRAR POR 'FECHA', SIGA EL FORMATO 'AÑO-MES-DIA', ASÍ COMO SE MUESTRA EN EL CUADRO")
    display(inventario_df.head(1))
    inventario_df.describe()
    respuesta_filtro_inventario =inventario_df.loc[inventario_df['FECHA'] == input("""
    \nINGRESE LA FECHA A FILTRAR: """)]
    display(respuesta_filtro_inventario)
    filtro_ingresos()
  elif filtro_inventario == "2":
    inventario_articulos_submenu()
```

Finalmente, el programa permite regresar al menú declarado bajo la función "*inventario_artículos_submenu()"* que hace referencia al menú en programa *Ingresar nuevas compras.*

5.1.1.2.3.2.3 Salir

Esta opción nos regresa al menú anterior.

5.1.1.2.3.3 Información pago mensual a los trabajadores

Esta opción dentro del programa nos redirige a la función denominada *"información_mensualidad()* esta función asignamos una variable la ruta Excel de nuestro contenido de información sobre el pago de los trabajadores.
Se agrega una ruta definida y se utiliza la librería *pandas* para poder leer el Excel (*pd.read_excel* ('ruta del archivo)). La ruta del archivo hace referencia al Excel con el nombre *Trabajo_Pagos_Mensualidad_Proyect2022.*
Acto seguido se imprime la tabla con el comando *"display"* y retorna al *menú_admistración*() para seguir realizando consultas.

```
#VISUALIZACION DE PAGOS MENSUALES A LOS COLABORADORES_ADMINISTRADOR
def Informacion_Mensualidad():
  ruta_pagos= '/content/Trabajadores_Pagos_Mensualidad_Proyect2022.xlsx'
  pago_mensualidad_df=pd.read_excel(ruta_pagos)
  display(pago_mensualidad_df)
```

5.1.1.2.3.4 Cuadros- Gráficos

Esta opción despliega un submenú que llama a la función llamada *graficos()* en donde se despliegan las opciones *graficos* , *hisotigramas* y *salir*

5.1.1.2.3.4.1 Gráficos

En este apartado se han designado 2 funciones, la primera función es *"salida_articulos"* y la segunda *"grafico_generales"*

5.1.1.2.3.4.1.1 Salida de artículos

salida_articulos: esta función asignamos una variable a nuestra ruta a nuestro Excel denominado *"Salida_Articulos_Proyec2022.xlsx"*. Leemos el archivo mediante el comando *pd.read_excel*. Una vez leída la información procedemos a ejecutar de forma inmediata la función *"grafico_generales"*

```
#GRAFICOS DEL SUBMENU_GRAFICOS_ADMINISTRADOR , OPCION 1 DEL SUB MENU 'GRAFICOS()' DE 'MENU_ADMINISTRACION' , OPCION 4:>1
def grafico_generales():
  print("\nLISTA DE OPCIONES DE GRAFICOS")
  global descarga_df
  descarga_cliente = list(descarga_df["CLIENTE"])
  descarga_cantidad = list(descarga_df['DNI'])
  descarga_articulo = list(descarga_df['ARTICULOS'])
  descarga_fecha = list(descarga_df['FECHA'])
  descarga_cantidad = list(descarga_df['CANTIDAD'])
  descarga_precio = list(descarga_df['PRECIO'])
  descarga_ganancia = list(descarga_df['GANANCIA VENTA'])
  descarga_cantidad_escalada = 1*np.array(descarga_cantidad)
  eleccion_grafico =input("""
  \t1 - GRAFICO RELACION GANANCIA - CANTIDAD DE ARTICULOS
  \t2 - GRAFICO CANTIDAD VENDIDA POR FECHA
  \t3 - SALIR
  \n - SELECCIONE UNA OPCION: """)
  if eleccion_grafico == "1":
    print("")
    plt.style.use('seaborn-white')
    fig,ax=subplots()
    plt.scatter(descarga_ganancia,descarga_cantidad_escalada, marker= 'o',color ='blue')
    ax.set_xlabel('Valor de articulos vendidos')
    ax.set_ylabel('Cantidad de veces')
    ax.set_title("GRAFICO RELACION GANANCIA - CANTIDAD DE ARTICULOS\n")
    plt.grid()
    plt.show()
    input("\nPRESIONE 'START' PARA CONTINUAR")
    grafico_generales()
  elif eleccion_grafico == "2":
    print("")
    plt.style.use('seaborn-white')
    fig,ax = subplots()
    plt.scatter(descarga_ganancia,descarga_fecha, marker= 'o',color ='red')
    ax.set_xlabel('Ganancia')
    ax.set_ylabel('Fechas')
    ax.set_title("GRAFICO RELACION FECHA - GANANCIA\n")
    plt.grid()
    plt.show()
    input("\nPRESIONE 'START' PARA CONTINUAR")
    grafico_generales()
  elif eleccion_grafico == "3":
    graficos()
  else:
    print("INGRESASTE UNA OPCION INCORRECTA")
    grafico_generales()
```

5.1.1.2.3.4.1.1.1 Gráficos generales

Se despliegan las 2 opciones principales: Graficos ganancia por fecha y Graficos cantidad vendida por fecha.

5.1.1.2.3.4.1.1.2 Gráficos Ganancia por fecha

Para hacer uso de los gráficos de dispersión utilizaremos el método *"plot"* y método *"scatter"*. Se usó la librería *matplotlib* para poder realizar las etiquetas de las axisas con el método *subplots()*. Usando el *plt.scatter* mostraremos el gráfico de dispersión para unas determinadas variables, en nuestro caso serán *"descarga_ganancia"* y *"descarga_cantidad_escalada". descarga _ganacia* hace referencia a la columna *Ganacia* del Archivo Excel *"Salida_Artículos_Proyect2022"* y

“descarga_cantidad_escalada” refiere a la columna cantidad la cual ha sido escalada previamente con la librería *numpy.array* de la siguiente manera *descarga_cantidad_escalada = 1*np.array(descarga_cantidad*). Se ha hecho uso de etiquetas dentro de nuestro gráfico de dispersión para los ejes.

```
descarga_ganancia = list(descarga_df['GANANCIA VENTA'])
descarga_cantidad_escalada = 1*np.array(descarga_cantidad)
eleccion_grafico =input("""
```

```
if eleccion_grafico == "1":
  print("")
  plt.style.use('seaborn-white')
  fig,ax=subplots()
  plt.scatter(descarga_ganancia,descarga_cantidad_escalada, marker= 'o',color ='blue')
  ax.set_xlabel('Valor de articulos vendidos')
  ax.set_ylabel('Cantidad de veces')
  ax.set_title("GRAFICO RELACION GANANCIA - CANTIDAD DE ARTICULOS\n")
  plt.grid()
  plt.show()
  input("\nPRESIONE 'START' PARA CONTINUAR")
  grafico_generales()
```

Seguido hemos usado los comandos *plt.grid(),* para que el gráfico posea un fondo de hoja cuadriculada, y *plt.show()* para mostrar el grafico de dispersión de relación dispersión ganancia – cantidad de artículos. Una vez mostrado el gráfico regresamos a nuestro menú anterior el cual se denomina *“graficos_generales()”.*

5.1.1.2.3.4.1.1.3 Gráficos cantidad vendida por fecha

Para hacer uso de los gráficos de dispersión utilizaremos el método *“plot”* y método “*scatter”*. Se usó la librería *matplotlib* para poder realizar las etiquetas de las axisas con el método *subplots().* Usando el *plt.scatter* mostraremos el gráfico de dispersión para unas determinadas variables, en nuestro caso serán *“descarga_ganancia”* y *“descarga_fecha”*. *Descarga _ganacia* hace referencia a la columna Ganancia del Archivo Excel *“Salida_Artículos_Proyect2022*” y *“descarga_fecha”* refiere a la columna *fecha.* Se ha hecho uso, además, de etiquetas dentro de nuestro gráfico de dispersión para los ejes.

```
elif eleccion_grafico == "2":
  print("")
  plt.style.use('seaborn-white')
  fig,ax = subplots()
  plt.scatter(descarga_ganancia,descarga_fecha, marker= 'o',color ='red')
  ax.set_xlabel('Ganancia')
  ax.set_ylabel('Fechas')
  ax.set_title("GRAFICO RELACION FECHA - GANANCIA\n")
  plt.grid()
  plt.show()
  input("\nPRESIONE 'START' PARA CONTINUAR")
  grafico_generales()
elif eleccion_grafico == "3":
  graficos()
else:
  print("INGRESASTE UNA OPCION INCORRECTA")
  grafico_generales()
```

Seguido hemos usado los comandos *plt.grid(),* para que el gráfico posea un fondo de hoja cuadriculada, y *plt.show()* para mostrar el grafico de dispersión de relación fecha – ganancia. Una

vez mostrado el gráfico regresamos a nuestro menú anterior el cual se denomina *"graficos_generales()".*

5.1.1.2.3.4.1.1.4 Salir

Nos regresa al menú anterior de Gráficos.

5.1.1.2.3.4.2 Histogramas

Este apartado nos permite ver las opciones basados en histogramas los cueles son: Histograma relación ganancia por valor en ventas y Histograma relación de cantidad de artículos por precio.

```
#HISTOGRAMAS VISUALIZACION DE AREA ADMINISTRACION , OPCION 2 DEL SUB MENU GRAFICOS() DE MENU_ADMINISTRACION , OPCION 4:>2
def histograma_graficos():
  print("\nLISTA DE OPCIONES DE HISTOGRAMA")
  global descarga_df
  descarga_cliente = list(descarga_df["CLIENTE"])
  descarga_cantidad = list(descarga_df['DNI'])
  descarga_articulo = list(descarga_df['ARTICULOS'])
  descarga_fecha = list(descarga_df['FECHA'])
  descarga_cantidad = list(descarga_df['CANTIDAD'])
  descarga_precio = list(descarga_df['PRECIO'])
  descarga_ganancia = list(descarga_df['GANANCIA VENTA'])
  eleccion_histograma =input("""
  \t1 - HISTOGRAMA RELACION DE GANANCIA POR VALOR EN VENTAS
  \t2 - HISTOGRAMA RELACION DE CANTIDAD DE ARTICULOS POR PRECIO
  \t3 - SALIR
  \n - SELECCIONE UNA OPCION: """)
  if eleccion_histograma == "1":
    print("")
    print(5*'-',"HISTOGRAMA RELACION DE GANANCIA POR VALOR EN VENTAS",5*'-')
    plt.style.use('seaborn-white')
    descarga_ganancia_arr = np.array(descarga_ganancia)
    plt.hist(descarga_ganancia_arr, bins = 5, alpha = 1, histtype='bar', color='steelblue', edgecolor = 'black')
    plt.show()
    input("\nPRESIONE 'START' PARA CONTINUAR")
    histograma_graficos()
  if eleccion_histograma == "2":
    print("")
    print(5*'-',"HISTOGRAMA RELACION DE CANTIDAD DE ARTICULOS POR PRECIO",5*'-')
    plt.style.use('seaborn-white')
    descarga_precio_arr = np.array(descarga_precio)
    plt.hist(descarga_precio_arr, bins = 5, alpha = 1, histtype='bar', color='steelblue', edgecolor = 'black')
    plt.show()
    input("\nPRESIONE 'START' PARA CONTINUAR")
    histograma_graficos()
  elif eleccion_histograma == "3":
    graficos()
  else:
    print("INGRESASTE UNA OPCION INCORRECTA")
    histograma_graficos()
```

5.1.1.2.3.4.2.1 Histograma Relación ganancia por valor en ventas

Utilizamos el método *plt.style. use('seaborn-white')* para definir el estilo a implementar. Seguido se usa la librería *numpy* de la forma *np.array* para asignar a una variable *"descarga_ganancia_arr"* para poder graficar. Procedemos a realizar la siguiente sintaxis para poder plotear el histograma: *plt.hist(descarga_ganancia_arr, bins = 5, alpha = 1, histtype='bar', color='steelblue', edgecolor = 'black')*. Seleccionamos el *bins =5* para no visualizar espacios vacíos en la gráfica, sirviéndonos como el número de secciones en los que se dividirán los datos. *Alpha* ajustamos la transparencia del gráfico *Histtype* seccionamos el modo de barra por ser el más idóneo visualmente para recoger este tipo de datos. Finalmente, mostramos en pantalla el histograma con *plt.show* y retornamos al menú anterior llamado *"histograma_graf".*

```
if eleccion_histograma == "1":
  print("")
  print(5*'-',"HISTOGRAMA RELACION DE GANANCIA POR VALOR EN VENTAS",5*'-')
  plt.style.use('seaborn-white')
  descarga_ganancia_arr = np.array(descarga_ganancia)
  plt.hist(descarga_ganancia_arr, bins = 5, alpha = 1, histtype='bar', color='steelblue', edgecolor = 'black')
  plt.show()
  input("\nPRESIONE 'START' PARA CONTINUAR")
  histograma_graficos()
```

5.1.1.2.3.4.2.2 Histograma Relación de cantidad de artículos por precio

Utilizamos el método *plt.style. use('seaborn-white')* para definir el estilo a implementar. Seguido se usa la librería *numpy* de la forma *np.array* para asignar a una variable *"descarga_precio_arr"* para poder graficar. Procedemos a realizar la siguiente sintaxis para poder plotear el histograma: *plt.hist(descarga_precio_arr, bins = 5, alpha = 1, histtype='bar', color='steelblue', edgecolor = 'black')*. Seleccionamos el *bins =5* para no visualizar espacios vacíos en la gráfica, sirviéndonos como el número de secciones en los que se dividirán los datos. *Alpha* ajustamos la transparencia del gráfico *Histtype* seccionamos el modo de barra por ser el más idóneo visualmente para recoger este tipo de datos. Finalmente mostramos en pantalla el histograma con *plt.show* y retornamos al menú anterior llamado *"histograma_graf".*

```
if eleccion_histograma == "2":
  print("")
  print(5*'-',"HISTOGRAMA RELACION DE CANTIDAD DE ARTICULOS POR PRECIO",5*'-')
  plt.style.use('seaborn-white')
  descarga_precio_arr = np.array(descarga_precio)
  plt.hist(descarga_precio_arr, bins = 5, alpha = 1, histtype='bar', color='steelblue', edgecolor = 'black')
  plt.show()
  input("\nPRESIONE 'START' PARA CONTINUAR")
  histograma_graficos()
```

5.1.1.2.3.4.3 Salir

Nos permite regresar a las opciones de gráficos.

```
elif eleccion_histograma == "3":
  graficos()
```

5.1.1.2.3.5 Salir

Nos envía a las opciones de histograma

```
else:
  print("INGRESASTE UNA OPCION INCORRECTA")
  histograma_graficos()
```

5.1.2 Loguearse como Almacén

Dentro de un bucle *while* se llama a la función *main()* que revisa, bajo la condición *if,* si se ha escogido Loguearse como Administrador. Se realiza la comparativa de si tanto __usuario2 y __clave2 coinciden ambas y al mismo tiempo para desplegar los diferentes menús que corresponden.

```
elif eleccion=="2":
    usuario2=input("Favor Ingrese su usuario: ")
    clave2=getpass.getpass("Favor Ingrese su contraseña: ")
    if usuario2== __usuario2 and clave2==__clave2 :
      menu_almacen()
    else:
      print("\nDATOS INGRESADOS INCORRECTOS (SI NO RECUERDA SUS DATOS COMUNIQUESE CON SU ENCARGADO)")
```

Para agregar seguridad al código y efectivizar su uso se encriptó la escritura de la clave utilizando la librería “*getpass*”

5.1.2.1 Privilegios

E almacenero al poseer menos funciones privilegiadas dentro de la empresa, solo podrá acceder a opciones de contabilidad de productos del almacén.

Solo visualizará el stock manera de tabla para que haya coordinación en cuanto a cantidad de productos en relación con la información que maneje el administrador, y podrá ingresar la salida de artículos cada que se cierre una venta.

```
def menu_almacen():
  global descarga_df
  print("")
  print(10*'-',"BIENVENIDO A LA LISTA DE OPCIONES DE USUARIO",'-'*10)
  eleccion_almacen=input("""
  \t1 - VISUALIZAR STOCK ACTUALIZADO
  \t2 - SALIDA DE ARTICULOS
  \t3 - SALIR
  \nINSERTE UNA OPCION :""")

  if eleccion_almacen == "1":
          stock_actualizado()
          input("\nPRESIONE 'START' PARA CONTINUAR: ")
          menu_almacen()
  elif eleccion_almacen=="2":    #mostrar el sub menu
          salida_articulos()
          descarga_articulos_submenu()
          input("\nPRESIONE 'START' PARA CONTINUAR: ")
  elif eleccion_almacen=="3":
          print("")
  else:
          input("NO INGRESASTE UNA OPCION CORRECTA...\nPulsar una tecla para continuar: ")
```

5.1.2.2 Acciones

5.1.2.2.1 Login

El personal de almacén cuenta con un cifrado de la siguiente forma: necesita ingresar “1234” en la solicitud Usuario y “1234” en la solicitud de contraseña.

5.1.2.2.2 Despliegue de menú

Si coinciden tanto el nombre de usuario como la contraseña, se despliega un menú que desarrolla los privilegios anteriormente mencionados para el trabajador de almacén. Cada opción le brinda al personal de almacén la opción de regresar a menús previos, edición de datos según donde se encuentre y funcionalidades interactivas que agilice su trabajo.

El menú mostrado es el siguiente:

1.- Visualizar Stock actualizado
2.-Salida de Artículos
3.-Salir

5.1.2.2.3 Funcionamiento del menú

```
################### MENU ALMACEN Y SUS FUNCIONES ################################
def menu_almacen():
  global descarga_df
  print("")
  print(10*'-',"BIENVENIDO A LA LISTA DE OPCIONES DE USUARIO",'-'*10)
  eleccion_almacen=input("""
  \t1 - VISUALIZAR STOCK ACTUALIZADO
  \t2 - SALIDA DE ARTICULOS
  \t3 - SALIR
  \nINSERTE UNA OPCION :""")

  if eleccion_almacen == "1":
          stock_actualizado()
          input("\nPRESIONE 'START' PARA CONTINUAR: ")
          menu_almacen()
  elif eleccion_almacen=="2":    #mostrar el sub menu
          salida_articulos()
          descarga_articulos_submenu()
          input("\nPRESIONE 'START' PARA CONTINUAR: ")
  elif eleccion_almacen=="3":
          print("")
  else:
          input("NO INGRESASTE UNA OPCION CORRECTA...\nPulsar una tecla para continuar: ")
```

5.1.2.2.3.1 Visualizar Stock actualizado

Este apartado permite visualizar a modo de tabla una lista detallada de artículos existentes hasta la fecha actual en la tienda Tu Tienda Perú. En la tabla se aprecia la descripción del artículo, el tipo de artículo, el stock (cantidad que se tiene de cada uno de los diferentes productos) y su ubicación dentro del almacén de la empresa. Se utilizó una variable global llamada *stockactualizado_df* y el comando *read_excel* para enlazar el archivo Excel original a utilizar con el programa en Python.

```
def stock_actualizado():
  global stockActualizado_df
  ruta_stock= '/content/Stock_Actualizado_Proyec2022.xlsx'
  stockActualizado_df=pd.read_excel(ruta_stock)
  display(stockActualizado_df)
  filtro_stock()
```

Una vez presentada la tabla se despliegan 2 opciones bajo la función *"filtro_stock()"* que le permiten al administrador filtrar los resultados por categoría de producto. Tenemos 3 grandes categorías que utiliza la empresa Tu Tienda Perú: Productos para Adultos, Productos para Jóvenes y Productos para Niños.

5.1.2.2.3.2 Salida de Artículos

El trabajador de almacén puede editar y utilizar el Excel llamado *Salida_Artículos_Proyec2022*

```
#RUTA DE DESCARGA DE ARTICULOS_AREA DE VISUALIZACION DE ALMACEN
def salida_articulos():
  global descarga_df
  print("\n DESCARGA DE ARTICULOS PARA TUTIENDAPERU")
  ruta_descarga= '/content/Salida_Articulos_Proyec2022.xlsx'
  descarga_df=pd.read_excel(ruta_descarga)
  Columns(descarga_df)
```

Inmediatamente se despliega un submenú

1 agrega nuevo artículo

2Mostrar lista de artículos vendidos

3Filtrar artículos vendidos

4salir

5.1.2.2.3.3 Agregar nuevo artículo

El trabajador de almacén podrá ingresar los artículos que la empresa vendió y compró para abastecerse en favor de sus ventas, es decir, para el llenado de su stock.
El trabajador de almacén solo tiene este privilegio superficial, puede agregar ´pr columnas información necesaria como: Artículos, Precio, Fecha de adquisición, Cantidad y Gasto de la compra.
Las librería utilizada ha sido *pandas*, con ella se le ha llamado *as pd* para agregar bajo el comando *append* los diferentes datos por artículo. Para ello se ha creado un diccionario *fila* y dentro de esta la función *columnas*.
Se creó la función *descraga_artículos()* invocando la variable global *descarga_df().* Creando un diccionario vacío con el nombre *"dicc_fila1"*. Dentro de este diccionario almacenamos los datos una variable llamada *"columnas_df"*, y en esta última relacionamos los datos por columnas creados en el Archivo Excel, los cuales son: *Artículos, precio, fecha, cantidad y gasto compra*.
Creando un controlador de flujo *for columnas in columnas_df:* que permite almacenar por columnas cada uno de los datos ingresados, estos se irán almacenando los datos dentro de la variable global *"descarga_df".* Finalmente, el administrador podrá visualizar todos los datos ingresados a manera de tabla y de forma detallada.

```
#AGREGAR FILAS AL EXCEL DE VENTAS DE LA TIENDA, REGISTRADO  DE VENTAS, PERTENECIENTE AL SUB_MENU_DESCARGA_ARTICULOS OPCION 2:>1
def descarga_articulos():
  global descarga_df
  dicc_fila1={}
  columnas_df=Columns(descarga_df)
  for columna in columnas_df:
    dicc_fila1[columna] = input('Ingrese valor de la columna {} :'.format(columna))
  descarga_df= descarga_df.append(dicc_fila1, ignore_index = True)
  display(descarga_df)
  #inventario_df.to_excel('/content/Inventarios_Proyect2022.xlsx')
```

5.1.2.2.3.4 Mostrar lista de artículos vendidos

Muestra la tabla relacionada con la variable *descarga_df. Y nos regresa a las opciones del trabajador de almacén.*

```
elif eleccion_descarga == "2":
  display(descarga_df)
  descarga_articulos_submenu()
```

5.1.2.2.3.5 Filtrar artículos vendidos

El trabajador puede llevar una contabilidad de todos los artículos que han ingresado y han salido (ventas) dentro del alamacén. También puede filtrarlas por fechas al igual que el administrador, aunque solo puede visualizar mas no editar el contenido ya que solo le sirve para orden y correcta comunicación con el administrador.

```
#FILTRO SALIDAS_PERTENECIENTE AL ALMACEN
def filtro_salidas():
  global descarga_df
  filtrosalidas=input("""QUE ACCIÓN DESEA REALIZAR:
  \t1 - FILTRAR
  \t2 - SALIR
  \nSELECCIONE UNA OPCION: """)
  if filtrosalidas== "1":
    print("POR EL MOMENTO SOLO SE PUEDE FILTRAR POR 'FECHA', SIGA EL FORMATO 'AÑO-MES-DIA")
    display(descarga_df.head(1))
    descarga_df.describe()
    respuesta_filtro_salida =descarga_df.loc[descarga_df['FECHA'] == input("""
    \nINGRESE LA FECHA A FILTRAR: """)]
    display(respuesta_filtro_salida)
    filtro_salidas()
  elif filtrosalidas == "2":
    descarga_articulos_submenu()
```

El programa utiliza la variable global *"descarga_df"* para filtrar solo por fecha con el objetivo de tener a la mano la visualización de un histórico de ingresos. Se creó una función llamada *respuesta_filtro_salida()* de la cual se despliega un sub menú con 2 opciones : filtrar y salir (regresar al menú anterior) en la opción 1 solo permite realizar el filtro por fecha. Utilizando el comando *"display"* se mostrará el encabezado haciendo uso de la variable global y el comando: "*descarga_df.head(1)".* El programa, dentro de una variable local llamada *"respuesta_filtro_salida"* se usa el método *".loc."* para que el trabajador de almacén filtre valores de un *data frame*, en este caso será la información de la columna *"Fecha*" bajo un input *"ingrese la fecha a filtrar*". El resultado es una tabla ordenada de manera ordenada por indexación todos los artículos ingresados en la fecha filtrada.

5.1.2.2.3.6 Salir

Nos devuelve a las opciones de almacén.

5.1.2.2.3.7 Salir

Nos regresa a las opciones de seleccionar el tipo de usuario

5.1.3 Salir

Cierra el programa por completo.

5.2 Pruebas y resultados del programa

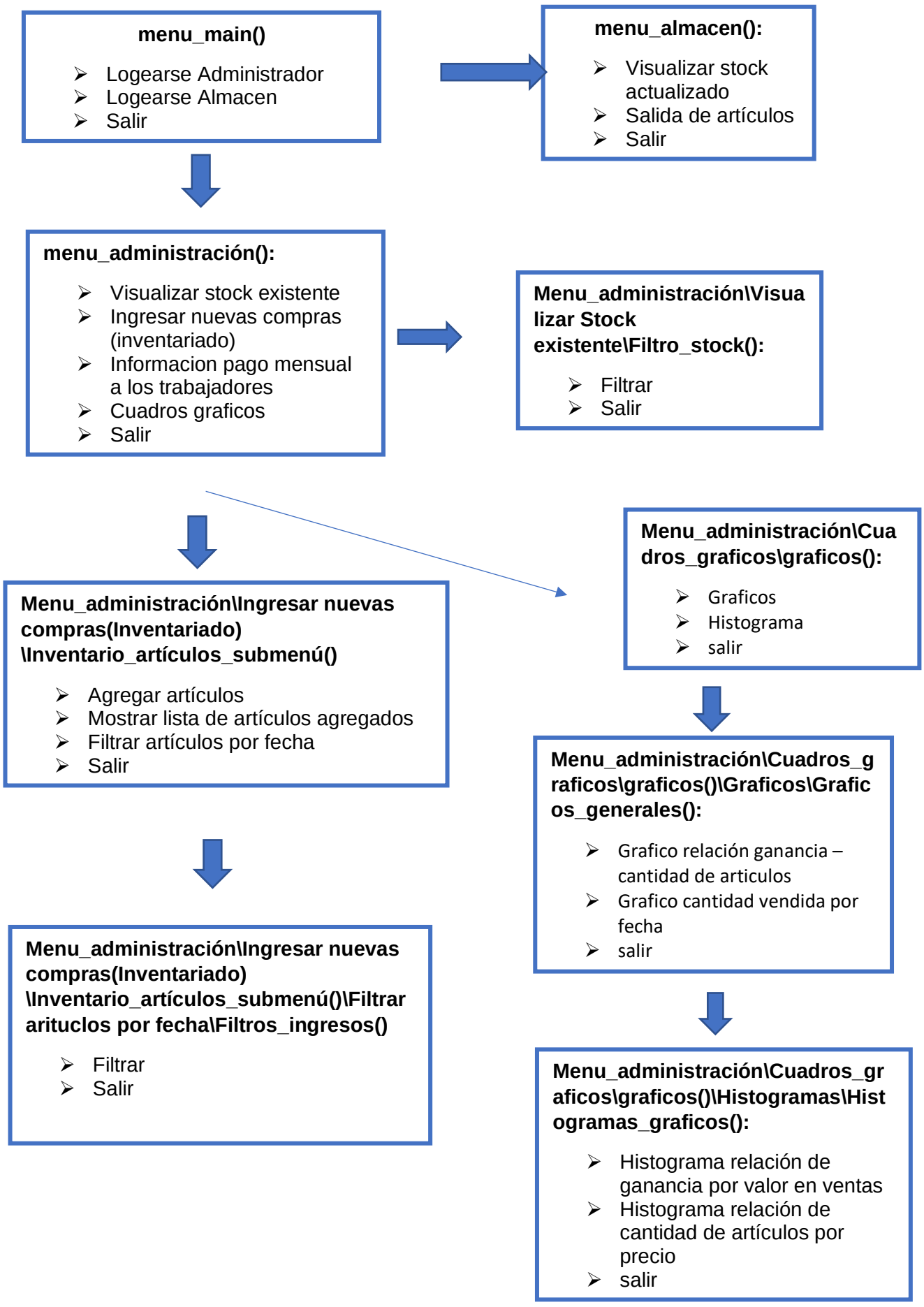
Se ha podido utilizar el programa sin inconveniente alguno utilizando las tablas Excel primigenias que ha usado Tu Tienda Perú. Las diferentes segmentaciones por trabajadores nos han permitido efectivizar el trabajo dentro del establecimiento y ha ahorrado tiempo, espacio y dinero al utilizar menos materia prima (papel) para el orden de los artículos, ventas y operaciones de los diferentes trabajadores de las dos áreas más exigentes del local.

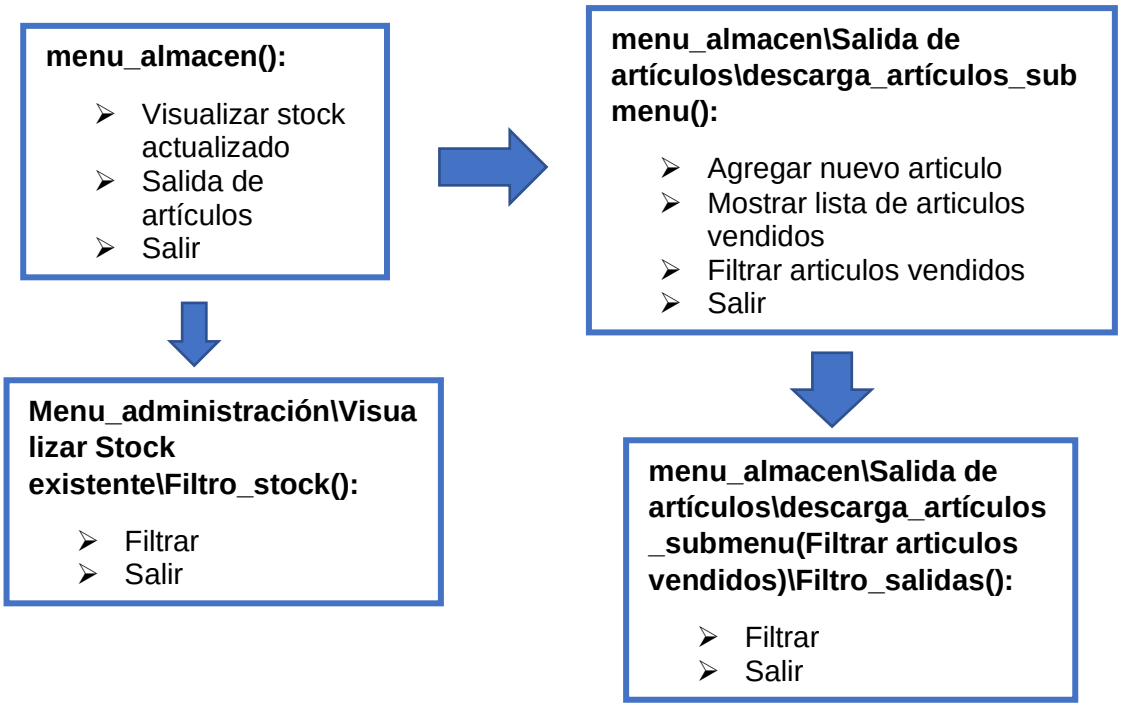
6 Resultados del uso de Python

Al ser las diferentes pruebas exitosas, podemos decir que el uso del leguaje Python nos ha servido para mejorar el trabajo interno dentro de una tienda que busca en primer paso incursionar en la digitalización de sus procesos administrativos y de contabilidad; y el uso que se le puede dar a este lenguaje de programación a nivel general y micro. Como grupo hemos explotado nuestras capacidades de análisis y lógica dentro de un nuevo lenguaje de programación, como lo es Python, y la plataforma Colab de Google.

7 Flujograma

FLUJOGRAMA: (Este flujograma muestra la visualizacion que tendra el usuario en el programa)





8 Conclusiones

Aprender el lenguaje de programación Python nos ha ayudado a comprender el uso real y flexible de sus diferentes librerías con respecto a otros lenguajes de programación. Se han aprendido el uso de librerías nuevas como la librería *getpsass* que permite encriptar información delicada y personal dentro de un programa; librerías como *numpy* para generar arrays que no es mas que coleccionar datos con dimensiones varias y así poder almacenar un sinfín de datos que ayuden a la empresa, en este caso, a crecer y trabajar con mayor holgura.

Tras la elaboración de este trabajo se ha podido comprobar que el aprendizaje de un lenguaje de programación como Python, que goza de una comunidad muy rica online, nos permite no solo trabajar correctamente en un ambiente laboral sino también, estudiantil, casero y de entretenimiento con las diferentes librerías y usos que se pueden encontrar gracias a la gran comunidad de programadores dentro de diferentes repositorios gratuitos y libres de derechos de autor.

9 Fuentes bibliográficas

Pedro Rotta. (2022a, enero). *Lectura2_Sintaxisyobjetos.ipynb* (N.o 2). Github. https://github.com/pedrorotta/PythonBasico2022/blob/main/Lectura2_Sintaxisyobjetos.ipynb

Pedro Rotta. (2022b, enero). *Lectura3_condicionales_coleccionesclase.ipynb* (N.o 3). Github. https://github.com/pedrorotta/PythonBasico2022/blob/main/Lectura3_condicionales_coleccionesclase.ipynb

Pedro Rotta. (2022c, enero). *Lectura4_Flujoycontrolclase.ipynb* (N.o 4). Github. https://github.com/pedrorotta/PythonBasico2022/blob/main/Lectura4_Flujoycontrolclase.ipynb

Pedro Rotta. (2022d, enero). *Lectura5_numpy.ipynb* (N.o 5). Github. https://github.com/pedrorotta/PythonBasico2022/blob/main/Lectura5_numpy.ipynb

Pedro Rotta. (2022e, enero). *Lectura6_Pandas.ipynb* (N.o 6). Github. https://github.com/pedrorotta/PythonBasico2022/blob/main/Lectura6_Pandas.ipynb

Pedro Rotta. (2022f, enero). *Lectura7_Matplotlib.ipynb* (N.o 7). Github. https://github.com/pedrorotta/PythonBasico2022/blob/main/Lectura7_Matplotlib.ipynb

Pedro Rotta. (2022g, enero). *Lectura8_Visualización.ipynb* (N.o 8). Github. https://github.com/pedrorotta/PythonBasico2022/blob/main/Lectura8_Visualizaci%C3%B3n.ipynb

Truzz Blogg. (2020, 3 marzo). *PROGRAMACIÓN EN PYTHON Cómo OCULTAR la CONTRASEÑA/PASSWORD introducido por consola en Python* [Vídeo]. YouTube. https://www.youtube.com/watch?v=kYgMGzcY9Oc&t=216s